1. Introduction

1.1 Basic Introduction

Voice over Internet Protocol (VoIP) is one of the most important technologies in the World of communication. VoIP is simply a way to make phone calls through the internet. VoIP transmits packet via packet-switched network in which voice packets may take the most efficient path. On the other hand, the traditional public switched telephone network (PSTN) is a circuit-switched network which requires a dedicated line for telecommunications activity [1]. Furthermore, Internet was initially used for transmit data traffic and it is performing this task really well. However, Internet is best-effort network and therefore it is not sufficient enough for the transmission of real-time traffic such as VoIP [2].

In addition, there are about 1 billion fixed telephone lines and 2 billion cell phones in the World that use PSTN systems. In the near future, they will move to networks that are based on open protocols known as VoIP [3]. That can be seen from the increasing number of VoIP users. For instance there are more than eighty million subscribers of Skype; a very popular VoIP commercial application [4]. VoIP has gained popularity due to the more advantages it can offer than PSTN systems especially that voice is transmitted in digital form which enables VoIP to provide more features [5]. However, VoIP still suffer few drawbacks which user should consider when deploying VoIP system [6].

Basically, VoIP system can be configured in these connection modes respectively; PC to PC, Telephony to Telephony and PC to Telephony [7]. VoIP consists of three essential components: CODEC (Coder/Decoder), packetizer and playout buffer [8], [9]. At the sender side, an adequate sample of analogue voice signals are converted to digital signals, compressed and then encoded into a predetermined format using voice codec. There are various voice codecs developed and standardized by International Telecommunication Union-Telecommunication (ITU-T) such as G.711, G.729, etc. Next, packetization process is performed which fragment encoded voice into equal size of packets. Furthermore, in each packet, some protocol headers from different layers are attached to the encoded voice. Protocols headers added to voice packets are of Real-time Transport Protocol (RTP), User Datagram Protocol (UDP), and Internet Protocol (IP) as well as data link layer header. In addition, RTP and Real-Time Control Protocol (RTCP) were designed at the application layer to support real-time applications. Although TCP transport protocol is commonly used in the internet, UDP protocol is preferred in VoIP and other delay sensitive real-time applications. TCP protocol is suitable for less delay-sensitive data packets and not for delay-sensitive packets due to the acknowledgement (ACK) scheme that TCP applies. This scheme introduces delay as receiver has to notify the sender for each received packet by sending ACK.

The packets are then sent out over IP network to its destination where the reverse process of decoding and de-packetizing of the received packets is carried out. During the transmission process, time variations of packets delivery (jitter) may occur. Hence, a playout buffer is used at the receiver end to smoothen the playout by mitigating the incurred jitter. Packets are queued at the playout buffer for a playout time before being played. However, packets arriving later than the playout time are discarded. The principle components of a VoIP system, which covers the end-to-end transmission of voice, are illustrated in Figure 1.

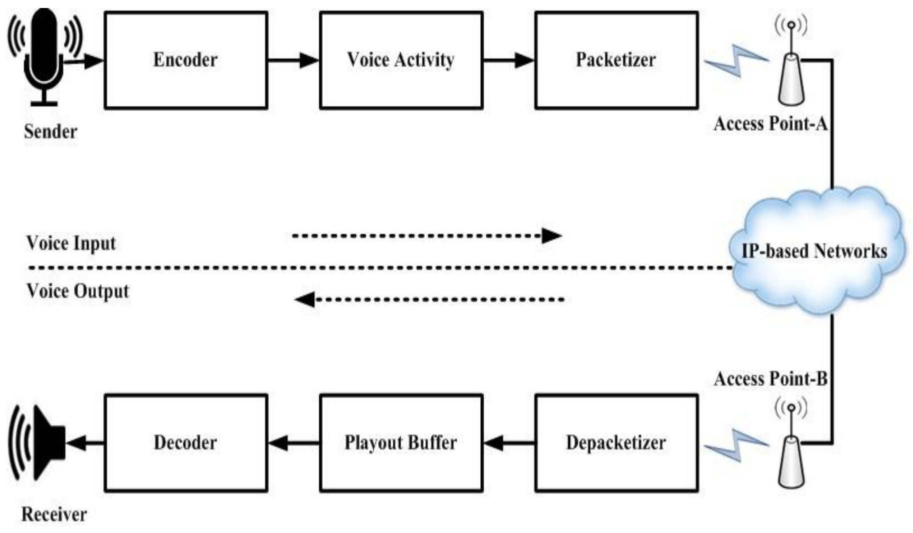

Figure 1. Basic VoIP components

For this project we are using NS2 to implement our VoIP network. We will be setting up several nodes on either side of two routers. We are going to be using an exponential traffic source to re-create a typical voice conversation over VoIP. Different protocols will be used to send voice information between terminals, starting with User Datagram Protocol (UDP), Transmission Control Protocol (TCP), and Realtime Transport Protocol (RTP). We are going to be measuring packet loss, throughput, end-to-end delay, and if the protocol permits, jitter. We will be plotting our results and comparing them to our theory based predictions.

2. Literature Survey and Background

2.1 Background

2.1.1 VoIP Advantages

VoIP offers several advantages over traditional circuit switching networks. These include:

Low cost: Utilizes the widespread availability of Internet to make phone calls using standard, readily available computer parts.

Integrate different web services with VoIP: An example would be calling a retail establishment while online shopping directly from your computer to make product inquiries.

Possibility for greater bandwidth efficiency: Due to the vast amounts of codecs available for VoIP networks (which may also be available to circuit-switching networks but are not implemented due to cost restrictions), voice data may be transmitted at rates different from the standard 64kbps, including variable bit rates [1].

2.1.2 Voice over TCP

Since IP is a best-effort protocol, it does not prevent errors such as packet loss from occurring. The Transmission Control Protocol (TCP) serves to mitigate this flaw by detecting errors, retransmitting lost packets, and ensuring data is ordered properly in order to provide reliable service. TCP is a connection oriented protocol that lies in the Transport layer. The connection is formed via a handshake between two hosts with connection requests and acknowledgments. Once the connection is formed, the data being transmitted is broken into segments. Before the segment is transmitted, a header is attached which contains a sequence number. The receiver will respond to the arriving packet with an acknowledgement if no errors are found. If no acknowledgement arrives at the original sender after a certain timeout period, the sender will re-transmit the packet [4].

Despite its reliable service, the main drawbacks of TCP are the long delays inherent in taking such preventative measures. One of the main challenges of VoIP is the delay restriction in a real-time phone call. According to the ITU-T Recommendation G.114, one-way delay should be no more than 400 milliseconds for international calls [5]. However, maintaining constant high quality audio during the phone call is also an important aspect – one that is challenged by packet loss and errors during transmission. In general, large delays are more undesirable than data loss with regards to voice communications due to the "real-time" aspect. Investigation of a TCP-based VoIP network is necessary to determine how much of a benefit the added reliability is at the cost of longer delay times.

2.1.3 Voice over UDP

The UDP is a simple protocol that passes data along from the application layer to IP to be transmitted. It performs none of the error checks that TCP does, and is therefore unreliable. A UDP header merely consists of an optional source port, a destination port, the length of the datagram, and a checksum [4]. As previously mentioned, the main reason for using UDP over TCP in VoIP applications is the reduced delay. In general, the sporadic loss of packets in a conversation will not be as disruptive as excessively long delay times. In fact, a packet loss of about 5% is said to be tolerable depending on how the losses are distributed [1]. We will investigate how well a UDP-based VoIP network performs in contrast to its TCP counterpart.

2.1.4 Voice over RTP

The RTP is an application layer protocol that attaches itself to UDP to provide added benefits for realtime applications. An RTP header includes a sequence number to help preserve the order of the transmitted packets. It also includes a timestamp, which is meant to provide information to the destination application so that it may compensate for problems such as delay or jitter if they arise. The optional companion protocol, RTCP (specified in RFC 3550), is used as a means of exchanging information on session quality, which can include the number of lost packets or the average delay time [2]. RTP strikes a balance between UDP and TCP for VoIP applications. It is designed to operate rapidly like UDP and it provides the receiver with valuable information pertaining to the VoIP session. The receiving application can then use this information to alleviate problems caused by out-of-sequence packets and jitter, thus improving the quality of the session. RTP is the protocol of choice for streaming media over the Internet and is widely used in VoIP applications [2].

2.2 Basic Working of VoIP

VoIP stands for Voice over IP (Internet Protocol), a variety of methods for establishing two-way multi-media communications over the Internet or other IP-based packet switched networks. Although VoIP systems are capable of some unique functions (for example: video conferencing, instant messaging, and multicasting), this appendix concentrates on the ways in which VoIP can be used to replicate the voice conversation functionality of the public switched telephone network (PSTN). There are several competing approaches to implementing VoIP. Each makes use of a variety of protocols to handle signalling, data transfer, and other tasks. Data is moved between the two endpoints using a media protocol, the Real-time Transport Protocol (RTP). A codec (coder/decoder) is used to convert the sound of each caller's voice to digital data, then back to analog audio signals at the other end. Conversation ends and the call is torn down. Again, this involves the signalling protocols appropriate to the particular implementation of VoIP, along with any Gateway or Gatekeeper functions. The instructions governing the call-the call setup and call teardown-are handled separately from the transmission of the actual data content of the call, or the encoding and packetization of voice media. There are several protocols and methods for VoIP calls – the commonest standards are termed SIP and H.323 – but they all have some basic features in common. To the user phone calls are made and handled in the same way as they always have been except that VoIP phones often have more features available from menus and buttons than regular phones. When a call is dialed, the system takes the phone number, connects over the local network to whatever system is providing service. That system figures out if the call needs to go into the regular phone network and if so switches it to a gateway that connects the call over the regular phone network. If the call can be completed without going over the regular phone network (the number dialed is also a VoIP system) then the provider system will route the call directly, performing protocol translation (to a different kind of VoIP) if needed. When traveling on the network, VoIP calls are treated like any other network data – they are broken down into little pieces of digital information (packets) and sent by whatever route the network determines to be fastest. That means different pieces arrive and different times and out of order and then are reassembled back into the proper sequence at the destination. This is why the 100+ kbps transmission rate is needed – so that the signal can be sent and reassembled quickly enough so that human users on both ends don't notice any delay. It is also one of the weaknesses of VoIP – if the network goes down or has performance issues, so will your VoIP calls. Since the very early days of distance communication, signals were sent in analog form, in waves. Many years ago, the communication world discovered that sending a signal to a remote destination could have been done also in a digital fashion: before sending it we have to digitalize it with an ADC (analog to digital converter), transmit it, and at the end transform it again in analog format with DAC (digital to analog converter) to use it. VoIP works like that: digitalizing voice in data packets, sending them and reconverting them in voice at destination. Digital format can be better controlled: we can compress it, route it, convert it to a new and better format, and so on. We also saw that a digital signal is more noise-tolerant than its analog version.

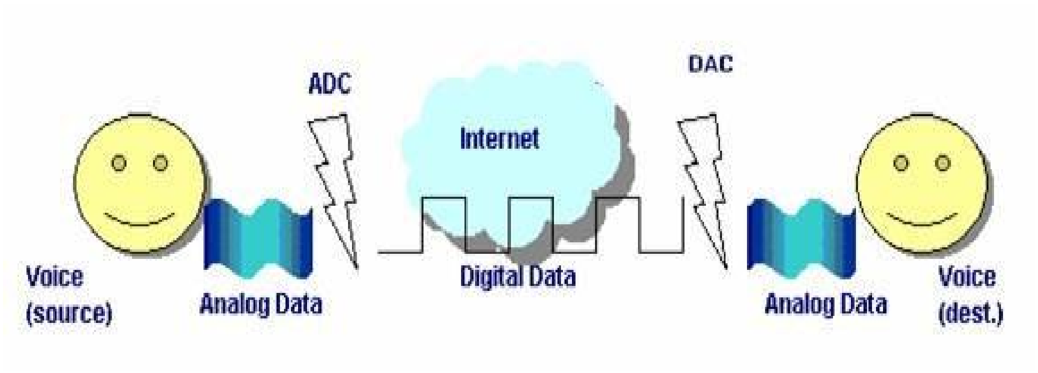

Figure 2.1. Basic Working of VoIP

2.3 Traffic Detection in VoIP

4 techniques are used to detect VoIP traffic:

- Port based technique
- Signature based technique
- Pattern based technique
- Statistical analysis based technique

Steps involved in VoIP call setup are signalling and media channel setup. Signalling is used to setup connection between communicating parties. Media channel setup is the actual voice transmission channel between two parties after a successful signalling. Media channel setup includes: digitization, encoding, packetization and transmission of voice packets over packet switched networks. SIP and H323 are mostly used signalling protocols and RTP is mostly used media transmission protocols. Some of detection techniques are applied on signalling traffic, some on media traffic. Use of complex encryption and tunnelling mechanisms for VoIP makes traffic detection very difficult.

2.4 VoIP packet structure

A VoIP packet is composed of the IP header, followed by the UDP header, followed by RTP header, and finally followed by the payload

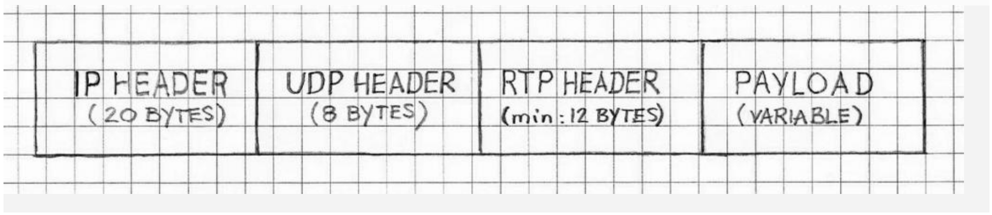

Figure 2.2. Structure of the VoIP packet (as in IPv4)

2.5 Voice Codecs

A codec is an algorithm most of the time installed as a software on a server or embedded within a piece of hardware (ATA, IP Phone etc.), that is used to convert voice (in the case of VoIP) signals into digital data to be transmitted over the Internet or any network during a VoIP call.

The word codec comes from the composed words coder-decoder or compressor-decompressor. Codecs normally achieve the following three tasks (very few do the last one):

- Encoding - decoding
- Compression - decompression
- Encryption – Decryption

2.5.1 Common VoIP Codecs

Codec	**Bandwidth/kbps**
G.711	64
G.722	48/56/64
G.723.1	5.3/6.3
G.726	16/24/32/40
G.729	8
GSM	13
iLBC	15
Speex	2.15 / 44
SILK	6 to 40

Table 1. Bandwidth/kbps of different codecs

3. Work Carried Out

3.1 About NS2

NS2 is an open-source simulation tool that runs on Linux. It is a discreet event simulator targeted at networking research and provides substantial support for simulation of routing, multicast protocols and IP protocols, such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks. It has many advantages that make it a useful tool, such as support for multiple protocols and the capability of graphically detailing network traffic. Additionally, NS2 supports several algorithms in routing and queuing. LAN routing and broadcasts are part of routing algorithms. Queuing algorithms include fair queuing, deficit round-robin and FIFO.NS2 started as a variant of the REAL network simulator in 1989 REAL is a network simulator originally intended for studying the dynamic behaviour of flow and congestion control schemes in packet-switched data networks. Currently NS2 development by VINT group is supported through Defence Advanced Research Projects Agency (DARPA) with SAMAN and through NSF with CONSER, both in collaboration with other researchers including ACIRI (see Resources). NS2 is available on several platforms such as FreeBSD, Linux, SunOS and Solaris. NS2 also builds and runs under Windows. Simple scenarios should run on any reasonable machine; however, very large scenarios benefit from large amounts of memory. Additionally, NS2 requires the following packages to run: Tcl release 8.3.2, Tk release 8.3.2, OTcl release 1.0a7 and TclCL release 1.0b11.

3.1.2 Tcl

Tcl is shortened form of **Tool Command Language**. John Ousterhout of the University of California, Berkeley, designed it. It is a combination of a scripting language and its own interpreter that gets embedded to the application, we develop with it. Tcl was developed initially for Unix. It was then ported to Windows, DOS, OS/2, and Mac OSX. Tcl is much similar to other unix shell languages like Bourne Shell (Sh), the C Shell (csh), the Korn Shell (sh), and Perl. It aims at providing ability for programs to interact with other programs and also for acting as an embeddable interpreter. Even though, the original aim was to enable programs to interact, you can find full-fledged applications written in Tcl/Tk.

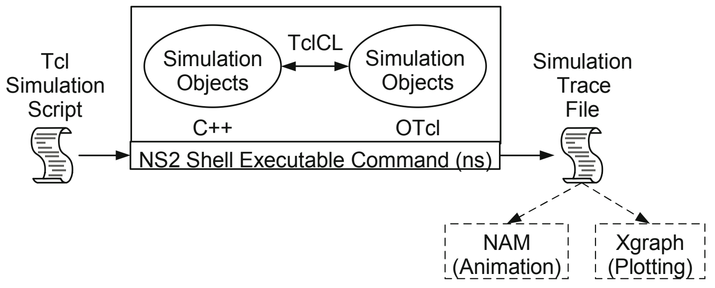

Fig 3.1. Basic Architecture of NS2

3.2 Implementation

In a standard circuit-switching network, an analog voice signal must be sampled at twice its maximum frequency at 8 bits per sample. Standard human speech reaches about 4000kHz, thus a bandwidth of 64kbps is required. The advancement of codec technology has improved bandwidth efficiency in telephony by only transmitting information when a person is talking [1]. Therefore, a variable bit rate on each end is required to accurately simulate a VoIP call. In our implementation, we assume the commonly used G.711 codec, which transmits information at a rate of 64kbps [2]. The size of the transmitted packets was chosen to be 128 bytes for UDP. TCP payloads are 1040 bytes and the ACK packets are 40 bytes. The phone call is established between Nodes 0 and 4. Node 0 transmits data with an average “on” time of 1200ms and idle time of 800ms. Node 4 is setup to transmit fewer packets over the 60 second simulation with an average “on” time of 800ms and idle time of 1200ms. The scale of the simulation was chosen to be nation-wide, thus the link between routers represented by Node 8 and 9 was chosen to be an Optical Carrier Level-1 (OC-1) line, which has a bandwidth of 51.84Mbps [4]. The link used in the simulation is a duplex link and thus has its bandwidth set to 25.92Mbps to properly mimic the OC-1 and a delay of 35ms.

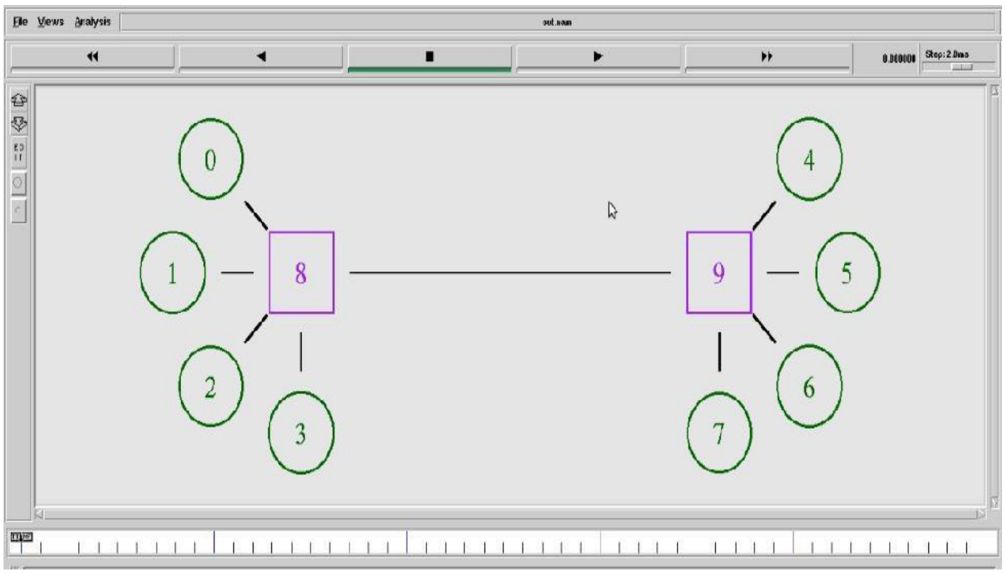

Figure 3.2: NS-2 implementation of a VoIP network

The background traffic of the network is supplied by Nodes 1, 2, 3, 5, 6, and 7 at constant bit rates. As the simulation begins, Nodes 1 and 5 create background traffic at a rate of 25.89Mbps each (while all other background sources are turned off), providing a sub-maximal load for the duplex link between the two routers. Then from 20s to 40s, Nodes 2 and 6 are turned on (while Nodes 1 and 5 turn off) to provide background traffic of 25.91Mbps each. This was chosen to slightly overload the link's capacity and thus cause congestion within the network. Finally, when the simulation reaches 40 seconds, Nodes 3 and 7 are tasked with providing the background traffic at a rate of 25.92Mbps each, greatly exceeding the network's bandwidth. At this point, it is expected that the queues become full and resulting in many dropped packets.

Depending on our simulation, UDP, TCP, or RTP agents were attached to Nodes 0 and 4. For the background traffic, we used UDP for all three scenarios. Furthermore, Traffic from Node 0 to Node 4 is colour coded as blue, whereas traffic from Node 4 to Node 0 is colour coded as red. The simulation can be visualized in NAM.

3.2.1 Filtering the Trace File with AWK

The NS2 code opens an output trace file, *out.tr*, and records every single event of the simulation for each individual packet, such as entering queues or being dropped. An important step in identifying the data pertaining to the phone call itself is to assign unique flow IDs – packets transmitted from Node 0 have a flow ID of 1, while packets transmitted from Node 4 have a flow ID of 2.

The NS-2 trace file is structured in columns as follows:

1. Event type
   - "+": packet enters queue
   - "-": packet leaves queue
   - "r": packet received
   - "d' packet dropped
2. Timestamp
3. Source Node
4. Destination Node
5. Packet Type (ie: "exp" for the variable bit rate used to simulate the call)
6. Packet Size
7. Flags (unused)
8. Flow ID
9. Source Address
10. Destination Address
11. Sequence Number
12. Unique Packet ID

A 60 second simulation with such a large amount of background traffic could result in trace files that are hundreds of MBs large, making parsing impossible. Since we are only interested in the data pertaining to the VoIP session between Nodes 0 and 4, the trace file is filtered using an AWK command:

```
awk '$8==1 || $8==2' out.tr > out.txt
```

This command takes lines containing flow ID 1 or 2 and transfers them to a new file.

3.2.2Throughput

In the case of data transmission from Node A to Node B, throughput refers to the total amount of data, measured in bytes/sec, received at Node B. As an example, for our simulation Node 0's throughput was Throughput = Bytes ReceivedΔ × 10
measured in kbps as follows:

$$_{Node\ 0} = \frac{\text{Bytes Received}_{Node\ 4}}{\Delta t \times 1\ 3}$$

The event type "r" in the first column of the trace was the primary tool used to track throughput.

3.2.3 Packet Loss

Packet loss is simply a measure of the amount of packets that are dropped at any point as they travel from Node A to Node B. Two measures were taken: one of instantaneous packet loss which describes how many packets are lost at each time interval and the other of cumulative loss which describes the total number of packets lost throughout the simulation. The main tool in tracking the loss of packets was the event type "d" in the first column of the trace file.

3.2.4 End-to-End Delay

End-to-end delay is the time it takes for a packet to travel from Node A to Node B. The end-to-end delay is measured as follows:

1. Obtain the time when a packet is created/transmitted
2. Obtain the time when the same packet reached its destination

3. Take the difference of the two times

To implement this procedure, first we find the time when the packet is created by checking for event type “+” at the node of creation. The unique packet ID specified in column 12 must be saved and a new search for the same packet ID with the event type “r” is performed, yielding the received time. The delay is then calculated as mentioned and the next packet with event “+” at the node of creation is found, thus repeating the process. The procedure is performed for all transmitted/received packets within a time interval and the resulting differences are averaged.

3.2.5 Jitter

Jitter, also more formally known as IP Packet Delay Variation (IPDV), is defined as the difference in end to-end delay of the transmitted packets. There are numerous ways of calculating this quantity.

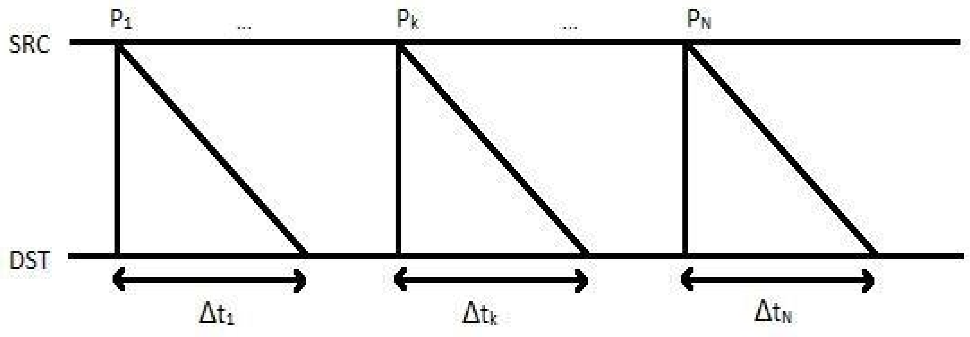

Figure 3.3. Illustration of packets with varying end-to-end delays

RFC 3393 suggests a simple method: An interval of time is selected where the first packet P_1 and last packet P_N transmitted during that interval have delay times of Δt_1 and Δt_N, respectively. The jitter is then given by $\Delta t_N - \Delta t_1$[3].