INGENIERÍA Y DESARROLLO

ARTÍCULO DE INVESTIGACIÓN / RESEARCH ARTICLE
http://dx.doi.org/10.14482/inde.33.2.6368

Testing environment for video streaming support using open source tools

Entorno de pruebas para el soporte de videostreaming usando herramientas libres

Franco Arturo Urbano O.*
Fundación Universitaria de Popayán (Colombia)
Gabriel E. Chanchí G.**
Universidad del Cauca (Colombia)
Wilmar Yesid Campo M.***
Héctor Fabio Bermúdez O.****
Evelio Astaiza Hoyos*****
Universidad del Quindío (Colombia)

* Ingeniero en Electrónica y Telecomunicaciones. Magister en Ingeniería. Área Telemática, Universidad del Cauca. Profesor del programa de Ingeniería de Sistemas de la Fundación Universitaria de Popayán. frurbano5@gmail.com.

** Ingeniero en Electrónica y Telecomunicaciones. Magister en Ingeniería Telemática. PhD(c) Candidato a Doctor en Telemática, Universidad del Cauca. Profesor del programa de Ingeniería Informática de la Institución Universitaria Colegio Mayor del Cauca. gchanchi@unimayor.edu.co.

*** Ingeniero en Electrónica y Telecomunicaciones. Magister en Ingeniería. Área Telemática. Doctor en Ingeniería Telemática, Universidad del Cauca. Profesor asistente de la Universidad del Quindío. Investigador grupo GITUQ. wycampo@uniquindio.edu.co.

**** Ingeniero en Electrónica y Telecomunicaciones. Magister en Electrónica y Telecomunicaciones. Profesor asociado Universidad del Quindío. Investigador grupo GITUQ. hfbermudez@uniquindio.edu.co.

***** Ingeniero en Electrónica y Telecomunicaciones. Magister en Ingeniería. PhD(c) Ciencias de la Electrónica. Profesor asociado Universidad del Quindío. Investigador grupo GITUQ. eastaiza@uniquindio.edu.co.

Correspondencia: Héctor Fabio Bermúdez O. Universidad del Quindío. Programa de Ingeniería electrónica. Tel 57 (6) 7359353 Ext 108, Carrera 15 Calle 12 Norte. Armenia Colombia.

Volumen 34, n.º 2
Julio-diciembre, 2016
ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

Abstract

Among the different technologies with important implications today in such areas as education, health and business, videostreaming is highlighted. This considering how this technology facilitates the access to multimedia content remotely, live or offline. The goal of this paper is to propose a test environment for the support of the video streaming service, using open source tools. Moreover, this work proposes, as part of the environment, a stress measurement tool (Hermes), which allows obtaining the response times to establish multiple RTSP connections to streaming servers. The methodology used in this work is divided into four phases: analysis of technologies and tools, configuration of the video streaming environment, design and implementation of Hermes, and finally tests. This methodology allowed the construction of the test environment and its evaluation, through the stress measurement tool Hermes. Finally, in this work we demonstrate how the proposed environment becomes a reference point for different application environments that require the implementation of a video streaming service.

Palabras clave: Hermes, open source tools, RTSP, test environment, video streaming.

Fecha de recepción: 11 de septiembre de 2014
Fecha de aceptación: 1 de junio de 2016

Resumen

Dentro de las tecnologías que hoy en día tienen implicaciones importantes en ámbitos como la educación, la salud y el sector productivo, se destaca el videostreaming. Esto teniendo en cuenta las ventajas que esta tecnología ofrece para el acceso a contenidos multimedia de manera remota, en vivo o fuera de línea. El objetivo de éste artículo es proponer un entorno de pruebas para el soporte del servicio de videostreaming haciendo uso de herramientas libres. Así mismo, este trabajo propone como parte de éste entorno, una herramienta para la medición de estrés llamada Hermes, la cual permite obtener los tiempos de respuesta producto de las múltiples conexiones RTSP a servidores de streaming. La metodología usada para el desarrollo de este trabajo está dividida en 4 fases: análisis de tecnologías y herramientas, configuración del entorno de videostreaming, diseño e implementación de Hermes y pruebas. Está metodología permitió la construcción del entorno de pruebas y su evaluación, a través de la herramienta de medición de estrés Hermes. Finalmente, mediante este trabajo se demuestra como el entorno de pruebas presentado, se convierte en un punto de referencia para diversos entornos de aplicación que requieran el montaje e implementación del servicio de videostreaming.

Keywords: entorno de pruebas, Hermes, herramientas libres, RTSP, videostreaming.

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

INTRODUCCIÓN

Internet has allowed file downloading since the early stages of its development. Initially it was intended to be a network to share information among distant people geographically located and to access files that would not have been available to consult before its existence; giving special emphasis on the access to the information, even if you had to wait a long time while the file was being downloaded. Nevertheless the Internet evolved, making access to audio and video with file sizes measured in Megabytes, with good quality and acceptable downloading times. However, until recently, the technology supporting the Internet for downloading audio and video required the client machine to completely download the files before the user could see and hear the content, which brought drawbacks such as: rather long transferring times and the difficulty of real-time visualization. This situation is further complicated when considering that until recently, in developing countries, the bandwidth of most networks was measured in kilobytes.

In this context is where a technology called streaming came about, which continuously requests video data (video streaming) or audio sent to the server and in response it sends streams of data, which are not all completely downloaded, in order to hear the sound or see the images on the client-side, instead, the video can be watched and the sound can be heard as the streams that make up the requested file arrive. This type of network workload significantly improves waiting times and allows manipulating multimedia files, live or recorded [1].

Among the advantages of the video streaming are its low requirements, since a conventional server and a connection of at least 512Kb is sufficient. In terms of the clients' firewalls, these will not cause any problems for transmission. Moreover, video streaming is not only intended to be used by a client to receive a media file, it is also an ideal tool to be used in education, business or management fields as it allows transmitting or retransmitting conferences, lectures, events, programs, seminars, tele-education, interviews, and more. Social networks on the internet are so far the more widespread social phenomenon because of the speed with which its users have multiplied in a very short period of time, and part of that success is due to video streaming [2].

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

Thus, the purpose of this research is to provide the scientific community with a reference environment for video streaming support using open source tools, supported by open source tools. Besides, we present a tool for measuring stress for video streaming servers called Hermes. Similarly the scientific pertinence of this research is to improve the quality of services offered by organizations that wish to bring these environments into practice, such as higher education institutions. To achieve the above, this research started from the following hypothesis: it is possible to build video streaming environments for different operating systems and different devices, using open source tools for it.

At a technical level, there are standardized protocols designed for communication between clients and streaming servers. The first streaming protocols were developed by multinational companies such as Microsoft, Real and Apple, which saw the enormous potential in providing real-time video. The following are two protocols that are commonly used: RTSP (Real Time Streaming Protocol) y RTMP (Real Time Messaging Protocol).

RTSP is a no connection-orientated protocol to stream real-time data which defines how the information is sent between the client and the server [3]. RTSP allows controlling the sending of multimedia content, either previously stored or live. This protocol works at an application level and ensures that the data is delivered successfully. RTSP defines different connection types and different sets of requirements, to try and ensure that the data is sent over IP networks as efficiently as possible. The RTSP protocol is independent from the transport protocol and it may operate on UDP or TCP. Nevertheless, in the majority of cases, the TCP protocol is used for controlling the player and the UDP protocol for RTP data transmission [4]. In a session, a client can establish or close reliable transport connections with the server using RTSP requests [5], [6].

RTMP is a protocol developed by Macromedia, now Adobe, for high performance transmissions of audio and video between Adobe Flash Platforms, including Adobe Flash Player (AFP) and Adobe AIR [7]. The protocol has been released as an open specification to develop products and technology that permit the delivery of audio, video and data in SWF, F4V and FLV, formats compatible with AFP. RTMP uses TCP at the transport layer and supports multimedia streaming encoded in FLV (Flash Video),

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

a format offered by the FMS (Flash Media Server Adobe System). RTMP has three variations: simple RTMP, which operates on TCP and uses port 1935, RTMPT (RTMP tunneled), which is encapsulated in HTTP requests to overpass firewalls and RTMPS (RTMP Secure); functioning as RTMP but with a secure HTTPS connection [8]. This protocol can be configured to work with the UDP protocol. Currently, streaming servers that implement the RTMP protocol are FMS, Wowza Media Server, Onlinelib VCS Video Communication Server, WebORB Integration (available to .NET Java and ColdFusion) and Red5 [9].

Based on the basic concepts and streaming protocols presented above, a set of free tools and technologies in a testing environment are provided in this work, which allows the transmission and reception of multimedia content to different types of costumers (computer, phone, tablet) and on multiple operating systems, using the RTSP streaming protocol. In the election of this protocol, it was taken into account the need for open source tools for studying and evaluating it in an academic environment. Thus, the test environment was configured considering the most appropriate and most widely used open source tools.

Experiments of this research were developed in the "Fundación Universitaria de Popayán", seeking to support the processes of distance learning offered by the university, as well as broadening the range of educational resources to students. In order to evaluate the proposed transmission environment, stress tests on the RTSP streaming server were performed under extreme conditions, in order to estimate its robustness and reliability. In the case of web servers, there are several tools that simulate sequential and simultaneous HTTP requests, being apache benchmark [10] one of the most prominent options. However, in regards to servers based on the RTSP streaming protocol, it has not been evidenced the existence of a specific tool (and therefore neither related work) that can assess the stress on a streaming server, given the proper RTSP protocol format, with respect to the format of the HTTP requests. From the above, in the present work, a tool for measuring stress called Hermes was developed, as part of the testing environment, which simulates the establishment of multiple simultaneous connections to the hosted media content on a streaming RTSP server. The Hermes tool was developed in Python programming language and it makes use internally

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

of invocations to the streaming clients Open RTSP and VLC that run in the background.

This work is intended as a reference for projects requiring the implementation of video streaming based on different application environments, such as in the health sector and the education services sector. This article is organized as follows: section 2 describes the methodology considered in the development of this work; section 3 presents the technologies and concepts used in the formulation of the testing environment; section 4 shows each of the modules that make up the testing environment; section 5 includes the memory consumption and connection establishment tests made on the server-side with the Hermes tool; finally section 5 presents the conclusions derived from this work, as well as possible future work based on this study.

METHODOLOGY

With the objective of performing the testing environment for video streaming support, we developed this research in four phases: analysis of technologies and tools, configuration of the video streaming environment, design and implementation of Hermes, and finally tests (see Figure 1) [6].

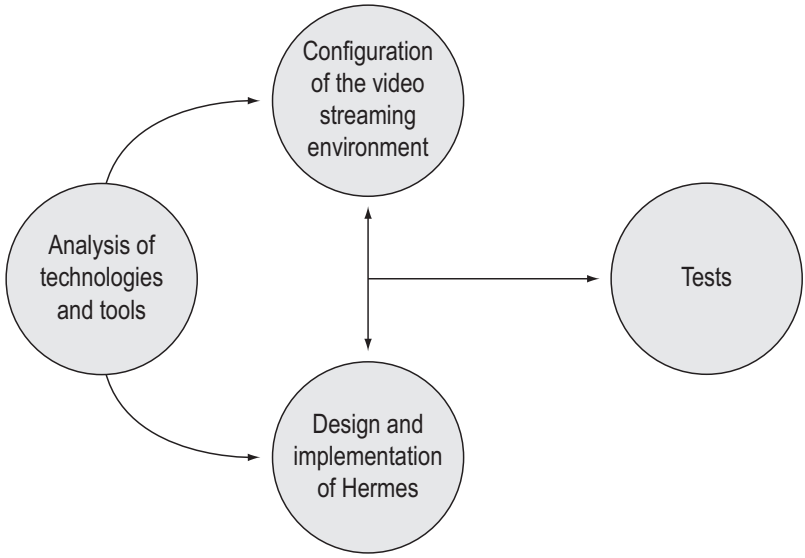


Figure 1. Methodology

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

In the first phase, we chose a set of open source tools for video streaming transmission, considering the processes of encoding, diffusion and reception of video. In the second phase, we configured an end-to-end video streaming scenario, taking into account the open source tools chosen in the first phase. In the third phase, we built a stress measurement tool called Hermes, using the characteristics of the RTSP server configured in the phase two. It is important to emphasize that the second and third phases were developed simultaneously, in order to allow the iterative building of the Hermes tool. Finally, in the fourth phase, we performed the memory consumption and connection establishment time tests from streaming server, using Hermes. This paper addressed the methodology phases as follows: section 3 comprehends the phase of analysis of technologies and tools, section 4 includes the phase of configuration of the video streaming environment and the phase of design and implementation of Hermes, finally section 6 covers the phase of tests [6].

TERMINOLOGY

A video can be understood as a series of images that are displayed one after the other, which gives the sense of movement. In turn, an image can be represented as a matrix of dots (pixels), each with its corresponding color. The properties of videos are described in [8]. Another key concept is that of a Codec, which is an algorithm that is both the compression and encryption of video, its final purpose is to compress, encode and encrypt information in order for it to be stored in a disk or transmitted over the network. In order to implement client applications, a codec can be found in an external library, and the application simply calls it when it detects that a video is in that particular format [11].

Among the most used video formats we have the following: MPEG-1 first set of standards and video compression formats and audio designed by MPEG (Moving Picture Experts Group). MPEG-2 is a higher quality version of MPEG, which can encode with interlacing. MPEG-4 is an ISO/IEC standard, improved version of the MPEG-2 codec and supports more audio and video than their predecessors [11], [12]. H.264 is technically identical to the MPEG-4 codec part 10, and its goal is to get better quality at a lower bitrate than older formats [13], and 3GP is a reduced version of MPEG-4 Part 14 container, designed to minimize storage and bandwidth requirements [14].

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

The basic requirements in choosing a tool for a video streaming system using open source technologies are: the use of the free and open software, added value services (playlists, rate adaptation, multicast), development flexibility (more alternatives for project implementation), and lower response times for operations performed by the system user (play, stop, forward). For the transmission environment configuration, the following servers, clients and codification tools were explored:

- Darwin Streaming Server: DSS, this server can transmit video over the internet, LAN and WLAN, using RTP, RTSP and SDP (Session Description Protocol) protocols. It is an open version of the Quicktime Streaming Server and is able to work as an MP3, MP4 and 3GP hinted file server. It does not support real time rate adaptation. Its main disadvantage is its unreliable statistics (CPU, throughput) [15].

- Catastreaming (Open Streaming Server), which is an Open Source project released under General Public License (GNU) whose main features are: an open source server, its support of protocols: RTP, RTSP, SDP, it does not support rate adaptation in real time, no GUI, but it allows connection to http servers and XML pages. In addition, there should be a http request for each query, there is no technical support, and the statistics are unreliable [16].

- Helix DNA Server is an open version of the Helix Universal Server available on its project webpage. It is distributed under the General Public License (GNU). Its main features: open source server and RTP, RTSP, SDP support. This server does not support 3gp files [17].

- Helix Universal Server is the commercial version of Helix DNA, which allows rate adaptation on the server. It also provides a complete software package that can be used, among other things, to encode the material. It has not been considered since it is not an open source software [17].

- Quicktime Streaming Server (QTSS) is the commercial version of Darwin Streaming Server. It contemplates all the characteristics of this type of server features, but it is also not a free version. In

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

fact, it is only distributed with a server version of the Mac OS X operating system [18].

LIVE555 is a streaming media server supported on the streaming protocols: RTSP, RTP and SDP. This server is an open application whose source code is available and can be modified for specific requirements. LIVE555 is compatible with media players like VLC and QuickTime. This server can generate different types of streaming media files, such as: MPEG TransportStream (.ts), WebM or Matroska (mkv or .webm), MPEG-1, MPEG-2 (mpeg), MPEG-4 (.m4e), H.264 (.264), DV (.dv), WAV (wav), MP3 (mp3), AMR (.amr), AAC (aac) .These streams can be received and / or reproduced by any RTSP / RTP media client, some of which include: VLC Media Player, OpenRTSP, QuickTime Player and Amino Set-Top Boxes [6], [19]. Considering that the state of the art that was analyzed, this server meets the specified requirements.

- VLC: It is an open and cross-platform (Windows, Linux, Mac OS X, Solaris, BeOS, etc.) media player. It can reproduce MPEG-1, MPEG-2 and MPEG-4 / DivX files from a CD-ROM, DVD, VCD hard drive or from a satellite card (DVB-S). VLC supports unicast or multicast transmission with IPV4 or IPV6. It allows the reproduction and distribution of content on demand using streaming protocols (RTP / RTCP, RTSP) and 3GP files. If a client makes a request like rtsp://ip:port/resource, then they can access the specified file. Although VLC is a good tool for multicast transmission and transcoding, it does not seem to be suitable to develop a streaming video on demand service. Although it is not an interesting option as a streaming server, it can be used as a transcoder, or even as an alternate server [6], [20]. It is being used in this article as a streaming client.

  To decode a stream, VLC first demultiplexes it. This implies that it processes the container format and separates audio, video, and subtitles. Thus, each of these streams is sent to their decoders, which do the mathematical processing and decompress it. Furthermore, VLC can work in silent mode, which means it can be run from command line from any of the supported operating systems and

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

can consume streaming without using GUI. It is also capable to filter audio or video, if so desired [6].

- OpenRTSP: This is a command line software used to make the complete request, but not to visualize the media content with all its features. This software can be used to establish connections, transmit, receive and record RTSP streaming. OpenRTSP retrieves the session description (SDP), which enables to control over each sub-session audio or video. The data received from each sub-session are written in different output files extracted from the payload of the RTP protocol [6], [14], [21].

  Both VLC and OpenRTSP can work in silent mode and were used in this research as video streaming service clients. In order for these tools to be launched in the background, they are invoked from the Python application (Hermes) proposed in this paper.

- Finally, as a coding tool, we explored FFmpeg, which is an open source multimedia system to decode, encode, transcode, multiplex, demultiplex, transmit, filter and play audio and video files. The FFmpeg project aims to provide the best possible technical solution for application developers and end users [21].

In this work LIVE555, VLC, OpenRTSP and FFmpeg were chosen, which have a certain degree of maturity and development in working with multimedia content, specifically as it relates to encoding, transmission and reception in multiple platforms. The Hermes stress assessment tool uses VLC OpenRTSP applications, which run in the background. Therefore, by using multithreaded programming from the Python language, it was possible to simulate simultaneous connections from multiple clients to the live555 streaming server.

TESTING ENVIRONMENT

In this section we present a modular diagram and the diagram for end to end implementation of all the different components of the testing environment, in which we highlight three key modules: the streaming RTSP server, the client for consuming multimedia content, which uses this protocol, and

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

the tool for measuring stress, which simulates multiple clients connecting to the server. Communication between the client, the stress measurement tool and the server is performed across a wireless network, taking into account the characteristics of mobility of the clients that were evaluated, see Figures 2 and 3.

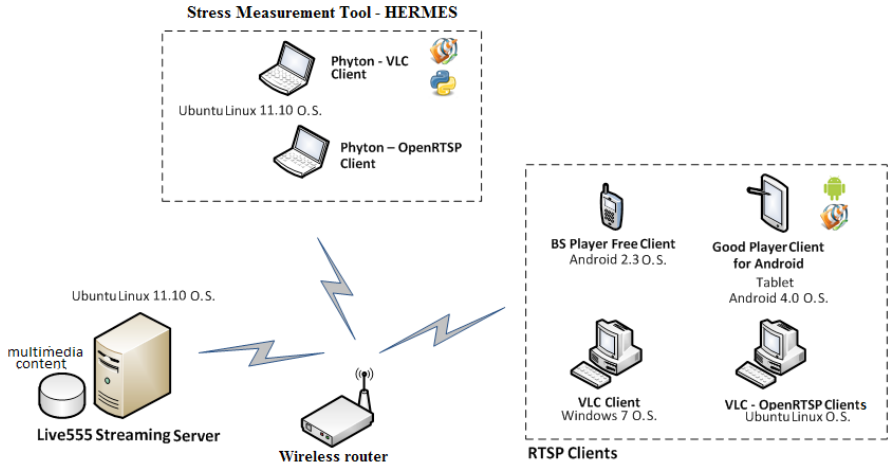


Figure 2. Testing Environment

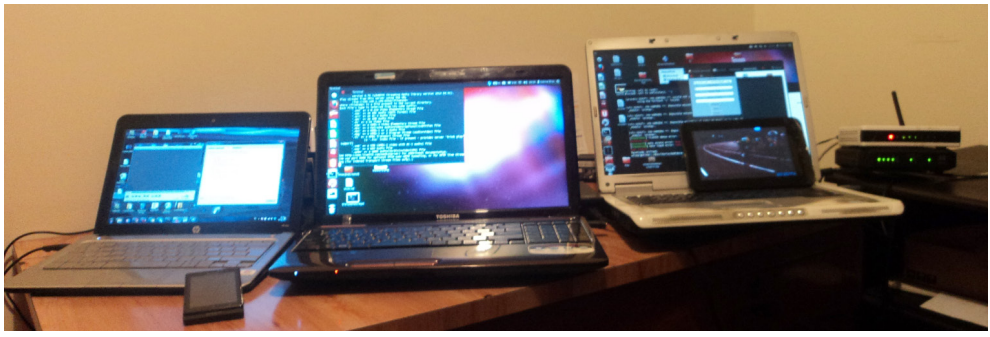


Figure 3. Experimentation Scenario

Server Module

This module consists of live555 streaming server and multimedia content in mpg format, see Figure 4. Live555 listens for RTSP requests on port

8554 and supports the following media containers: .264, .aac, .ac3, .amr, .dv, .m4e, mkv, .mp3, mpg, .ts, vob, wav, .webm. The media contents used were coded in MPEG-1 codec (mpgv for video and mpga for audio) using the open source encoding tool FFmpeg. This module was deployed on an AMD Quad Core laptop with the Ubuntu 11.10 operating system.

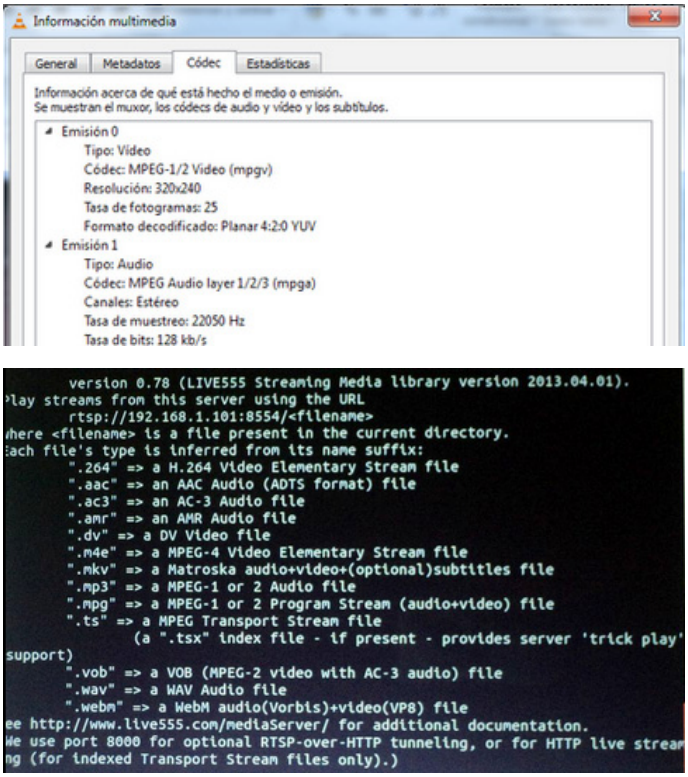


Figure 4. Streaming Server and Diffused Content

Client Module

This module consists of three different mobile clients: a tablet with the Android 4.0 operating system, a phone with the Android 2.3 operating system, and a laptop with the Ubuntu 11.10 operating system. On the tablet, the free media player used was Good Player For Android [22], which supports the mpg container and the RTSP streaming protocol; on the cell phone the free media player BS Player Free [23] was used, which supports the mpg container and RTSP; and finally, on the laptop we made use of two open

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

source tools that are widely used in the media field: VLC and OpenRTSP. The basic difference between these 2 clients is that OpenRTSP is a software module that has no graphical interface and has been developed to evaluate the connection and consumption of multimedia content through RTSP, see Figure 5.

Figure 5. Streaming Clients

Stress Assessment Module

This module consists of two streaming clients deployed on two laptops with Ubuntu 11.10. Each client runs the Hermes tool in order to measure the stress upon establishing a connection, using in each case a different tool in the background to connect to the LIVE555 server (VLC and OpenRTSP). For measurement purposes, each client is executed independently. In other words, first measurements are taken with the VLC client running in the

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

background and second measurements are taken with the OpenRTSP client running in the background, see Figure 6.

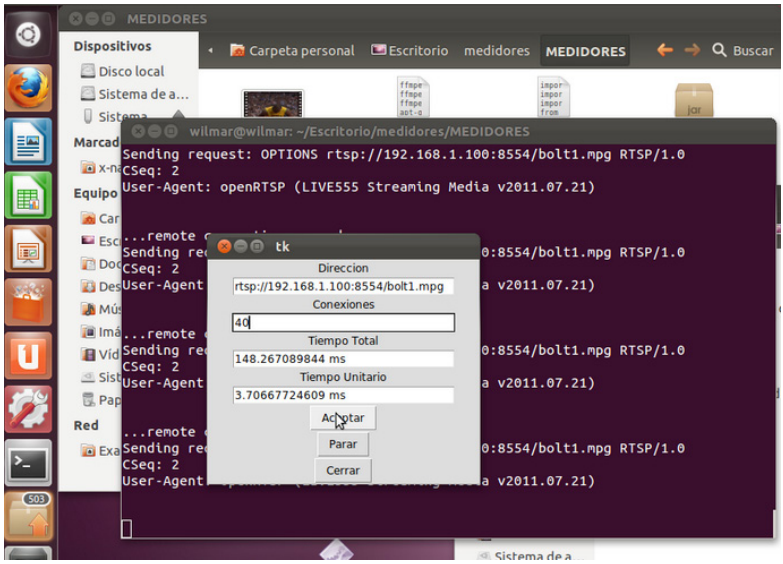


Figure 6. Stress measurement tool, Hermes

The Hermes stress measurement tool consists of the functional modules presented in Figure 7. In this diagram, the following main functional blocks are distinguished: Graphical User Interface (GUI), Pitcher threads, executing commands and Timer.

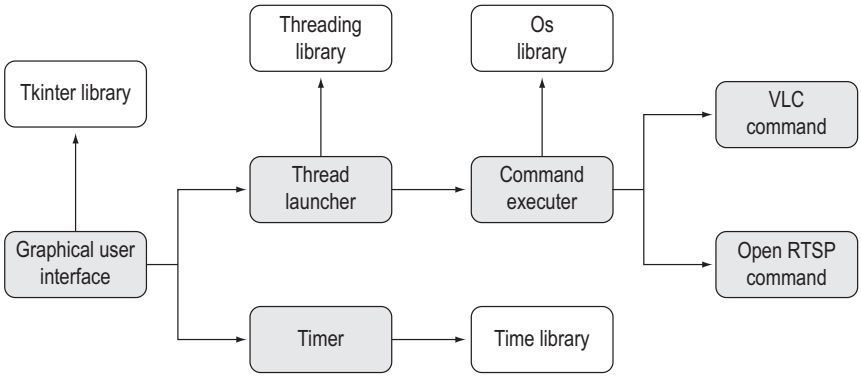


Figure 7. Modular Diagram for the Stress Measurement Tool Hermes

Ingeniería y Desarrollo. Universidad del Norte. Vol. 34 n.° 2: 333-353, 2016
ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

The GUI module uses the Tkinter library, which enables to create a set of GUI components (buttons and text fields), through which are prompted the address for the RTSP server to be evaluated and the number of simultaneous connections to be established. This interface also has two text fields in which the total time to establish the simultaneous RTSP connections and the average time of each connection are displayed. Meanwhile, the Thread Launcher module creates an instance of a thread for every simultaneous connection requested from the GUI, the above using the Python threading library. Each started thread established an RTSP connection with the streaming server LIVE555, so that for every n launches, n threads are generated in parallel with the connections to the server.

The Command Executer module is invoked when a thread is started and has the function of executing a command from the operating system (Linux or Windows), to establish connections with a RTSP Server. The executed command can be of two types: OpenRTSP Command or VLC Command. The OpenRTSP command launches the OpenRTSP on the operating system's console, in order to establish the RTSP connection with the streaming server, while the VLC command launches the VLC client on the OS console in silent mode (without opening GUI), in order to establish a connection with the RTSP server. In order to run the operating system command, the Python OS library is used, which allows to interact directly with the system console from within the programming language.

Simultaneously, when the threads are released and until the connection to the streaming server is established, the Timer module begins counting in milliseconds using the Python Time library. When the connection establishment time for n simultaneous clients, it is possible to stop the process of connecting using the OS library and the "killall" command on the Linux operating system.

STRESS TESTS AND USE OF MEMORY

Figure 8 shows the results of the connection establishment time tests performed on the live555 server, and the ones obtained by applying a different number of concurrent connections using the Hermes tool. This tool can establish two types of RTSP connections: using the VLC tool or OpenRTSP tool, both of these cases running in silent mode.

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

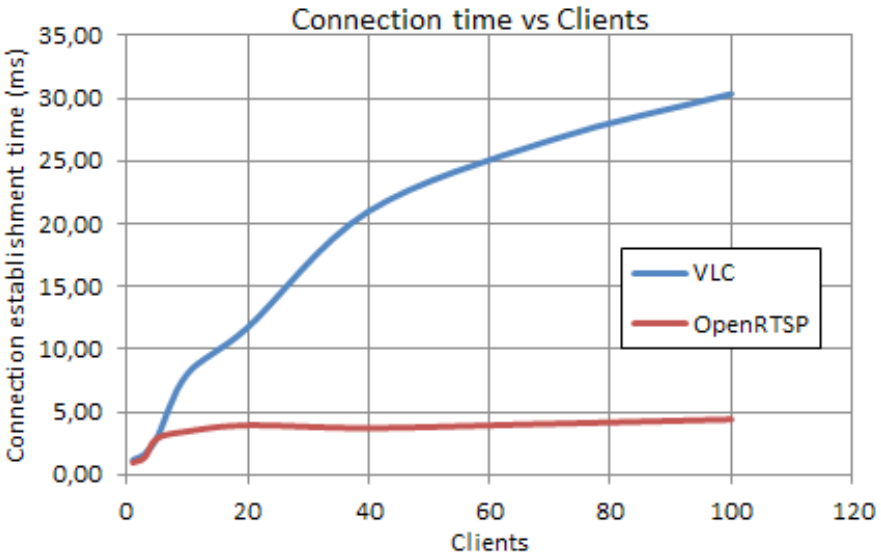


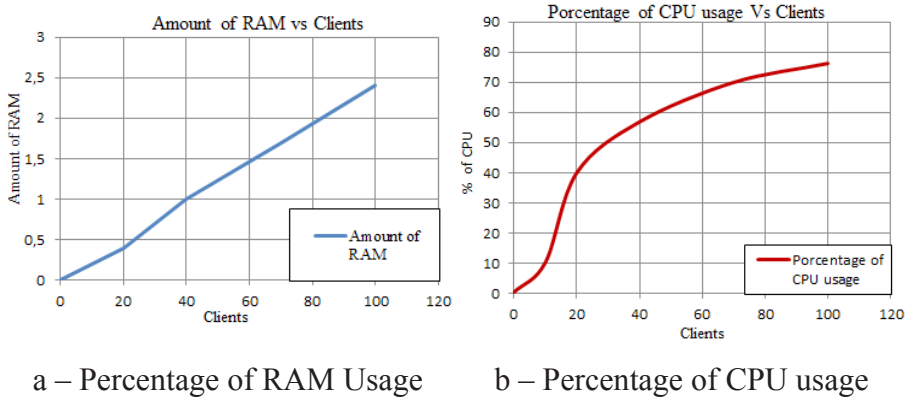
Figure 8. Connection Times vs Clients

According to Figure 9, when RTSP connections are established with the Hermes tool (running VLC client in the background), the ratio of the number of clients and the connection establishment time is directly proportional, reaching a value of 1500 milliseconds from 0 to 60 clients, and 3000 milliseconds from 60 to 100 clients.

However, when establishing RTSP connections using the Hermes tool (running OpenRTSP client in the background), the ratio of the number of clients and the connection establishment time grows slower than when it is done with VLC client. According to this graph, the connection time reaches values of 250 milliseconds between 0 and 60 clients and 500 milliseconds between 60 and 100 clients.

Furthermore, in Figure 9, the results of CPU usage rate and RAM (random-access memory) consumption tests performed on the LIVE555 server are presented. These tests were made by sending to the streaming server multiple and simultaneous connections with the Hermes tool, which allows different number of instances of the open source tools to run in the background: VLC and OpenRTSP.

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)



a – Percentage of RAM Usage b – Percentage of CPU usage

Figure 9. Test of memory usage

From the simultaneous connections generated with the Hermes tool, data memory consumption and CPU usage percentage are obtained with the "ps aux" command in Linux, From the simultaneous connections generated with the Hermes tool, data memory consumption and CPU usage percentage are obtained with the "ps aux" command in Linux, which provides a report on the amount of RAM and the percentage CPU, which is being used by each of the active processes of the operating system. This information was filtered by the awk programming language (included in the Linux operating system), giving specific live555 process consumption data.

In regard to Figure 9.a, the ratio between the number of clients (simultaneous) and the percentage of RAM used is directly proportional, varying about 0.5% each time the number of clients connected to the sever increases by 20. Meanwhile, in Figure 9.b, the ratio between of the number of clients (simultaneous) and CPU percentage usage is shown. As is shown in this figure, the ratio is directly proportional, having a greater variation of growth up to 40 clients (60% increase) and a more stable variation for 40 to 100 clients (20% increase).

CONCLUSIONS AND FUTURE WORK

The testing environment presented in this work integrates the most appropriate open source software in the world used to implement video streaming services based on the RTSP protocol.

The video streaming server LIVE555, proved to be a suitable software for the transmission of video information for different devices with different operating systems. Their mode of operation is based on the transport protocol RTSP, through which video content is transported over the network. While using the FFmpeg open source software, it is possible to edit a video in order to meet the transportation requirements desired, such as formats and codecs. Furthermore, FFmpeg video files can be packaged in transport streams, so that they can be played on different devices, such as tablets, phones or computers.

The correct operation of the test environment with different operating systems and different devices was demonstrated, using a single video streaming server, allowing its extension to other scenarios. Thus, the proposed framework seeks to provide guidance for projects requiring the implementation of video streaming-based application environments with different services, such as health and education sector.

From the connection establishment testing, it can be concluded that the times obtained for simultaneous connections below 100 clients are acceptable and allow proper reception of media content. The difference in behavior between OpenRTSP and VLC is that the first is merely a tool to run from console (silent mode), so it responds best to the RTSP connections.

According to the memory consumption and CPU usage tests, and considering the trend set by 8a and 8b graphs, the percentage of CPU usage caused by connections from over 100 clients to the live555 server can create processing problems in the transmission of the media content, while causing difficulties in the correct consumption of such by the clients.

The Hermes stress measurement tool, facilitates obtaining simultaneous connection times for multiple clients on a RTSP streaming server. Moreover, Hermes allows for secondary memory consumption measurements on the server-side, facilitating the evaluation of the server's performance when faced with multiple and simultaneous RTSP connections.

The use of Python in the construction of the Hermes tool facilitates the invocation of operating system commands (VLC and OpenRTSP), through which it is possible to connect to the RTSP streaming server. Likewise,

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

the multithreaded support characteristics by the Python language allow launching multiple RTSP connections, to evaluate the performance of the streaming server.

Future work could consider, in the test environment, modules to control the quality of multimedia transmission, according to the characteristics of the network. It could also aim to develop traffic studies, which can help plan a network's transmission capacity, telematic video features and protocols that support them. Additionally, future work could extend the operation of the Hermes stress measurement tool, so that it can generate graphics automatically, with response times provided by the server.

ACKNOWLEDGEMENTS

This work has been carried out under the project "Testbed for video streaming service support of educational content at FUP", developed at the Fundación Universitaria de Popayán. This work has been supported by the PhD program of National Colciencias, call 528 2011.

REFERENCES

[1] D. Wu, Y. T. Hou, W. Zhu, Y.-Q. Zhang and J. M. Peha, "Streaming video over the Internet: approaches and directions," *IEEE Trans. CircuitsSyst. Video Technol.*, vol. 11, no. 3, pp. 282-300, Mar. 2001.

[2] S. B. Barrio and J. Soto Vázquez, "Las posibilidades didácticas y manejo de Video Streaming en las clases de lengua y literatura," *Tejuelo Didáctica Leng. Lit.*, no. 4, pp. 84-100, Mar. 2009.

[3] Y. Liu, B. Du, S. Wang, H. Yang, and X. Wang, "Design and Implementation of Performance Testing Utility for RTSP Streaming Media Server,"in 2010 *1stInt. Conf. on Pervasive Computing Signal Processing and Applications (PCSPA)*, Harbin, China, 2010, pp. 193-196.DOI: 10.1109/PCSPA.2010.55.

[4] N. B. Yoma, J. Hood, and C. Busso, "A real-time protocol for the Internet based on the least mean square algorithm," *IEEE Trans. Multimedia*, vol. 6, no. 1, pp. 174-184, Feb. 2004. DOI:10.1109/TMM.2003.819582.

[5] D. Chu, C. Jiang, Z. Hao, y W. Jiang, «The Design and Implementation of Video Surveillance System Based on H.264, SIP, RTP/RTCP and RTSP», en 2013 Sixth International Symposium on Computational Intelligence and Design (ISCID), 2013, vol. 2, pp. 39-43

[6] G. E. Chanchí Golondrino, F. A. Urbano Ordoñez, W. Y. Campo Muñoz, "Stress tests for videostreaming services based on RTSP protocol," Revista Tecnura, [S.l.], vol. 19, n. 46, pp. 27-36, nov. 2015. DOI: http://dx.doi.org/10.14483/udistrital.jour.tecnura.2015.4.a02.

[7] X. Lei, X. Jiang, and C. Wang, "Design and implementation of streaming media processing software based on RTMP,"in *Image and Signal Processing (CISP), 2012 5th Int. Congr.*,Chongqing,Southwest China, 2012, pp. 192-196. DOI: 10.1109/CISP.2012.6469981

[8] W. Y. Campo, J. L. Arciniegas, R. García, and D. Melendi, "Análisis de Tráfico para un Servicio de Vídeo bajo Demanda sobre Recles HFC usando el Protocolo RTMP,"*Inf. Tecnológica*, vol. 21, no. 6, pp. 37-48,Jan. 2010.

[9] X. Lei, X. Jiang, y C. Wang, «Design and implementation of streaming media processing software based on RTMP», en 2012 5th International Congress on Image and Signal Processing (CISP), 2012, pp. 192-196. DOI: 10.1109/CISP.2012.6469981.

[10] W. Y. Campo, "Modelo de Tráfico para Servicios Interactivos de una Comunidad Académica Virtual, con Contenidos de Audio y Video de Alta calidad,"Ph.D.dissertation, Facultad de Ingeniería Electrónica y Telecomunicaciones. Univ. Of Cauca, Popayán, Colombia, 2014. DOI: 10.4067/S0718-07642010000600006.

[11] A. Acosta, M. S. GarciaVazquez, and J. ColoresVargas, "MPEG-4 AVC/H.264 and VC-1 Codecs Comparison Usedin IPTV Video Streaming Technology,"in *Electronics, Robotics and Automotive Mechanics Conf. –CERMA08*,Cuernavaca, Morelos, México. 2008, pp. 122-126.DOI: 10.1109/CERMA.2008.93.

[12] S. Kim and Y. Yoon, "Video Customization System Using MPEG Standards,"in *Inter. Conf. on Multimedia and Ubiquitous Engineering -MUE 2008*, Busan, Korea, 2008, pp. 475-480.DOI: 10.1109/MUE.2008.34.

[13] T.C. Lin, C.W. Chen, C.C. Lin, and T.-K. Truong, "Very Low Bit Rate Video Coding Using H.264 Codec and Cubic Spline Interpolation,"in *3rd Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP 2007)*,Kaohsiung. Taiwan, 2007, vol. 1, pp. 11-14.DOI: 10.1109/IIHMSP.2007.322.

[14] H. Zhang, G. Jiang, K. Yoshihira, H. Chen, and A. Saxena, "Resilient Workload Manager: Taming Bursty Workload of Scaling Internet Applications," in *Proc. of the 6th Int. Conf. on Autonomic Computing*, New York, NY, USA, 2009, pp. 45–46.DOI: 10.1109/TCAD.2014.2316094.

[15] Z. Hao, C. Guang-Li, and C. Hua-Xiang, "Performance Improvement of DSS Based on High-Definition Video on Demand,"in *2012 Int. Conf. on Com-*

ISSN: 0122-3461 (*impreso*)
2145-9371 (*on line*)

puter Science Service System (CSSS),Nanjing, China, 2012, pp. 761-764.DOI: 10.1109/CSSS.2012.195.

[16] E. Catalán, "Implementación de un servidor de streaming de Vídeo adaptativo,"M.S. thesis, Universidad Politécnica de Cataluña, Cataluña, 2009.

[17] L. Wang and C. Meinel, "Mining the Students' Learning Interest in Browsing Web-Streaming Lectures", in *Computational Intelligence and Data Mining, 2007 - CIDM 2007. IEEE Symp.*, Honolulu, HI, 2007, pp. 194-201.DOI: 10.1109/CIDM.2007.368872.

[18] K. De Vogeleer, A. Popescu, M. Fiedler, and D. Erman, "Content dependency of the traffic control in the darwin streaming server," in *Next Generation Internet (NGI), 2012 8th EURO-NGI Conf.*,Karlskrona, Suecia, 2012, pp. 65-70. DOI: 10.1109/NGI.2012.6252166.

[19] N. Vunand M. Ansary, "Implementation of an embedded H.264 live video streaming system,"in*Consumer Electronics (ISCE), 2010 IEEE 14th Int. Symp.,* Braunschweig, Germany, 2010, pp. 1-4.DOI: 10.1109/ISCE.2010.5523699.

[20] P. V. Phuoc, S.T. Chung, H. Kang, S. Cho, K. Lee, and T. Seol, "Design and implementation of versatile live multimedia streaming for IP network camera,"in *2013 Inter. Conf. on Advanced Technologies for Communications (ATC)*, Hochiminh City, Vietnam, 2013, pp. 525-530.DOI: 10.1109/ATC.2013.6698171.

[21] J. Bailey, "Live Video Streaming from Android-Enabled Devices to Web Browsers", M.S. thesis, Department of Computer Science and Engineering, College of Engineering, University of South Florida, 2011.

[22] S. Jin, H. Li, and Y. Liu, "Research on media player based on Android,"in *9th Inter. Conf. on Fuzzy Systems and Knowledge Discovery (FSKD)*, Chongqing, Sichuan, China, 2012, pp. 2326-2329.DOI: 10.1109/FSKD.2012.6234021.

[23] M. Terbuc, "Use of Free/Open Source Software in e-education,"in *Power Electronics and Motion Control Conf. IPEMC 2006. CES/IEEE 5th Int*. Shanghai, Chinese, 2006, pp. 1737-1742.DOI: 10.1109/EPEPEMC.2006.4778656.