Performance Evaluation of a Wireless Network using a VoIP Traffic Generator on a Mobile Device.

Ghislaine L. Ngangom Tiemeni
Department of Computer Science
University of the Western Cape
Private Bag X17
Bellville, 7535 South Africa
+27 21 959 3010
3261404@myuwc.ac.za

Isabella M. Venter
Department of Computer Science
University of the Western Cape
Private Bag X17
Bellville, 7535 South Africa
+27 21 959 3010
iventer@uwc.ac.za

William D. Tucker
Department of Computer Science
University of the Western Cape
Private Bag X17
Belville, 7535 South Africa
+27 21 959 3010
btucker@uwc.ac.za

ABSTRACT
The problem of generating different patterns of traffic to emulate real user behaviour is receiving considerable attention with the construction of new and more complex network architectures. The theoretical modelling of waveforms or signals that flow through networks is valuable in a variety of scenarios including performance analysis and the design of communication systems. In the literature, many computer-based performance evaluation tools have been discussed. However, these tools lack the ability to run on affordable technologies such as mobile phones. The fundamental contribution of this work is the design of a traffic generating tool called MTGawn which is able to run on a mobile device. Design Science Research was the research methodology used for the design and deployment of a prototype of the proposed system. VoIP traffic was emulated using an implementation of well-known real time transport protocols such as RTP and cRTP, and parameterization was defined by using three codecs namely: G.711, G.723, and G.729. An evaluation was performed in a laboratory wireless network testbed and preliminary results were collected and analysed. The results of the experiments show that such a measuring instrument can be deployed on a mobile phone. More experiments are being done to ensure the accuracy of the data and also to compare the results with that of computer-based systems. Furthermore additional functionalities, similar to the functionality found on the computer-based open source tools, are being added to the mobile tool.

Categories and Subject Descriptors
B.8.2 **[Performance and Reliability]** Performance Analysis and Design Aids; C.4 **[Performance of Systems]** Measurement techniques, Performance attributes

General Terms
Measurement, Performance, Experimentation

Keywords
Traffic generator, VoIP emulation, Mobile application, performance evaluation.

SAICSIT2014, September 29 - October 01 2014, Centurion, South Africa
Copyright 2014 ACM 978-1-4503-3246-0/14/09…$15.00
http://dx.doi.org/10.1145/2664591.2664626

1. INTRODUCTION
Wireless packet-switched networks have grown exponentially since sending data through the Internet rather than through the Public Switched Telephone Network (PSTN) has become a better option in terms of cost for both users and service providers. This reduction of telephony cost over Internet protocol (IP) networks added to the dramatic increase of mobile phone usage and has created a huge demand for voice applications over IP networks [12]. Consequently, the load on IP networks has increased dramatically with the result that bandwidth has become a real problem. Lack of bandwidth prevents users from generating and sharing content and affects the quality of service (QoS) of real time applications such as voice over IP (VoIP), which are very sensitive to delay. A high packet loss ratio and network delay also has a negative impact on the quality of multimedia transmission. Network traffic generation is useful when performing the measurement of traffic load to improve throughput and to limit delay for services such as VoIP, in order to optimize end-user experience.

In the literature, researchers rely on both passive and active monitoring techniques for the evaluation of networks. While passive monitoring consists of packet capture and classification, the active approaches generate and inject test packets into the network or send packets across the network and extract performance metrics at the reception of the packets [3, 9, 11, 17].

The performance monitoring tools presented in the literature are all PC-based. This paper investigates the design of a framework capable of generating network traffic representative of a wide range of traffic conditions on a mobile device. The aim of this research is to make a mobile tool available to evaluate the performance of wireless networks in remote areas where the deployment of computers or dedicated traffic generators would be difficult and impractical. For this purpose, a mobile tool namely MTGawn (Mobile Traffic Generator For Analysis of a Wireless Network) is proposed to ease feasibility testing and monitoring in the field. The proposed application monitors mobile phone transmission statistics within any network interface and emulates real user behavior by generating VoIP traffic. Furthermore, the generated traffic is captured at the receiving end of the network for extracting performance metrics such as delay and jitter.

The rest of the paper is organized as follows. Section 2 gives some background information about VoIP. In Section 3, some of the prominent computer-based network analysis tools are reviewed; Section 4 describes the research methodology—Design Science Research (DSR)—and methods used to design a prototype for the mobile traffic generator with analysis capabilities. Section 5 and 6 describe the setup of the experiments and the results obtained while using the tool to evaluate the performance of a wireless test-bed mesh network

deployed in a laboratory. Finally, Section 7 draws conclusions and identifies future work.

2. VOICE OVER IP BACKGROUND

VoIP refers to real-time delivery of voice packets across networks using Internet Protocol. The conversion of an analogue waveform to a digital form is carried out by a codec. The voice is divided into data packets and transmitted over the network. The data is moved between endpoints using a media protocol called the Real-time Transport Protocol (RTP). VoIP can use either H.323 and SIP (Session Initial Protocol), or some other protocol, for voice calls. It can be transported across the network using one of the common transport protocols such as User Datagram Protocol (UDP) or Transmission Control Protocol (TCP). However, the UDP protocol is primarily used for voice transport so as to decrease overhead and increase speed and efficiency [22].

In recent years, VoIP technologies have emerged that has led to voice application becoming one of the hottest trends in telecommunications. However, as with many other technologies, VoIP introduced both opportunities and problems [22]. Although it offers lower cost, greater flexibility and more features than traditional telephony infrastructures [22], one of the typical concerns with VoIP is the delay and loss in packet delivery, which are two important concepts of voice QoS requirements.

2.1 Real-time Transport Protocol (RTP)

The payload of a voice packet is wrapped in successive layers of information in order to deliver it to its destination. These layers are: IP, UDP, and RTP. UDP adds 8 octets, and routes the data to the correct destination port. It is a connectionless protocol and does not provide any sequence information or guarantee of delivery. IP adds 20 octets, and is responsible for delivering the data to the destination host. It is connectionless and does not guarantee delivery or that packets will arrive in the same order they were sent. RTP adds an extra 12 octets to the payload. In total, the IP/UDP/RTP headers add a fixed 40 octets to the payload [16].

RTP is an application level protocol, which provides the transport of real-time data packets. RTP is flexible and provides the information required by a particular application and will often be integrated into the application processing rather than being implemented as a separate layer. A RTP packet format header contains useful information such as the payload type, the sequence number and timestamp.

The payload type specifies the format of the payload in the RTP packet. An RTP sender emits a single RTP payload type at any given time. An RTP packet can contain portions of either audio or video data streams. To differentiate between these streams, the sending application includes a payload type identifier within the RTP header. This identifier indicates the specific encoding scheme used to create the payload [14].

The sequence number is a value that is randomly initialized and incremented by one for each RTP data packet sent. This value is used at the receiver side to reorder packets and detect losses.

A timestamp is a 32 bytes random initial value used to represent the sampling instant (or creation time) of the first audio/video byte in each packet. This value does not represent the actual time of day when the packet was generated. It is incremented monotonically and linearly and the resolution of the timer depends on the desired synchronization accuracy required by the application [14].

A variant of RTP is compressed RTP (cRTP), which eliminates much of the overall packet's header. A cRTP packet has only 2-4 Bytes of IP/UDP/RTP header. Therefore, the network is more efficient and the user can place approximately twice as many calls as compared to a system running standard RTP [13].

2.2 Voice Codecs

VoIP relies on codecs (coder/decoder) to convert analog voice signals into digital data packets suitable for transmission over a digital network and reverses the process when the digitized voice reaches its destination. The primary goal of a voice codec and transmission process is to accurately reproduce the original speech. The codec ensures that the quality of the voice is as good as the quality of a call made over a traditional PSTN [15]. The codec determines the actual amount of bandwidth that the voice data will occupy. It also determines the rate at which the voice is sampled [16].

A codec is characterized by the number of bits produced per second and the sample period, which define how often samples are transmitted. These two parameters determine the size of the frame [16]. Larger frames allow for more efficient encoding but introduce larger delays and higher sensitivity to packet loss. Therefore the choice of a good frame size is equally important as the choice of codec [6]. The codec samples the waveform at regular intervals and generates a value for each sample. These samples are typically taken 8,000 times a second, or 8KHz [16]. These individual values are accumulated for a fixed period (sample period) to create a frame of data. A packet can contain one or many frames of data. For example, the G.729a codec works with a 10 ms sample period and produces a very small frame (10 bytes). It is more efficient to place two frames in each packet. This decreases the packet transmission overhead without increasing the latency excessively [16].

2.3 Voice Quality

Voice quality is affected by both the choice of the codec and compression methods together with QoS of the network such as packet transmission delay, jitter, and packet loss caused by network congestion [15].

The main issue of VoIP is a greater potential for degraded voice quality due to packet latency when the underlying network links experience heavy traffic load. When a VoIP media server is streaming voice traffic, audio data is periodically processed and sent out over the network. The user then receives packets at regular intervals. During the process, delayed packets may need to be dropped so as not to disrupt real-time playback [10]. Packets that contain voice data can be lost for several reasons including: insufficient bandwidth due to poor capacity planning; packets arriving at their destination too late; and network outages. Since voice quality is very sensitive to packet loss (even 1% packet loss can affect voice quality), such packet drops caused by delayed packet delivery, can result in degraded voice quality. Therefore, timely processing and delivery of audio data is critically important to VoIP services [10].

Voice-carrying packets should not excessively be dropped, delayed or be subjected to high variation in delay to ensure an intelligible audio reception [4]. Delay can have a considerable impact on conversational quality. Delay leads to conversational interaction problems. For example, it can lead to call participants interrupting each other (doubletalk) or to excessive pause in speech. Frequent interruptions can be annoying and excessive silence periods might be confused with delays in response, which can change the apparent emotional content of speech [20].

The variation in delay is called jitter, which also causes damage to voice quality. If the delay variation is too high, packets arrive too late to be of any use, and are discarded. A jitter buffer helps reduce the impact of this effect by buffering the packets for a short while before playing them back. The jitter buffer will also fix any out-of-order errors by looking at the sequence number in the RTP frames. This has the effect of smoothing the packet flow, increasing the resiliency of the codec to packet loss, delayed packets and other transmission effects. Even though this technique effectively reduces packet loss, the downside of the jitter buffer is that it can add a higher delay [1, 2]. Monitoring jitter and packet loss in a network can be achieved by monitoring the media stream by using Real-time Transport Control Protocol (RTCP).

3. LITERATURE REVIEW

Network analysis can be phrased as the process of capturing or monitoring network traffic and inspecting it closely to determine what is happening on the network [17]. For the network performance analysis, the purpose is to evaluate a statistic of a metric related to the performance of the system. In communication networks, it is very important for network administrators to be aware of and having to handle the different types of traffic that are traversing their networks. Traffic monitoring and analysis is essential in order to more effectively troubleshoot and resolve issues when they occur [5]**.** A performance monitoring system generate or capture representative packet or flow in real or emulated network environments and create representative workloads in a simulation environment. An active or passive monitoring approach is used to measure performance characteristics of the traffic.

Numerous network-monitoring tools apply passive techniques to analyze the performance of networks. Some of the prominent passive tools include Wireshark and Tcpdump. A passive approach uses devices or packet sniffers to watch and capture the traffic flowing on the network. Passive methods simply perform an analysis of the traffic that flows through the network, without changing it. They help to determine the characteristics of the traffic that flows through the measurement point, like the average rate, the mean packet size or the duration of the connections [21]. Passive monitoring compute traffic statistics that are helpful to identify the type of protocols involved the communication problems and the bandwidth usage [8].

Initially developed in 1997 by Gerald Combs, Wireshark [17], previously called Ethereal, has become one of the most popular tools for network monitoring and performance evaluation. Wireshark is a network protocol analyzer which can be used to read and capture files from a variety number of sources including others sniffers (such as Tcpdump), routers and network utilities. It works on Windows and UNIX. It uses the well-known library "libpcap"-based capture format but also has the ability to read captures in a variety of other formats. It currently supports over 750 protocols including VoIP. Wireshark has the ability to read packets and display the ASCII in an easy to read format. It provides a graphical user interface to browse the captured data, viewing summary and detail information for each packet. It implements a filter which helps to find a desired packet without sifting through all of them. Wireshark uses both capture and display filters. The capture filter allows capturing certain types of traffic and the display filter provides a powerful syntax to sort the captured traffic.

Another prominent passive monitoring tool is Tcpdump. Tcpdump [9] is a powerful Unix-based command-line packet analyzer. It also uses libpcap, a portable C/C++ library for network traffic capture. It has the ability to intercept and display packets being transmitted or received over a network. Tcpdump uses a program called Tcptrace to analyze network behavior, performance of applications that generate or receive network traffic. Tcptrace [19] is a tool written by Shawn Ostermann at Ohio University, for analysis of Tcpdump files**.** It has the ability to read and analyze a variety of other files generated by several popular packet-capture programs such as Windump (Windump is the Windows version of Tcpdump) and can produce several different types of output containing information on each connection seen, such as elapsed time, bytes and segments sent and received, retransmissions, round trip times, window advertisements, throughput, and more. It can also produce a number of graphs for further analysis.

Contrary to passive monitoring, which mainly consists of packet capturing and classification, the active approach injects test packets into the network or sends packets to servers and applications, following them and measuring services obtained from the network. The active approach provides explicit control of the generation of packets for measurement scenarios. This includes control of the nature of traffic generation, the sampling techniques, the timing, frequency, scheduling, packet sizes and types (to emulate various applications), statistical quality, the path and function chosen to be monitored [7].

Some of the commonly used performance evaluation tools that apply the active strategy include Iperf and D-ITG. Iperf [11] is a tool to measure the maximum TCP and UDP bandwidth performance and the quality of a network link. It creates TCP and UDP data streams and utilizes the client/server architecture to send a select amount of data from an Iperf client to a listening Iperf server, and measures the time that it takes to transmit/receive the data. Iperf allows the tuning of various parameters and UDP characteristics. Iperf reports bandwidth, delay jitter, datagram loss. Iperf enables an Iperf client to run multiple simultaneous connections to the server as well as both sending and receiving of data at the same time.

Avallone et al. developed D-ITG that allows generation of transport layer traffic (TCP and UDP) and other types of traffic including VoIP and Video. D-ITG [3] has additional functionality such as using different network loads or different network configurations to study scalability problems. It allows the generation of complex and various traffic sources, and offers the possibility to repeat many times exactly the same traffic pattern (not only its mean value) and get information about both received and transmitted packets. D-ITG enables the measurement of both the round trip time and one-way delay.

Both passive and active monitoring offers benefits but both have drawbacks as well. Active techniques create extra traffic on the network, and the traffic or its parameters are synthetic (the traffic is either simulated or emulated). However, the volume and other parameters of the introduced traffic are fully adjustable and small traffic volumes are enough to obtain significant measurements. Passive techniques do not have the overhead that active monitoring has. However, the amount of data gathered can be extensive especially if one is doing flow analysis or trying to capture information on all packets. With passive monitoring, measurements can only be analyzed off-line and not as they are collected. This creates another problem with processing the huge data sets that are collected [5]. Yet, the combination of both techniques can be applied in order to provide useful results.

4. DESIGN AND METHODS

This section briefly describes the methodology and strategy used during the research process to design a prototype for the proposed system as well as the architecture of the proposed

system. The methodology used was Design Science Research (DSR). We follow the approach proposed by Peffers et al. They suggest six phases (DSRM activities) in the DSR iterative process [18] (see Figure 1). The knowledgebase of theories and existing artifacts—collected through document analysis and literature surveys—feeds into the design of the prototype.

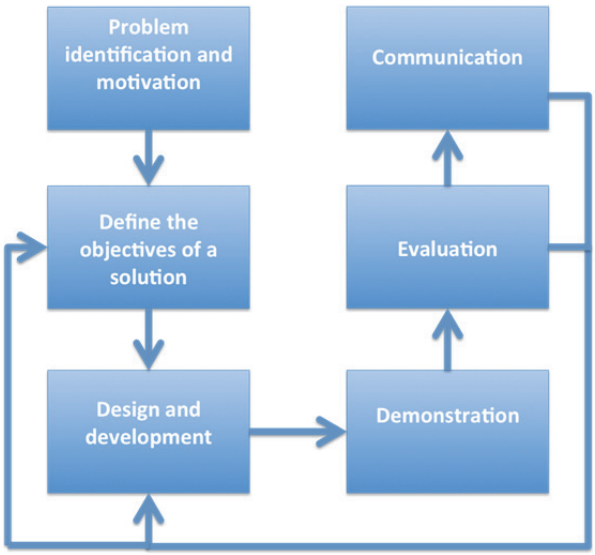

Figure 1: Design science research methodology (DSRM)
Source: adapted from Peffers, Tuunanen, Rothenberger and Chatterjee (2008)

The first phase, the problem identification was done through document analysis and a literature survey. We identified the problem as being the inability of existing performance analysis tools to run on a mobile device. The objective of this research was to design a new tool that will ease feasibility testing and monitoring in the field.

During the design and the development phase of the prototype, one of the main challenges was to find a way to design an easy to use interface to allow users to generate VoIP traffic using various configurations. For instance, the user is required to define the number of voice packets to send as well as the codecs used to emulate the voice (G.711, G.723, G.729) and the voice transport protocol (RTP or cRTP). Afterward, we had to create a model capable to represent relevant features of real-life VoIP traffic flows. To create a successful model, it is very important to classify the network activities because different user activities produce different traffic patterns and each traffic pattern can be characterized by various parameters. Two significant parameters were taken into consideration: the size of each transmitted packet and the elapsed time between packet transmissions (the rate at which packets are transmitted over the network). Those parameters vary depending on the codecs. The protocol used to transport real time data affects the size of packets as well since RTP adds a 12 bytes header to the payload while cRTP compressed RTP header adds to 2-4 bytes.

Having chosen a type of codec and a protocol for the transport of voice, the next step was to emulate VoIP traffic in an Android environment. The emulation used the implementation of the UDP transport protocol to emulate UDP packets and send them over a wireless network through an Android DatagramSocket. This class implements a UDP socket for sending and receiving DatagramPackets. A DatagramSocket object can be used for both endpoints of a connection for a packet delivery service.

The system was divided in four modules with different roles. In the first module, a UDP DatagramPacket is used to emulate voice traffic. RTP packets are created and encapsulated in the data area of a UDP packet and is subject to the same constraints as UDP. Then, the second module is in charge of generating and sending the RTP packet over the wireless network through the UDP socket. At the receiver's side, the flows are captured by another UDP socket and finally proceed by extracting properties such as packet loss, delay and delay jitter, which are of critical importance for interactive or streaming multimedia. Figure 2 depicts the architecture of the MTGawn tool.

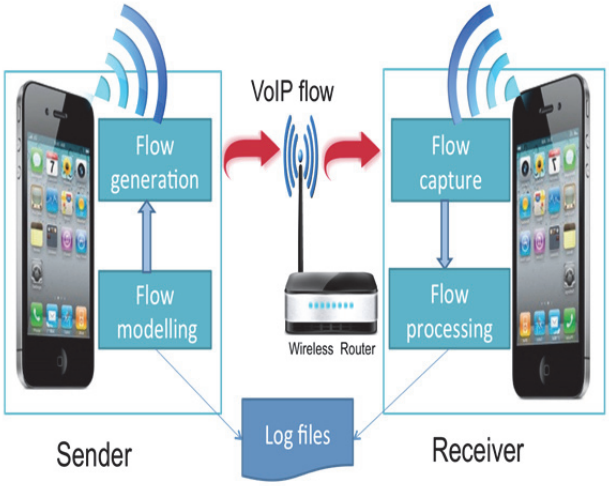

Figure 2: Architecture of the proposed system.

Once the design and development of a prototype was achieved throughout the previous phase, the demonstration phase in this case focused on running the prototype on a wireless laboratory testbed environment. This test was done to demonstrate that the prototype was indeed able to generate VoIP traffic on an Android device over a wireless interface. As a real time application is very sensitive to delay, the testing was to evaluate how well and how fast the voice was transmitted. Thus the following performance metrics were selected: delay and jitter as being the appropriate evaluation metrics.

The evaluation was carried out by analyzing the log files of both sender and receiver. Both files contained detailed information—such as the flow identifier, the packet identifier, the time sent, the time received and the payload size of the packet—for each packet sent and received during the generation process.

5. EXPERIMENTAL SETUP

The experiments for this paper were performed in a testbed environment deployed in a laboratory. We designed and configured a small wireless network, which consisted of a Mesh Potato device acting as wireless access point. We then connect two Android phones to the wireless network and configured one as a sender and the other as a receiver. The MTGawn tool was installed on both phones to send and receive VoIP traffic.

In the experiments, the two Android phones used were identical in term of hardware and software. Both phones were running an Android version 4.1.2 with 1GHz Dual core processor and 8GB internal memory. The Mesh Potato (MP) device used is an Atheros AR2317 system on a Chip (SoC) running Asterisk 1.4.11 Firmware with MIPS 4k processor 180 MHz and 16 MByte RAM. Figure 3 illustrates the network setup.

Figure 3: Wireless testbed network setup.

To simulate the VoIP flows, three types of codecs were taken into consideration: G.711, G.723 and G.729. As mentioned above, voice quality is affected by packet loss, delay and variation in delay (jitter). However, for this research performance metrics of interest were delay and jitter. These performance metrics were computed as follow:

1. **Delay (1):** If *Si* is time transmitted at the sender for packet *i*, *Ri* is the time of arrival at the receiver for packet *i*, then for two consecutive packets sent *i* and *j*, Delay D_i is to be expressed as:

   The variation in delay is defined as the difference D in packet spacing at the receiver R compared to the sender S for a pair of packets. For two packets (*i* and *j*), D is expressed as:

   $$D(i, j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$$

2. **Jitter (2):** Jitter is computed as the signed maximum difference in one-way delay of the packets over a particular time interval [12]. n represents the number total of packets transmitted.

   $$\textbf{Jitter} = \boldsymbol{max}_{i=1}^{n}[(R_i - R_{i-1}) - (S_i - S_{i-1})] = \boldsymbol{max}_{i=1}^{n}[\boldsymbol{D}(i, i-1)]$$

The experiments involved the sending of several flows for every codec type. Each flow contained 1000 packets of a codec equivalent to a VoIP call. Each codec has its own standard parameters as depicted in Table 1.

Table 1: Parameters for each codec [16].

Codecs	G.711	G.723	G.729
Sample period (ms)	20	30	20
Frame size (payload)	160	20/24	20
Rate (Packets/s)	50	33	50
Bandwidth (Kbps)	64	5.3/6.4	8

The higher the bit-rate (bandwidth) used for the codec, the better the voice quality. However, higher bit rate codecs take up more space on the network and also allow for fewer total calls on the network [13]. So it is required for a codec to use low bandwidth.

Long sample periods produce high latency, which can affect the perceived quality of the call. Long delays make interactive conversations difficult, with the two parties often talking over each other. Based on this fact, the shorter the sample period, the better the perceived quality of the call. However, the shorter the sample period, the smaller the frames and the more significant the packet headers become. For the smallest packets, the packet headers take up over half of the bandwidth used; which is not an advantageous case [16].

6. RESULTS

Experiments, as described in the previous section, were executed in order to evaluate the performance of a wireless network testbed by means of generating VoIP traffic on an Android device.

Two iterations of the DSR cycle were executed. In the first DSR cycle, the focus was on creating an easy to use interface on an Android device. This task was very challenging due to the screen size limitation together with the limited computing power of the Android mobile device. Figure 4 and Figure 5 present some screen shots of the user interface of the MTGawn traffic generator on an Android device.

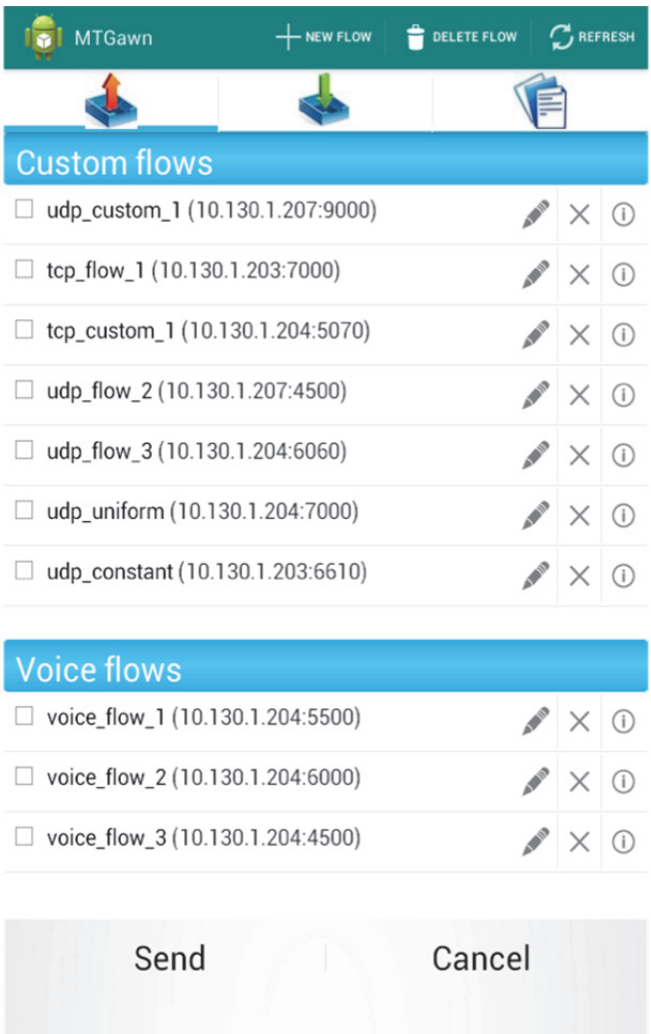

Figure 4: Flow sender interface

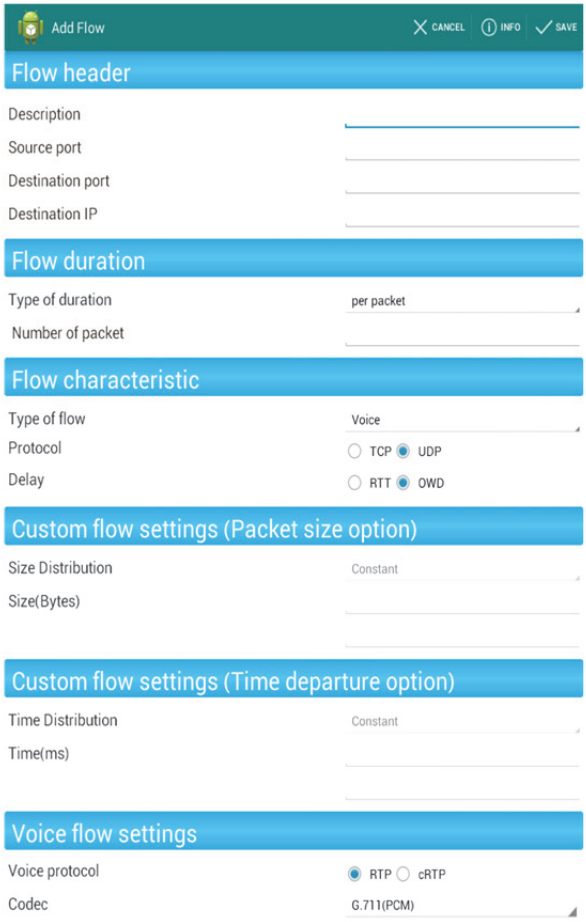

Figure 5: Flow modelling interface

During the second DSR cycle, voice traffic was emulated according to the parameterization defined by each codec and a RTP packet was created and encapsulated in a UDP packet to carry the voice. At the sender side, a total number of 10 flows (1000 packets per flow) were generated and transported over the network using UDP sockets to minimize delay. At the receiver side, the packets were captured and performances metrics such as delay and jitter were extracted. Figure 6, Figure 7 and Figure 8 illustrate the jitter variation in milliseconds (depicted on the y-axis) for 10 flows (depicted on the x-axis) obtained for each codec, respectively.

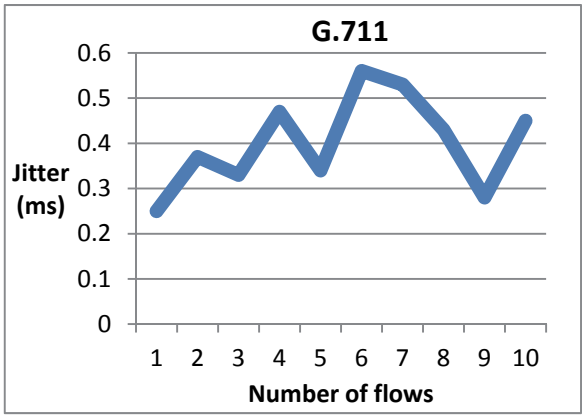

Figure 6: Jitter variation for G.711 codec

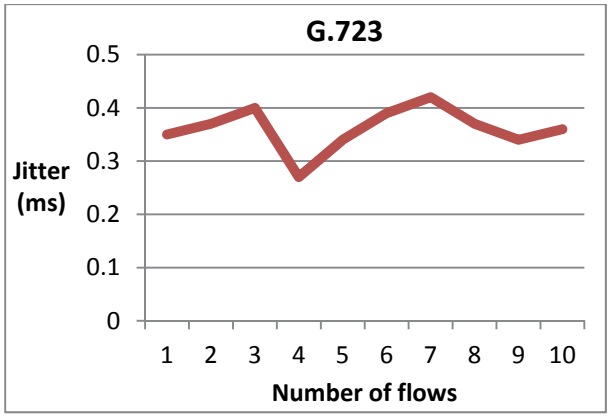

Figure 7: Jitter variation for G.723 codec

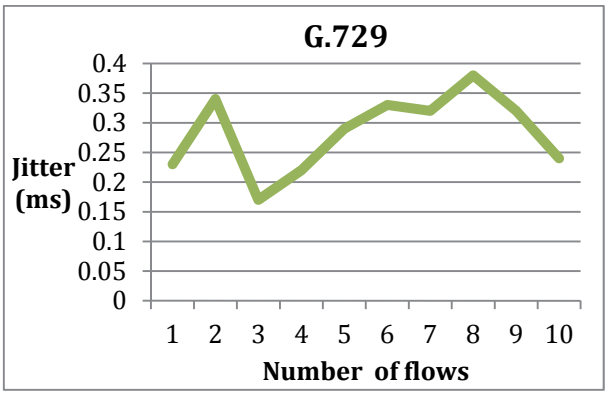

Figure 8: Jitter variation for G.729 codec

As can be seen from these figures that the MTGawn tool effectively shows the jitter variation for the three codes: for the G.711 codec, we observed that the jitter vary between 0,25 ms and 0,56 ms for each flow sent: and we observed a jitter variation from 0,27 ms to 0,42 ms for the G.723 codec and a jitter variation from 0,17 to 0,38 for the G.729. From these graphs it appears that the network is stable and the jitter is within a specific range for all three codecs.

7. CONCLUSION AND FUTURE WORK

In this paper, we proposed MTGawn, a tool for a mobile device to evaluate the performance of any wireless network in terms of delay and jitter by emulating and generating VoIP traffic. The tool can emulate both RTP and cRTP packets according to the parameterization defined by three different codecs (G.711, G.723 and G.729). The fundamental contribution of our work was the design of a traffic generator that is able to run on a mobile device. An evaluation was performed in a laboratory with a wireless network testbed and preliminary results were collected and analysed. This research reports on a limited number of experiments. Currently more experiments are being done to ensure the accuracy of the data and also to add additional functionality to the mobile tool similar to the functionality found in common open source tools.

Future work includes:

- Sending a large amount of flow and repeating the same experiments several times in order to achieve more accurate results by including more samples for each codec;
- Emulating voice traffic using other types of codecs;

- Collecting further performance metrics such as the round trip time (RTT), packet loss and throughput;
- Sending multiple flows simultaneously;
- Testing the tool on an actual rural wireless network; and
- Comparing the results against a standard PC-based performance tool such as D-ITG.

8. ACKNOWLEGEMENTS

We thank Carlos Rey-Moreno and Antoine Bagula for their input. We also thank Telkom, Cisco, Aria Technologies and THRIP (Technology and Human Resources for Industry Partnership) for their financial support via the Telkom Center of Excellence (CoE). This work is based on the research supported in part by the National Research Foundation of South Africa (Grant number (UID) 75191). Any opinion findings and conclusion or recommendations expressed in this material are those of the authors and therefore the NFR does not accept any liability in this regard.

9. REFERENCES

[1] Aktas, I., Schmidt, F., Weingärtner, E., Schnelke, C.-J., & Wehrle, K. (2012). An Adaptive Codec Switching Scheme for SIP-based VoIP. *Internet of Things, Smart Spaces, and Next Generation Networking*, 347-358.

[2] Alvarion. (2006). Implementing VoIP Service Over Wireless Network. Alvarion Technologies.

[3] Avallone, S., Pescape, A., & Ventre, G. (2003). Distributed Internet Traffic Generator (D-ITG): analysis and experimentation over heterogeneous networks. *International Conference on Network Protocols (ICNP 2003 Poster Proceedings).* Atlanta, Georgia.

[4] Barbosa, R., Kamienski, C., Mariz, D., Callado, A., Fernandes, S., & Sadok, D. (2007). Performance evaluation of P2P VoIP applications. *ACM NOSSDAV, 7*.

[5] Cecil, A. (2012). A Summary of Network Traffic Monitoring and Analysis Techniques. *Conference on Instruction & Technology (CIT)*, (pp. 10-25).

[6] Christiansen, T., Giotis, I., & Mathur, S. (n.d.). Performance Evaluation of VoIP in Different Settings.

[7] Cottrell, L. (2001, March 11). *Passive vs Active Monitoring*. Retrieved June 20, 2014, from http://www.slac.stanford.edu/comp/net/wan-mon/passive-vs-active.html

[8] Deri, L. (2004). Improving passive packet capture: Beyond device polling. *Fourth International System Administration and Network Engineering Conference (SANE 2004)*, (pp. 85-93). Amsterdam.

[9] Garcia, L. M. (2010). *TCPDump & LIBPCAP*. Retrieved 06 20, 2014, from http://www.tcpdump.org/

[10] Heo, J. (2011). Voice over IP (VoIP) Performance Evaluation on VMware vSphere®5 . Palo Alto CA: VMware.

[11] Iperf. (n.d.). Retrieved from http://iperf.sourceforge.net/

[12] Jadhav, S., Zhang, H., & Huang, Z. (2011). Performance Evaluation of Quality of VoIP in WiMAX and UMTS. *12th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, (pp. 375-380). Gwangju, Korea.

[13] JSDU. (2006). *VoIP overview.* JSDU uniphase corporation.

[14] Marsic, I. (2010). *computer networks:performance and quality of service.* New Jersey: Rutgers University (The State University of New Jersey).

[15] Mundra, S., & Hernandez, C. E. (2004). *Patent No. 20040032860.* USA.

[16] Newport Networks . (2005). *VoIP Bandwidth Calculation.* Newport Networks Ltd .

[17] Orebaugh, A., Ramirez, G., Burke, J., Morris, G., Pesce, L., & Wright, J. (2007). *Wireshark & Ethereal Network Protocol Analyzer Toolkit.* Canada: Syngress Publishing.

[18] Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2008). A design science research methodology for information systems research. *Journal of Management Information Systems , 24*(3), 45-78.

[19] tcptrace. (n.d.). Retrieved from http://www.tcptrace.org/

[20] Telchemy. (2006). VoIP Performance Managemen:Impact of Delay in Voice over IP Services. Telchemy.

[21] Veiga, H., Pinho, T., Oliveira, J. L., Valadas, R., Salvador, P., & Nogueira, A. (2005). Active traffic monitoring for heterogeneous environments. *Fourth International Conference on Networking (ICN 2005)* (pp. 603-610). Springer-verlag Berlin Heidelberg.

[22] Walsh, T. J., & Kuhn, R. D. (2005). Challenges in Securing Voice over IP. *IEEE SECURITY & PRIVACY*, 44-49.