

---

# A real-time network analysis tool to aid in characterizing VoIP system performance

Christopher M. White, Edward J. Daniel and Keith A. Teague

*School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, USA*

*E-mail: teague@okstate.edu*

**Abstract** This paper details the development of an educational and research tool, the Network Performance Application (NPA), which provides real-time network performance measurement through a simulated voice over IP session using live real-time data transmission over packet networks. This application, written in Visual C++ for MS Windows environments, is used to collect Quality of Service (QoS) statistics such as packet loss, end-to-end delay, inter-arrival delay jitter, and out-of-order packet delivery. The paper also demonstrates a practical application using a government standard for VoIP transmissions.

**Keywords** network modeling; network performance; Quality of Service; Voice over IP

Modern communication applications and media types have challenged the capabilities and usefulness of the analog plain old telephone system (POTS). Cellular telephones and personal computers have created demand for real-time secure multimedia communications between heterogeneous clients. This has led to a new set of application possibilities with multimedia communications, and consequently new challenges. In contrast to the POTS, the Internet is a packet data oriented communication network. A continuous stream of voice data transmitted across a POTS connection/network is broken up into discrete packets when sent across an Internet connection. Real-time voice data is expected to arrive at the destination in sequence and on a timely basis if continuous playback and low latency is to be achieved. However, packet-based transmission can disrupt data flow if packets are delayed or lost.

Unlike connection oriented networks where data travels along a dedicated channel, packet networks comprise a collection of paths linked by routers and gateways. Therefore, packets belonging to the same conversation (in an audio application) might take different paths arriving out-of-order. Packets may also experience different queuing delays at network routers causing them to arrive at their destination with varying inter-arrival delays, referred to as jitter. Congestion at network nodes (routers) may impede packets during times of high network use which may lead to packets experiencing large end-to-end delays or being discarded in an attempt to maintain sending rates. These characteristics collectively contribute to what is generally known as Quality of Service (QoS) characteristics as they describe the quality of the network medium to reliably deliver data.

## Purpose of study

This study details the design and development of an educational and research tool, the Network Performance Application (NPA), to observe and characterize the non-ideal behavior of packet networks. This application simulates communication in a voice over IP (VoIP) system over an actual network connection (or concatenation of network connections) by implementing the transfer of real-time packet data and real-time collection of statistics to characterize the QoS of the underlying packet network(s). This application is useful in predicting and modeling of network behavior as it affects a voice packet stream.

Current problems associated with VoIP using similar or mixed networks demand investigation. Error management schemes and uninterrupted transmission flow require foreknowledge of patterns of Internet behavior. This study simulates communications scenarios to qualitatively analyze the impacts of specific application requirements such as large packet size or fast send rate on overall QoS. The development of this tool also facilitates the development of an information base with which to make educated decisions during the design of an Internet VoIP communication application, or during its refinement.

## Preexisting network utilities

Several free utilities and commercial products exist for measuring aspects of network performance. These include NetWisdom,<sup>1</sup> Network Performance Monitor,<sup>2</sup> DBS,<sup>3</sup> NetSpec,<sup>4</sup> OPNET,<sup>5</sup> Ping,<sup>6</sup> netperf,<sup>7</sup> TTCP,<sup>8</sup> and others. The purpose and methods of these tools vary to suit network computing needs but they fail to provide features for measuring critical aspects of performance with respect to a VoIP system as found in NPA. Many of the tools are designed to measure large corporate network efficiency. They are not designed to measure performance for a modular real-time system across varied network segments. Ping, netperf, and TTCP are typical tools used to measure some aspects of network performance. All are free public utilities designed for UNIX systems of which some have MS Windows-based implementations.

The ping utility is a system administration tool used to check computer operation and network connection functionality. Ping is available from both DOS and UNIX terminal windows. Ping uses the Internet Control Message Protocol (ICMP) Echo function. A terminal sends a small packet to a specified IP address then listens for a return packet. The successful receipt of a packet indicates a good connection as well as provides information about the network. Timing information can be used to estimate latency on a given network path. While overall network latency information is useful, it is insufficient for analyzing QoS such as inter-arrival time, loss, and out-of-order packets. Furthermore, it does not measure the QoS response to varied packet parameters such as size and send rate.

The netperf utility is designed for client-server use on UNIX systems. Netperf is designed to use platform-dependent CPU utilization measurement options. Typically, netperf is used to measure performance of large data transfer. Such tests

measure the speed the client can send data to a server as well as the speed at which the server can receive. Request/response performance can also be investigated with netperf. Netperf request/response performance is determined by a 'transaction' for a request and response size. By definition a transaction is the exchange of a single request and a single response. The rate of transaction can be used to estimate latency. Again, similar to ping, netperf does not allow the collection of many QoS statistics.

The Test TCP (TTCP) utility runs as a command-line UNIX tool for measuring TCP and UDP performance between a client and server. TTCP operates efficiently by using a memory buffer with data which is repeatedly transmitted. The results of the transfer illustrate the approximate performance of the path between the source and destination. The size of the sent packet is variable. TTCP is used primarily to compare TCP throughput to UDP throughput and how throughput varies with changes in packet size. This utility returns kilobytes/second throughput for each packet. It does not monitor QoS statistics for a stream of packets or record information about loss, order, or inter-arrival.

All of these utilities, as well as the products listed above, fail to meet the needs for evaluating QoS and designing a VoIP system. An appropriate test application needs to exactly mimic a packaging protocol and collect statistics across the exact network(s) of interest for a VoIP system. The application must have the flexibility to accommodate varied testing parameters such as send rate, frame rate and size, and packet size to fully evaluate their effect of QoS. NPA overcomes many of these limitations.

## Internet protocols

From a desktop PC-to-PC application (which is the basis of this research) the Internet Protocol (IP) is a natural protocol of choice, given IP's availability to desktop platforms and ubiquitous presence. IP handles routing of packets from a source system, potentially through several intermediate networks, and finally to the destination end system.

There are many physical and data link protocols that can be used to transport IP traffic. One of the goals of the designers of IP was to make it portable, with the intention of the possibility of making 'IP over everything'. Networks that use IP include LANs, WANs, corporate intranets, the Internet, and cellular data networks, among others.

Voice over packet (IP) systems utilize the transport layer services to deliver end-to-end messages. These systems can employ TCP or UDP. Both of these protocols use IP at the network layer for the end-to-end delivery of messages. Delay constraints in a real-time voice application preclude the use of a connection-oriented protocol such as TCP. While TCP maintains reliable end-to-end transmission through its timeout-and-retransmissions, a single packet loss causes TCP to decrease its transmission rate with either its congestion avoidance or slow start techniques.<sup>9,10</sup> Therefore, it is difficult for TCP to provide adequate throughput with acceptable jitter over a lossy path, resulting in large delay.

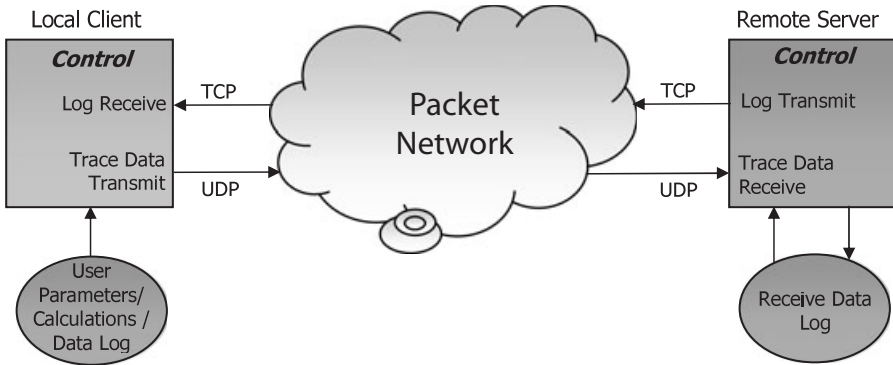


Fig. 1 NPA network protocol interaction.

UDP, in contrast to TCP, provides connectionless transmission of stream data. UDP however does not provide reliable service; it is a ‘best effort’ type of service. There are neither acknowledgments sent nor retransmission of data. It is up to the end user to provide this functionality along with a means of re-sequencing out-of-order packets. UDP is a good candidate for transmission of real-time voice data in a VoIP system because it provides efficient, although unreliable, transport of data.

NPA is designed to simulate VoIP traffic between any two network endpoints capable of supporting an IP connection. The network can consist of anything from a simple LAN connection to concatenations of widely dissimilar networks, including both wired and wireless media. NPA simulates a VoIP session by sending a specified number (or time interval) of fixed-size network packets consisting of an integer number of simulated VoIP frames and headers, at a specified rate between the two endpoints, labeled client and server in Fig. 1. UDP is used exclusively for transporting simulated VoIP packets. The client and server track the transmission and delivery of each packet using precision timing information provided by NTP. At the conclusion of the test session, the server returns to the client a detailed transaction log of the simulated session using TCP. The log is used to perform offline statistical calculations that provide detailed network performance data for the duration of the test. Figure 1 illustrates the signaling and data communication between the server and client.

### Quality of Service (QoS)

Delay, loss, and inter-arrival time between packets constitute the major factors affecting the Quality of Service (QoS) in a system that uses packet networks (IP) to transmit real-time voice data. Excessive packet delay can occur when the queue in a router becomes full and subsequent packets entering the queue are delayed. Large end-to-end delay (latency) causes speech overlap as well as mutual silence at the receiver system. Large packet inter-arrival delays (jitter) cause packets to arrive too late for their designated playback time, which forces the receiver to discard the late

packets. If there are large queuing delays, consecutive packets may cluster at the nodes and be delivered almost simultaneously to the receiver. As a result, the packets received may exceed the capacity of the playout buffer causing an overrun. In addition to late packet loss at the application, loss can occur as a result of congestion at routers, where packets are discarded because the router can not service all the packets in its queue. Congestion at routers can result in consecutive packet loss, known as burst loss. Burst loss can have a large impact on system performance, potentially leaving large gaps in audio output. Burst loss is difficult to correct in packet audio applications. Congestion may also cause subsequent packets to take an alternate path to their destination. Packets may encounter different delays as they take different paths as some nodes may have varying queuing delays (wait times) along the path, possibly arriving at their destination out-of-order. If uncorrected, this could cause garbled speech output at the receiving system.

The audio output of the communication system can be seriously degraded resulting in low overall QoS if there are no mechanisms in place to manage and correct for the aforementioned QoS issues. Understanding the frequency and probability of these issues offers insight into preventative measures that will increase the system QoS in the presence of network anomalies. Furthermore, as modern communication moves toward packet network transmission, there is potential for aggregate concatenations of dissimilar networks where non-ideal behavior becomes more complex, particularly with error prone packet wireless networks. NPA provides a mechanism for providing a better understanding of real network behavior and QoS through collection of empirical data over a real network connection in a process that closely simulates an actual VoIP application.

## NPA application design

### Architecture

#### *Programming structure*

NPA is written in the C++ programming language and is designed for implementation in MS Windows environments. C++ was chosen because of the nature of MS Windows network programming, the need for multitasking and precision timing, and the extensive literature base in the Microsoft Foundation Class (MFC) libraries. NPA has a graphical user interface (GUI) that allows control either by mouse or keyboard in any Windows operating system released later than 1998. Multitasking is accomplished by using a multithreaded architecture to coordinate different processing events. NPA uses eight threads: *control*, *TCP*, *UDP*, *filewrite*, *NTP*, *scheduler*, *statcalc*, and *main window*. The threads are given different priorities for processing resources ranging from normal to time critical to ensure accurate timed events. Each event is supplied as a service to make the program efficient and reliable.

The *control thread* dominates the flow of control by managing and creating other threads and handling events that occur throughout the course of a transmission. The *TCP thread* controls the TCP connection and initial interaction between client and server. It transfers network setup information for a UDP connection that will simu-

late a VoIP connection. The *UDP thread* controls the simulated VoIP data packet transmission; *filewrite*, *scheduler*, *statcalc*, and *main window* threads coordinate the capture, storage, and display of statistical information. The *NTP thread* ensures synchronization and precision for time-dependent statistics.

### *Timing methodology*

A precise system clock is required at both the client and the server in order to provide time stamps with millisecond accuracy for both packet transmission and reception. This is easily provided by the real-time clock present in most personal computers. However, in order to provide accurate end-to-end time measurement (packet latency), the clock at each endpoint must be precisely synchronized. The Network Time Protocol (NTP) is used to synchronize the time of the client and server clocks to a reference time source. It provides millisecond accurate time control on LANs relative to Coordinated Universal Time (UTC) via a Global Positioning System (GPS) receiver. The NTP configuration uses redundant servers and dissimilar network paths to ensure accuracy and consistency.

NTP was specifically selected for the accuracy in timing that it permitted. Traditional MS Windows timers do not provide accurate timing beyond the millisecond range, and tend to drift an excessive amount over even relatively short time intervals. When taking measurements that deal with sub-millisecond events, such error and drift would eliminate validity of the experiment. NTP constantly updates a computer clock by polling to reference clocks that also poll to a hierarchy of clocks. Over a period of time this structure effectively establishes and maintains consistent timing accuracy. Calibration algorithms for NTP developed by Dr David Mills of the University of Delaware<sup>11</sup> are used to provide millisecond accuracy in this application. By having both the client and server running NTP, calculations that involve both sending and receiving time can be relied upon typically to an accuracy of approximately one millisecond or better, and calculations involving one computer can give less credence to the possibility of drift. NTP creates a log that shows the clock offset, the drift compensation, and other information including the corrections that have been made providing further evidence of reliability.

### Flow control

#### *Graphical user interface (GUI)*

NPA begins by opening a dialog box that contains a status window and button to allow the user to enter connection parameters, as shown in Fig. 2. Choosing to enter parameters opens a second dialog box of parameters requiring information to proceed, shown in Fig. 3. The user first chooses to be a listener or a sender, disabling the other option upon selection. Choosing 'sender' enables the selection of parameter options to tailor each simulation. The client must enter an IP address, TCP port number, UDP port number, packet or frame size, send rate, and test run length. Small message boxes appear when the user fails to enter critical information such as an IP address, or if the information entered does not match the type or length required. Upon entering all the parameters, the user has the option to schedule the

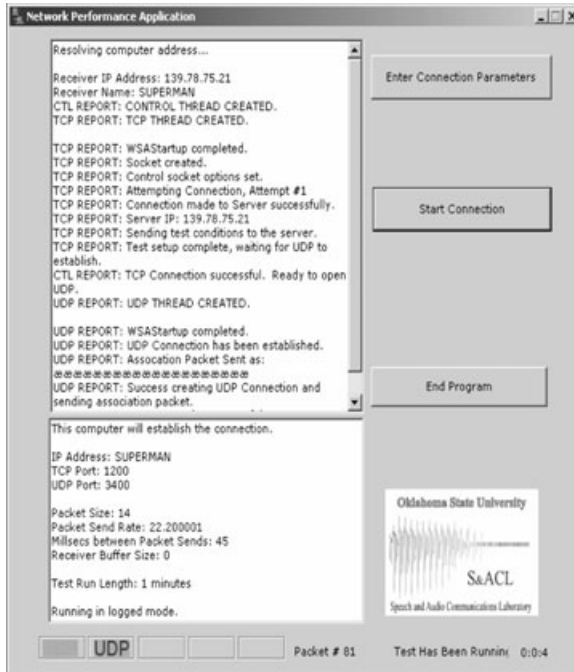


Fig. 2 Graphical user interface.

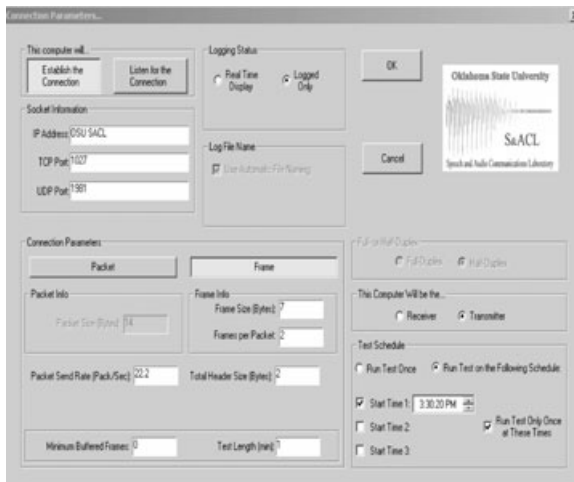


Fig. 3 Connection parameters.

test run or begin a connection immediately. By selecting the scheduler option the user can choose up to three test runs and specify the times they occur. The listener merely responds to a connection initiation and loops back into a listening state as soon as the session ends allowing multiple remote tests to be completed without anyone being present at either terminal. This allows the server application to run ‘stand alone’ at a remote location.

*The application*

The user initiates the connection and the *control thread* creates a *TCP thread* which begins network negotiation. When the remote listener acknowledges the TCP request and sends back acceptance of the connection and UDP port number, the *control thread* creates a *UDP thread* to begin data transfer. A randomly generated symbol becomes association packets that are sent from the client to the server using the connectionless UDP channel. The test session begins once the association packets are successfully transmitted and received over the UDP channel. Throughout data transfer, two data logs are simultaneously being created in each terminal. These logs record the sequence number of each packet, and the send and receive times in seconds and microseconds. The data logs also contain connection information such as time of day, IP address, as well as the user entered parameters including frames per packet and send rate. This process is illustrated in Fig. 4.

At the end of the session the server application shuts down the UDP thread and waits for acknowledgment via the TCP of connection termination (or all data received) from the client application channel. Then the server transfers its log file on the TCP channel to the client. The *statcalc* (statistics calculation) *thread* takes both log files and performs calculation of packet loss, latency, inter-arrival time, receive buffer state, and packets received out-of-order. The *statcalc thread* stores the results in logs which are comma separated value (CSV) documents. This file format allows the information to easily load into other graphing and data manipulation tools such as MATLAB. The flow of control for NPA is shown in Fig. 5.

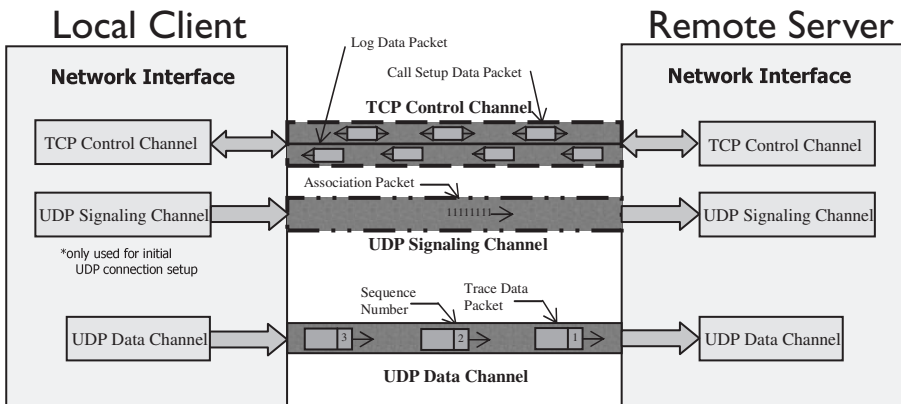


Fig. 4 Detailed operation.

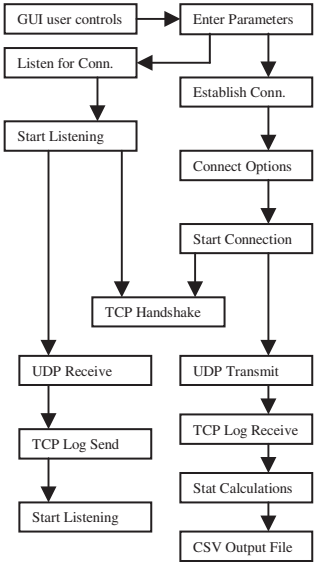


Fig. 5 NPA flow of control.

NPA data output format

The output file format of the data log recorded by NPA is eight columns of data containing: packet number, inter-arrival time, latency, loss, out of order, and three columns corresponding to receiver buffer state (currently implemented as a fixed length buffer).

A single clock measures the arrival of successive packets. The measurement of latency requires each clock’s time stamp for a packet. Missing and out-of-order packets are calculated by checking for gaps in the sequence of received packet numbers. The receive buffer statistics calculations entail the use of a fixed length buffer to determine how various receiver buffer sizes would perform under particular network conditions. If delays are larger than the fixed buffer length a buffer ‘*under-run*’ is logged; if data has ‘*built up*’ in the network due to router congestion and packets arrive close together where the receiver buffer length is too small to accommodate the data, an ‘*overrun*’ is logged.

Data analysis using MATLAB

A MATLAB script is used to read in the NPA output CSV file, calculate, and display the information graphically. The script parses the columns of data within the CSV file, then processes and graphs the information for a more illustrative representation (Figs 6 and 7). In addition to graphing the information, the script performs statistical calculations, taking the inter-arrival delay and end-to-end latency data and calculating the mean and standard deviation. The loss data is used to calculate burst loss and total packet loss rate. The MATLAB script also estimates and displays the resulting probability density functions for delay and loss.

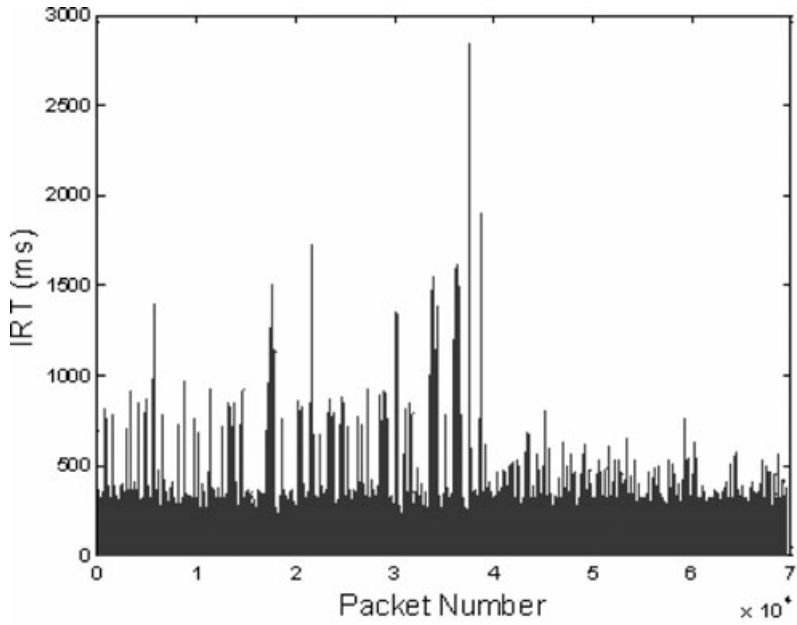


Fig. 6 Inter-arrival time CDMA to LAN.

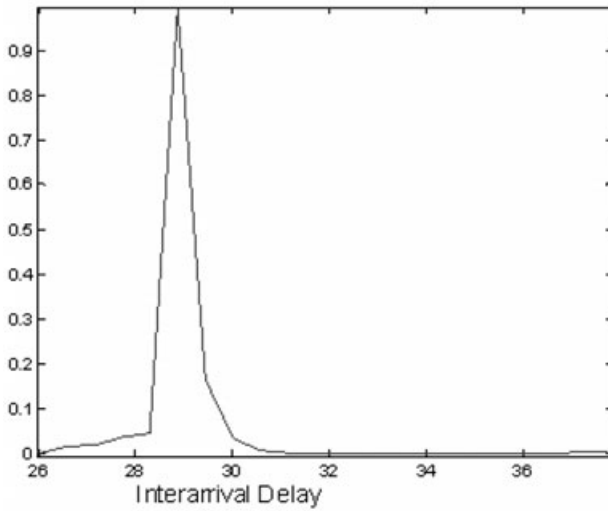


Fig. 7 Frequency distribution of inter-arrival time CDMA to LAN.

## Collected statistics

### FNDBT using MELP simulation

As an example of how NPA can be used to characterize the performance of a VoIP system, we examine the effects of a concatenation of Internet and CDMA cellular data networks for secure voice communications using the federal standard Mixed Excitation Linear Prediction (MELP) codec<sup>12</sup> in a VoIP application. MELP is the current military standard (MIL-STD-3005) for low-rate voice communication. MELP is a linear predictive vocoder that operates at 2400 bps. MELP requires input frames consisting of 180 samples (16-bits per sample) of raw speech at 8000 samples per second, for an effective frame length of 22.5 ms. To achieve 2400 bps, MELP produces output parameters that are packed into 54 bits per frame giving approximately a 53:1 compression ratio and 44.4 frames per second. The parameterized output frames MELP produces to represent speech include: Line Spectral Frequencies (LSF), Fourier magnitudes, gain (2 per frame), pitch, overall voicing, band-pass voicing, aperiodic flag, error protection, and sync bit. This study provides insight to potential performance issues the government's future secure communications protocol may encounter when deployed over mixed packet networks using the MELP codec for low rate secure voice communication.

The CSV output file from NPA provides all the needed information for packet network data analysis. NPA simulates the transfer of secure voice (MELP) data over packet networks following the government's secure protocol. This enables study of the effects different network characteristics have on the performance of the protocol when this system is deployed over packet networks. With NPA we are able to simulate sending packet sizes with multiples of 7 byte frames, secure MELP crypto sync frame sizes, at multiples of the MELP frame interval of 22.5 ms. Simulations can be compared to real time voice transmissions in lossy and lossless environments, and results can be used for comparison and verification with previously published studies.

### Data analysis

Data collected from NPA, shown in Figs 6 and 7, show the inter-arrival delay during a mixed network connection between a CDMA cellular data connection and a computer on a local area network simulating secure MELP (Fig. 8). The mean inter-arrival time for the test was 52.35 ms with a standard deviation of 77.95 ms. This indicates that the average delay was higher than the predicted 45 ms packet send time due to the nature of mixed network communication. Also, the high standard deviation indicates long delay bursts. Figure 7 illustrates that while the mean was 52.35 ms the mode was near 29 ms which shows that the large bursts of delay are followed by quick bundles of packets that arrive sooner than the send time so that the average inter-arrival time is lower. Figure 9 shows the frequency distribution of consecutive packet loss. The total packet loss rate is 13.9% and the mean burst loss is 1.06 packets, which indicates 1 to 2 consecutive packet losses occur most often. This is evident in the graph as the slope decays geometrically fast from the origin (1 packet). This information provides significant insight into overall network

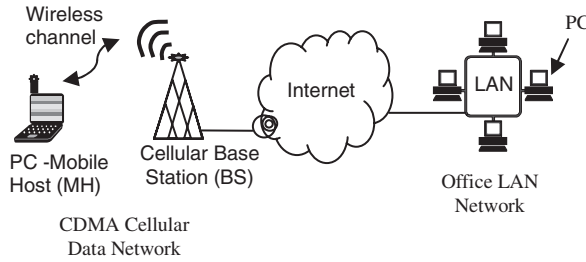


Fig. 8 Mixed network communication.

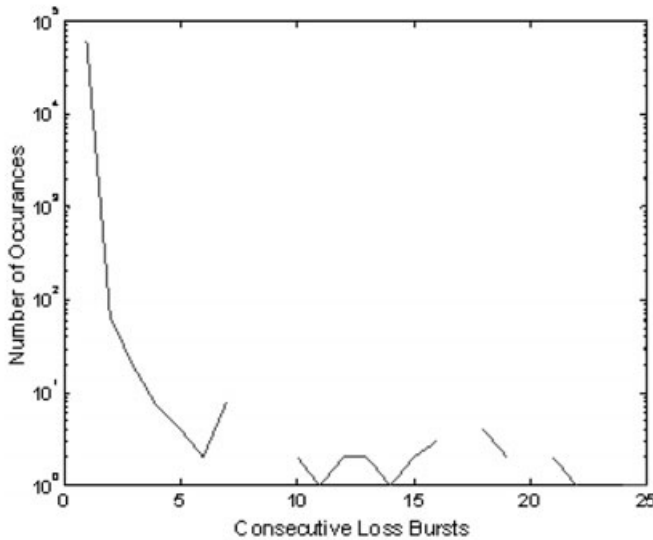


Fig. 9 Frequency distribution of consecutive burst loss.

performance, and can be used to gain additional understanding of the types of errors that a user will observe using such a VoIP system. Data from such a real-time simulation provides information that can be used to design more effective VoIP applications through techniques for overcoming or masking some of the errors that may occur.

### Conclusions

NPA is a real-time application useful for evaluating actual network performance for VoIP simulations. One important aspect of NPA is the ability to simulate various data streams, representing different communications standards, formats, and types. NPA is written in Visual C++ using MFC for MS Windows environments. The program runs as a server/client pair over two computers connected via one or more

IP networks. Network Time Protocol (NTP) is used to synchronize each PC for accurate time measurement of packet arrivals. The user specifies transmission parameters such as packet send rate, frames per packet, packet size, etc. NPA initiates a connection, records all packet sending / arrival times, generates a data log, and performs statistical calculations.

The parameter settings within NPA can be adjusted to monitor the performance for an arbitrary voice codec. It can imitate the transmissions for nearly any voice codec. With the quality of the codec set as the baseline for performance, understanding the QoS statistics from a given network on the codec data stream becomes very constructive. This information can help in deducing problems with the performance of a particular codec as well as contribute to a decision about whether a codec is sufficiently fit for a packet network environment. Within educational settings this can demonstrate variable quality of different codecs given a network environment.

Many communication parameters have proportionate effects on the probability of error, and can be better understood if they can be changed when simulating a VoIP system. NPA, a multimedia over IP simulator, allows user-specification of the major parameters: packet size, frames per packet, and send rate. These options establish different communication scenarios that allow statistical analysis of packet network communication. NPA records and calculates major characteristic information: packet number, inter-arrival time, latency, loss, out-of-order packets, and a fixed buffer state. It produces statistics for analysis including mean, standard deviation, loss rate, burst loss rate, etc. NPA provides information that can be used to develop more robust VoIP systems. This could include, for example, the development of dynamic buffer adaptation, loss recovery techniques, and congestion / error rate control algorithms.

## References

- 1 <http://www.finisar.com/nt/netwisdom.php>
- 2 <http://www.solarwinds.com>
- 3 <http://www.kusa.ac.jp/~yukio-m/dbs/goal.html>
- 4 <http://www.tisl.ukans.edu/Projects/AAI/products/netspec.old/>
- 5 [http://www.opnet.com/services/consulting/enterprise\\_services.html](http://www.opnet.com/services/consulting/enterprise_services.html)
- 6 <http://www.ping127001.com/pingpage.htm>
- 7 <http://www.netperf.org/netperf/training/Netperf.html>
- 8 <http://www.clarkson.edu/projects/itl/HOWTOS/PCATTCP-jnm-20011113.htm>
- 9 W. R. Stevens, *TCP/IP Illustrated*, Vol. 1 (Addison Wesley, New York, 1994).
- 10 M. Borella, D. Swider, S. Uludag and G. Brewster, 'Internet Packet Loss: Measurement and Implication for End-to-End QoS', *Proc. IEEE ICPP Workshop '98*, pp. 3-12, 1998.
- 11 D. Mills, 'Internet Time Synchronization: the Network Time Protocol' *RFC-1129* (Internet Engineering Task Force, University of Delaware, Oct. 1989).
- 12 L. Supplee, R. Cohn and J. Collura, 'MELP: The New Federal Standard at 2400bps', *IEEE International Conference on Acoustic, Speech, and Signal Processing ICASSP'97*, pp. 1591-4, 1997.