UNIVERSIDAD NACIONAL DEL ALTIPLANO

ESCUELA DE POSGRADO

PROGRAMA DE MAESTRÍA

MAESTRÍA EN INFORMÁTICA

TESIS

OPTIMIZACIÓN DE LA COMUNICACIÓN MULTIPATH TCP USANDO SOCKETS BASADOS EN NOMBRES

PRESENTADA POR:

HELARF FERRER CALSINA CONDORI

PARA OPTAR EL GRADO ACADÉMICO DE:

MAGÍSTER SCIENTIAE EN INFORMÁTICA

MENCIÓN EN GERENCIA DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIONES

PUNO, PERÚ

2017

UNIVERSIDAD NACIONAL DEL ALTIPLANO

ESCUELA DE POSGRADO

PROGRAMA DE MAESTRÍA

MAESTRÍA EN INFORMÁTICA

TESIS

OPTIMIZACIÓN DE LA COMUNICACIÓN MULTIPATH TCP USANDO SOCKETS BASADOS EN NOMBRES

PRESENTADA POR:

HELARF FERRER CALSINA CONDORI

PARA OPTAR EL GRADO ACADÉMICO DE:

MAGÍSTER SCIENTIAE EN INFORMÁTICA

MENCIÓN EN GERENCIA DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIONES

APROBADA POR EL SIGUIENTE JURADO:

PRESIDENTE

Dr. EDGAR ELOY CARPIO VARGAS

PRIMER MIEMBRO

Dr. JOSÉ EMMANUEL CRUZ DE LA CRUZ

SEGUNDO MIEMBRO

M. Sc. LUIS HUGO HUACASI VÁSQUEZ

ASESOR DE TESIS

M. Sc. ALEJANDRO APAZA TARQUI

Puno, 27 de enero de 2017

ÁREA: Redes Teleprocesos
TEMA: Protocolos

DEDICATORIA

A mi madre y hermanos con el más sincero cariño y eterna gratitud, por su constante apoyo y estímulo.

A los instructores de la Academia CISCO de la Universidad Nacional del Altiplano, en especial al Ing. José Cruz, Ing. Ferdinand Pineda, Ing. Christian Romero e Ing. Néstor Huaracha, por su apoyo y colaboración.

AGRADECIMIENTOS

- A la Universidad Nacional del Altiplano y la Escuela de Posgrado, por permitirme cursar mis estudios de Maestría.
- Mis sinceros agradecimientos a los docentes formadores de la maestría en informática y también a la Academia CISCO de la Universidad Nacional del Altiplano, quienes hicieron posible mis estudios de maestría y el desarrollo de la presente investigación.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE CUADROS

INDICE DE ANEXOS

RESUMEN

En la investigación desarrollada se presenta un análisis de las técnicas de multicamino pueden ofrecer para optimizar el rendimiento gestión de direcciones IP, rendimiento y robustez que se realiza en el proceso de índice de transferencia de datos. En concreto se evalúa el protocolo MultiPath, mejorando el rendimiento e incrementando la robustez frente a posibles fallos en una de las rutas. En la operacionalización de variables se toma en cuenta la implementación de la comunicación MultiPath TCP y la integración de sockets basados en nombres como variables independientes y como variable dependiente la optimización de la comunicación MultiPath TCP. El procedimiento de captación y campos de análisis son flujo de datos, tasa de bits, tipo de protocolo, direcciones IP origen, dirección IP destino, Bytes enviados, Bytes recibidos, tiempo de inicio, puerto origen y puerto destino, utilizando software de monitoreo de trafico de red ARGUS. La implementación del servidor MultiPath es realizada en la plataforma Linux, específicamente en Ubuntu 14.04, para el análisis de flujo de datos se efectuó con el software ARGUS en el servidor y cliente, y la herramienta Wireshark para la captura unidades de datos de protocolo. La hipótesis y los objetivos son logrados con los datos evaluados con el método t-student con la aceptación de la hipótesis alterna. Para poder conocer los beneficios y problemas que supone la integración de sockets basados en nombres. Y, a través de una extensa labor se implementó la comunicación MultiPath TCP y se realizó la captura de flujo de datos, se concluye que la comunicación MultiPath se optimiza en términos del índice de transferencia de datos o flujo de datos (Mbps) en un diez por ciento en comparación de una comunicación MultiPath TCP normal.

Palabras Claves: Comunicación, Índice de transferencia de datos, MultiPath TCP, Optimización y Sockets.

ABSTRACT

In the research developed we present an analysis of multichannel techniques can offer to optimize performance IP address management, performance and robustness that is performed in the data transfer index process. In particular the MultiPath protocol is evaluated, improving the performance and increasing the robustness against possible failures in one of the routes. In variables operation takes into account the implementation of MultiPath TCP communication and the integration of sockets based on names as independent variables and as a dependent variable optimization of MultiPath TCP communication. The capture procedure and fields of analysis are data flow, bit rate, protocol type, source IP addresses, destination IP address, Bytes sent, Bytes received, start time, source port and destination port, using software network traffic monitoring ARGUS. The implementation of the MultiPath server is done on the Linux platform, specifically in Ubuntu 14.04, for data flow analysis was performed with the ARGUS software on the server and client, and the Wireshark tool for capturing protocol data units. The hypothesis and the objectives are achieved with the data evaluated with the t-student method with the acceptance of the alternative hypothesis. To know the benefits and problems involved in the integration of sockets based on names. And, through extensive work was implemented the Multipath TCP communication and the data flow capture was performed, it is concluded that the MultiPath communication is optimized in terms of data transfer rate or data flow (Mbps) by ten percent, compared of a normal TCP MultiPath communication.

Keywords: Communication, Data Transfer Index, MultiPath TCP, Optimization and Sockets.

INTRODUCCIÓN

El crecimiento de las telecomunicaciones involucra muchas dificultades, como movilidad, Multihoming y escalabilidad. Las redes actuales tienen como objetivo de brindar la disponibilidad de infraestructura para los servicios y usuarios. Por ejemplo, cabe mencionar que en un datacenter poseen múltiplos caminos paralelos siendo muchos de ellos caminos de *backup* para permitir una redundancia de excelencia, pero carecen de un sistema de control de la congestión de datos en los caminos paralelos, a diferencia el protocolo Multipath permite realizar el balanceo de carga.

El solo hecho de dejar sin servicio a los usuarios o dejar sin conexión un servidor de internet por un corto tiempo pasa a ser una situación crítica e impensable. A situaciones como antes mencionadas los operadores de red tratan de garantizar la disponibilidad de los servicios a través de rutas redundantes. Otro aspecto que toman para asegurar la rentabilidad de sus inversiones los proveedores de servicios de internet u operadores de red utilizan diferentes técnicas de ingeniería de tráfico que permita balancear la carga de tráfico a través de los enlaces disponibles y una de ellas es el Multihoming. En la actualidad la mayoría de dispositivos informáticos son portátiles, móviles y deben de tener acceso a internet en todas partes. Para conectarse a un proveedor de servicio de internet atraviesan por redes heterogéneas con diferentes propiedades. Las direcciones IP son uno de los métodos fundamentales de los dispositivos informáticos para identificarse, conectarse y comunicarse con la mayoría de ordenadores pares.

El inconveniente del Multihoming es que funciona para determinados entornos con IPv4 y el protocolo TCP. Multihoming requiere de un único host para tener

y gestionar múltiples direcciones IP. La gestión de direcciones IP en aplicaciones juega un papel muy importante para garantizar una óptima comunicación entre los hosts y como solución a los problemas.

En la presente investigación se ha visto que el protocolo Multipath TCP proporciona una mayor fiabilidad y aumento en el rendimiento, está diseñado para superar restricciones del protocolo TCP y permite utilizar simultáneamente múltiples caminos entre los pares, en cambio los sockets basados en nombres son una solución frente a la compleja gestión de direcciones IP. Descarga aplicaciones del encabezado, y lo mueve al sistema operativo. Se utiliza un nombre constante en lugar de una dirección IP para establecer una conexión, lo que permite los cambios de dirección IP durante el tiempo de la conexión. Sin embargo, los ordenadores de hoy en día están conectados a internet a través de varias rutas. Con el fin de conseguir la mejor y máxima utilización de los recursos dentro de la red, varias de las rutas disponibles se pueden utilizar simultáneamente.

Para la presente investigación se consideró la hipótesis general “La comunicación MultiPath TCP se optimiza usando sockets basados en nombres” y los objetivos específicos: “Implementar la comunicación MultiPath TCP” e integrar los sockets basados en nombres para optimizar la comunicación MultiPath TCP” y como objetivo principal “ Optimizar la comunicación MultiPath TCP usando sockets basados en nombres” como: En la primera parte de este trabajo es el planteamiento del problema donde se describe el porqué de la presente investigación, los objetivos y la hipótesis.

CAPITULO I

PROBLEMÁTICA DE LA INVESTIGACIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

En la actualidad se da un enorme crecimiento de las tecnologías, así mismo, el despliegue y la popularidad de los dispositivos inalámbricos han provocado un replanteo del esquema clásico de conexión de red, que corresponde la salida al exterior con un único dispositivo de salida y comunicación directa con el internet. Pero la mayoría de los sistemas actuales ya cuentan con varias tecnologías de acceso a radio (Bluetooth, IEEE 802.11, redes de acceso a celular UMTS/LTE, etc.) posibilitando en un futuro que todas ellas puedan entenderse, trabajando de forma simultánea y eficiente. Por otra parte, para entregar servicio a esa gran expansión de dispositivos se ha realizado un sobredimensionamiento de las redes de acceso, teniendo en la mayoría de los casos más de un punto de acceso disponible. Esta redundancia surge para dar un nivel mayor de robustez pero, que podría ser aprovechado para ofrecer un mayor grado de servicio.

En las conexiones tradicionales se utiliza el protocolo de transporte TCP, que solo trabaja con un único camino y mencionamos algunos de los inconvenientes: Ineficiente utilización de recursos de la red, deficiente fiabilidad, bastante tiempo de convergencia en caso de fallos de enlace y el mejor camino elegido por el protocolo de enrutamiento puede no ser necesariamente el camino menos congestionado en la red.

Esto ha ameritado el gran interés para comunidad investigadora y uno de los aspectos más destacables es el protocolo MultiPath TCP, surgido a raíz de la creación de un grupo de trabajo propio en el IETF (Grupo de Trabajo de Ingeniería de Internet). Se puede considerar como una evolución natural de TCP, que permite la transmisión simultánea a través de múltiples caminos en una misma conexión TCP. A través de este mecanismo las aplicaciones pueden beneficiarse de un rendimiento más alto en términos de Throughput (volumen de trabajo o información), así como de una mayor robustez frente a los posibles fallos producidos durante la transmisión.

Los inconvenientes antes mencionados son la motivación para utilizar la comunicación MultiPath TCP. En la capa de transporte, una de las soluciones para abordar un problema multi-homing era extensiones TCP el cual es TCP Multi-Home (TCP-MH) que no se ha aplicado ni desplegados. (A. Matsumoto, 2011) Muchos otros protocolos como TCP paralelo (MTCP) (Hsieh & Sivakumar, 2012), Multi-Path Transmission Control Protocol (M/TCP), MTCP (Zhang, J., Krishnamurthy, Peterson, & Wang.) fueron inventadas para proporcionar capacidad de trayectos múltiples en TCP. Sin embargo, ninguno de estos protocolos llegó a la

corriente principal. Esto redujo las opciones disponibles de protocolos con soporte multi-homing a otros dos protocolos de la capa de transporte, Protocolo Sesión de control de transporte (SCTP) y Multipath TCP (Barre, Paasch, & Bonaventure, 2014).

MultiPath TCP, por otro lado es pertinente y el mejor ajuste para el escenario en el que un host está conectado a la red a través de más de una interfaz. MultiPath TCP es el más ambicioso de extensión para TCP para ser normalizados dentro de la IETF. A pesar de las buenas bondades de MultiPath TCP, una de las limitaciones actuales es MultiPath TCP no tiene compatibilidad en las comunicaciones en los escenarios de redes en redes heterogéneas IPv4. Por lo tanto, no hay una manera de determinar rápidamente si la dirección de IPv4 es óptima para comunicarse con el servidor. Factores como la demora, las tasas de pérdida y la capacidad tienen un gran impacto en el rendimiento de una conexión, pero para el tráfico que va a través de Internet es muy complicado.

1.1.1. Formulación del Problema

¿En qué medida se optimiza la comunicación Multipath TCP usando sockets basados en nombres?

1.1.2. Objetivos de la Investigación.

1.1.2.1. Objetivo General

Optimizar la comunicación MultiPath TCP usando sockets basados en nombres.

1.1.2.2. Objetivos Específicos

- Implementar la comunicación MultiPath TCP.
- Integrar los sockets basados en nombres para optimizar la comunicación MultiPath TCP.

1.1.3. Hipótesis de la Investigación

La comunicación MultiPath TCP se optimiza usando sockets basados en nombres.

CAPITULO II

MARCO TEÓRICO

2.1. ANTECEDENTES

De acuerdo a la investigación realizada por Paasch (2014) en su tesis de Ph.D. titulado “Improving Multipath TCP (Mejoramiento de Multipath TCP)”, en la universidad de Universit’e catholique de Louvain – Bélgica, que, en esta tesis menciona un objetivo “Improving Multipath TCP in heterogeneous environments”o traducido “mejoramiento de Multipath TCP en escenarios heterogeneas” cuyo trabajo concluye en lo siguiente “Up to today, the reactive approach with retransmission and penalization seems to provide reasonably good results. Nevertheless, improvements in the scheduler might be promising to improve the performance of Multipath TCP.” Traducido “Hasta hoy, el enfoque reactivo con la retransmisión y penalización parece proporcionar razonablemente buenos resultados. Sin embargo, las mejoras en el planificador pueden ser prometedores para mejorar el rendimiento de Multipath TCP.”.

Asímismo, Purushothama (2011) en el Royal Institute of Technology - Stockholm, Sweden con su tesis de Master Science titulado “Multipath communication using names”, que tiene como uno de sus objetivos

"Integration of the existing implementation of multi-path TCP with name-based sockets" traducido "Integración de la aplicación existente de trayectoria múltiple TCP con sockets basados en nombres" concluye que en el desarrollo de esta tesis "The integration and testing of MPTCP and sockets was not possible during this thesis work, due to the above mentioned constraints in heterogeneus network", traducido "La integración y evaluación de MPTCP y sockets no fue posible durante el trabajo de esta tesis, debido a las restricciones mencionadas en redes heterogéneas".

En un trabajo de investigación presentada a la IETF (2014) denominada "Multipath: From Theory to Practice" de los autores Barré, Paasch, y Bonaventure, de la universidad de Université catholique de Louvain – Bélgica, tienen como objetivo "based on our implementation of MultiPath TCP in the Linux kernel, we explain how such an implementation can be optimized to achieve high performance and report measurements showing the performance of receive buffer tuning and coupled congestion control." traducido "en base a nuestra implementación de MultiPath TCP en el kernel de Linux, explicamos cómo una aplicación de este tipo puede ser optimizado para lograr un alto rendimiento y reportar mediciones que muestran el desempeño de recibir la sintonización de amortiguación y control de congestión acoplada.", concluyen que "MultiPath TCP is a major extension to TCP that is being developed within the IETF. Its success will depend on the availability of a reference implementation."

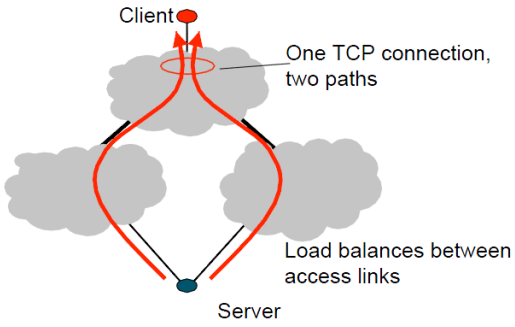
2.2. SUSTENTO TEÓRICO

2.2.1. Protocolo MultiPath TCP

El protocolo MultiPath TCP (MPTCP) es una versión modificada de TCP que implementa un transporte Multi-caminos. Consigue este objetivo mediante la agrupación de múltiples rutas dentro de una única conexión en la capa de transporte, de forma transparente para la aplicación. (Raiciu, Handley, & Wiscik, March, 2011.)

FIGURA 1

VISIÓN DEL PROTOCOLO MULTIPATH TCP



Fuente: Raiciu, Handley, & Wiscik, (Marzo 2011.). Architectural Guidelines for Multipath TCP Development.

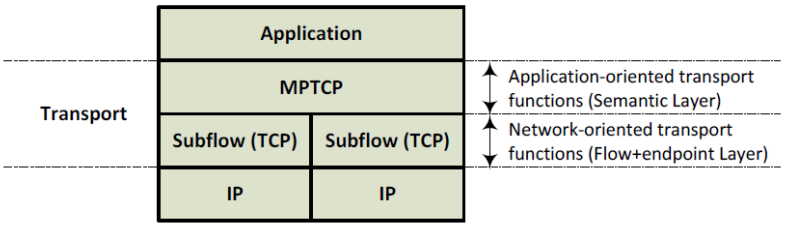
2.2.1.1 Modelo de referencia

La arquitectura de MPTCP está basada en las ideas propuestas por “Transport nextgeneration” (Tng). Tng divide la capa de transporte en dos subcapas (Semántica y Flow+endpoint). MPTCP, que proporciona compatibilidad de aplicaciones a través de la conservación de la semántica de TCP (ordenación global de datos y fiabilidad), es una instancia de la capa semántica "orientada a la aplicación". Y el componente de subflujo TCP, que proporciona compatibilidad con la red al aparecer y comportarse como un flujo TCP en la red, es una instancia

de la capa Flow+endpoint “orientada a la red”. (Raiciu, Handley, & Wiscik, March, 2011.)

FIGURA 2

CAPA DE TRANSPORTE EN MPTCP



Fuente: Raiciu, *et al* (2011.). Architectural Guidelines for Multipath TCP.

Como se puede observar en la figura 2 la instancia MPTCP situada bajo la capa de aplicación debe gestionar múltiples subflujos TCP, para ello implementa las siguientes funciones: (Ford, Raiciu, Handley, & Bonaventure, 2013.)

De acuerdo a Ford *et al* (2013) mencionan las siguientes funciones:

- *Gestión de rutas:* Es la función para detectar y utilizar varias rutas entre dos hosts. Permite publicar direcciones IP alternativas y establecer nuevos subflujos para que se unan a la conexión MPTCP existente.

- *Planificación de paquetes:* Esta función separa el flujo de bytes recibidos desde la aplicación en segmentos para que sean transmitidos sobre uno de los subflujos disponibles. MPTCP hace uso de un mapeo de secuencia de datos, asociando los segmentos enviados en diferentes subflujos a un número de secuencia de nivel de conexión, permitiendo así que los segmentos enviados en diferentes subflujos sean reordenados correctamente en el receptor.

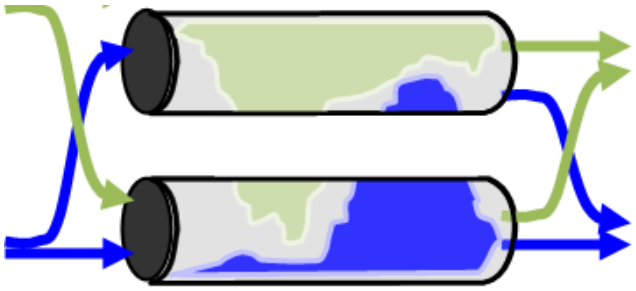
- *Interfaz de subflujo (TCP de ruta única):* El componente de subflujo toma segmentos desde el componente de Planificación de paquetes y los transmite sobre la ruta especificada. MPTCP utiliza TCP por debajo para la compatibilidad con la red.

2.2.1.2 Control de congestión

Uno de los componentes más importantes de TCP es su control de congestión, lo que le permite adaptar el ancho de banda (BW) de forma dinámica en respuesta a las cambiantes condiciones de la red. Para ello cada emisor TCP mantiene una ventana de congestión (cwnd), que regula la cantidad de paquetes que el emisor puede enviar sin esperar un asentimiento, si todo va bien la ventana de congestión crece linealmente y se reduce a la mitad cuando se produce una pérdida de paquetes (posible congestión). Por último, el control de congestión asegura la equidad: cuando múltiples conexiones utilizan el mismo enlace congestionado, cada uno de ellos de forma independiente converge al mismo valor promedio de la ventana de congestión. (Purushothama, 2011)

FIGURA 3

CONGESTIÓN SOBRE MÚLTIPLES RUTAS



Fuente: Ford *et al*. (2013.). TCP Extensions for Multipath Operation with Multiple Addresses.

Ford, Raiciu, Handley, & Iyengar (2011.) mencionan que de acuerdo con el control de congestión de TCP, MPTCP tendría que solventar el inconveniente de la congestión en Múltiples rutas (figura 3) y por lo tanto define tres objetivos que se deben cumplir.

- Garantizar la equidad para TCP. Si varios subflujos de la misma conexión MPTCP comparten un cuello de botella con otras conexiones TCP, MPTCP no debería tomar más BW que TCP.

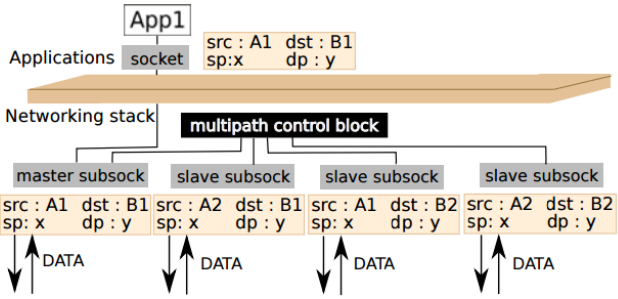
- El ancho de banda de todos los subflujos juntos debería ser al menos igual al de una conexión TCP sobre cualquiera de las rutas utilizadas por la conexión MPTCP, garantizando que existe un incentivo para el despliegue de MPTCP.

- MultiPath TCP debería preferir rutas eficientes, lo que significa que debería enviar un mayor porcentaje de tráfico por las rutas que experimentan menor congestión.

2.2.2. Linux MultiPath TCP arquitectura global

Barre, Paasch, & Bonaventure (2014), muestran el esquema general de la arquitectura MPTCP como se puede apreciar en la figura 4.

FIGURA 4
ARQUITECTURA LINUX- MPTCP



Fuente: Barre, Paasch, & Bonaventure (2014). “Multipath TCP: FromTheory to Practice".

Como se muestra en la figura 4, los tres mayores bloques de construcción de MPTCP son:

- El maestro subsocket
- El esclavo subsocket
- El bloque de control de MultiPath

Se comporta como un TCP regular cuando sólo hay una interfaz disponible en ambos hosts. En tal caso se utiliza solo el subsocket maestro. Incluso cuando múltiples interfaces están disponibles, así es como se establece la conexión inicial entre el cliente y el servidor. Sin embargo, cuando se descubren direcciones adicionales, el cliente intenta establecer un nuevo subflujo mediante la creación de un nuevo subsocket esclavo. Una vez que los subsockets esclavo son creados, entonces el bloque de control de multipath (MPCB) actúa como una capa intermedia entre la aplicación y el subflujo TCP, MPCB es responsable de la ejecución de una planificación y reordena los segmentos en el nivel de conexión. (Barre, Paasch, & Bonaventure, 2014)

2.2.1.1 Establecimiento de una conexión MPTCP

Para explicar de forma general como se establece una conexión MPTCP, se utiliza el esquema de la figura 5, así como las recomendaciones planteadas por Ford *et al* (2013).

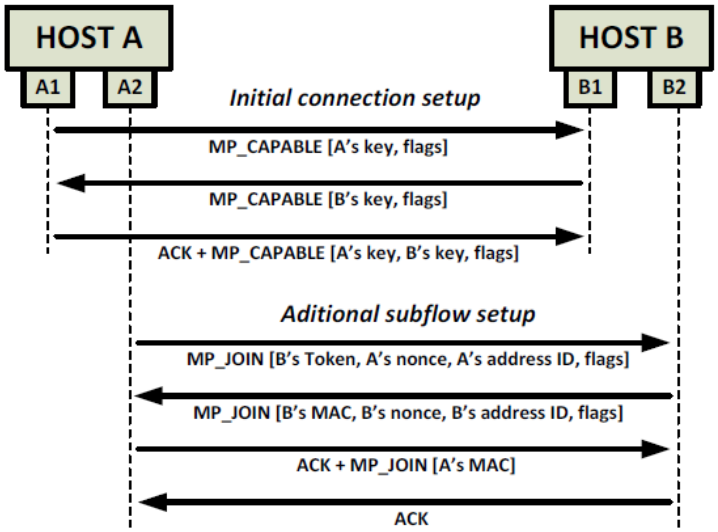
Raiciu, Handley, & Bonaventure (2013), describen, primero el Host A envía un paquete SYN al Host B, igual al procedimiento realizado por una conexión TCP normal. Este paquete contiene sin embargo la opción

MP_CAPABLE en el campo Opciones de TCP. Luego el Host B responde con un paquete SYN/ACK que también contiene la opción MP_CAPABLE (solo cuando se puede utilizar MPTCP). Los dos paquetes también contienen las llaves (key) de Autenticación e información adicional necesaria por MPTCP.

Finalmente el Host A responde con un ACK al Host B, el cual contiene las dos llaves de los paquetes anteriores. En este punto el primer subflujo está configurado y al igual que TCP es una negociación en tres pasos. Posteriormente se pueden establecer subflujos adicionales a través de las demás interfaces. En el ejemplo tanto el Host A como el Host B podrían crear un nuevo subflujo, pero se recomienda que el emisor (Host A) realice esta operación. (Raiciu, Handley, & Bonaventure, 2013)

FIGURA 5

ESTABLECIMIENTO DE CONEXION MULTIPATH TCP



Fuente: Raiciu, Handley, & Wiscik (Marzo 2011.). Architectural Guidelines for Multipath TCP Development.

2.2.1.2 Escenario MultiPath TCP o Multihoming

El Multihoming es una técnica utilizada para incrementar la fiabilidad de la conexión de un host dentro de una red. Está definido como la conexión de un host a más de una red. (Raiciu, Handley, & Wiscik, Marzo 2011.)

MultiPath TCP es la más apropiada cuando los hosts comunicantes tienen más de una interfaz para conectarse a Internet. Un escenario de referencia simple donde es mejor aplicable MultiPath TCP se muestra a continuación. Dos hosts A y B se muestra a continuación se multihomed y multiaddressed. El Host A y el host B tienen dos interfaces para conectarse a Internet. Las direcciones se conocen como 'A1', 'A2' en el host A y 'B1', 'B2' en el host B. Con esta configuración, existen cuatro caminos diferentes entre los dos ejércitos: A1-B1, A1-B2, A2-B1 y A2-B2. Puede haber cualquier número de direcciones disponibles en cada host, siempre y cuando el número de rutas disponibles entre los dos hosts es dos o más. (Raiciu, Handley, & Wiscik, March 2011.)

En la figura 6, se observa un escenario simple de Multihoming, que contiene dos host, un host A, con interfaces A1 y A2 y al otro extremo un Host B, con dos interfaces B1 y B2.

FIGURA 6

ESCENARIO DE USO SIMPLE DE MULTIHOMING

Fuente: Purushothama. (2011). MultiPath Communications

2.2.1.3 Ventajas de MPTCP respecto a TCP

- Mejora de la capacidad de recuperación
- Utilización mejor recurso
- Proporciona más ancho de banda y recursos de la puesta en común
- Mejor utilización para el operador de red
- Separar algoritmo de control de congestión para cada ruta automáticamente y ajustar rápidamente a la congestión en la red. (Beijnum., 2009.)

2.2.3. Sockets Basados en Nombres

Ubillos, Xu, Ming, & Vogt (2011), indican que en el escenario actual, una interfaz de socket tradicional es utilizada por las aplicaciones para iniciar sesiones de comunicación. Esto toma una dirección de socket, que es la combinación de una dirección IP y un puerto como entrada de las aplicaciones. Por lo tanto, la gestión de direcciones IP que incluye la responsabilidad de detectar la dirección IP del interlocutor remoto, el recorrido de los cortafuegos, la selección de un par de direcciones IP locales y remotas que tienen buena conectividad y manejo de cambios de direcciones IP junto con el restablecimiento de sesiones de comunicación activos se deja a las aplicaciones.

Las siguientes son las ventajas de sockets basados en nombres:

- Se supera las deficiencias de las soluciones existentes.
- No reduce la escalabilidad del sistema de enrutamiento global, ya que permite alta significación topológica de direcciones IP.

- Se asegura la coordinación con aplicaciones a través de la semántica explícitas de la nueva interfaz, en lugar de esconderse nueva funcionalidad por debajo de una interfaz existente.

- Evita la sobrecarga administrativa adicional para la accesibilidad de los ejércitos existentes proporcionando accesibilidad a través del DNS directamente, en lugar de una asignación adicional entre el sustituto y direcciones IP regulares.

- Permite la continuidad de las sesiones de comunicación a través de actualizaciones de direcciones IP, debido a la representación bilateral de sesiones de comunicación a través de los nombres de dominio. Ubillos, Xu, Ming, & Vogt (2011)

2.2.3.1. Gestión de dirección en Sockets basados en nombres

Ubillos, Xu, Ming, & Vogt (2011), indican que socket basado en nombres se compone principalmente de tres módulos:

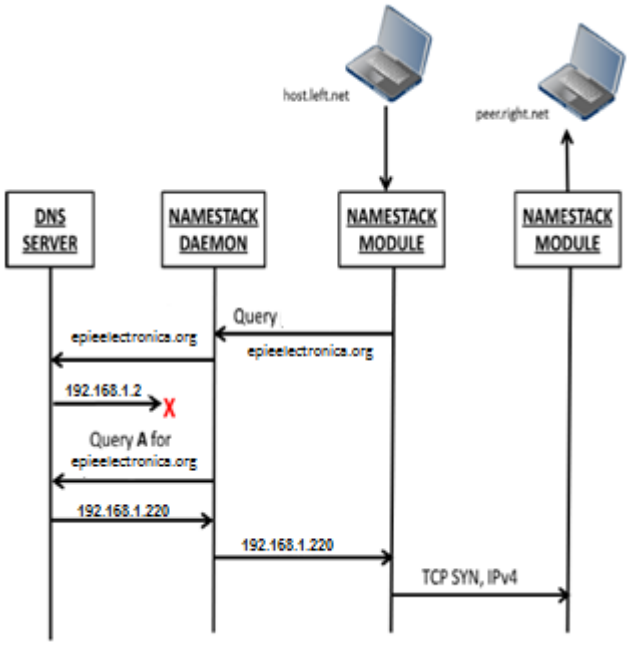
- Módulo del kernel - Consiste en un conjunto de patches que se debe aplicar al kernel con el fin de apoyar a una nueva familia de direcciones AF_NAME para sockets basados en nombres.

- Módulo daemon Namestack - Es un módulo daemon de user-space que se resuelve el nombre solicitado por el módulo namestack poniéndose en contacto con el servidor DNS.

- Módulo Namestack - Es el módulo del kernel que envía un mensaje al namestack daemon para resolver un nombre de host y realiza el intercambio inicial del nombre con el par.

FIGURA 7

GESTION DE DIRECCIÓN EN SOCKETS BASADOS EN NOMBRES



Fuente: Ubillos, Xu, Ming, & Vogt (2011). Name-Based Sockets Architecture draft-ubillos-name-based-sockets-03. IETF.

2.2.4. Protocolo de transferencia de archivos (FTP)

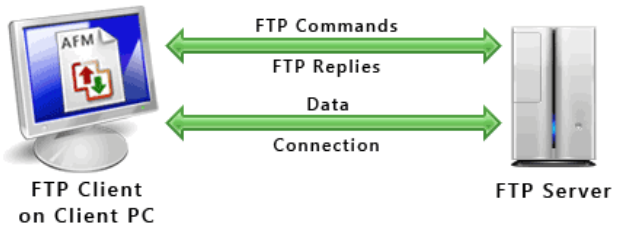
El Protocolo de transferencia de archivos (FTP) es uno de los protocolos más viejos y populares que se encuentran en la Internet hoy día. Su objetivo es el de transmitir archivos exitosamente entre máquinas en una red sin que el usuario tenga que iniciar una sesión en el host remoto o que requiera tener conocimientos sobre cómo utilizar el sistema remoto. FTP permite a los usuarios acceder a archivos en sistemas remotos usando un conjunto de comandos estándar muy simples. (Linuxtopia, 2017)

En la página web llamada Linuxtopia (2017), FTP utiliza una arquitectura cliente/servidor para transferir archivos usando el protocolo de red TCP. Puesto que FTP es un protocolo más antiguo, no utiliza una

autenticación de usuarios y contraseña encriptado. Este escenario se ilustra en la figura 8.

FIGURA 8

FUNCIONAMIENTO DEL PROTOCOLO FTP



Fuente: http://www.deskshare.com/lang/sp/resources/articles/ftp-how-to.aspx

2.2.4.1. Servidor FTP:

Un servidor FTP es un programa especial que se ejecuta en un equipo servidor normalmente conectado a Internet (aunque puede estar conectado a otros tipos de redes, LAN, MAN, etc.). Su función es permitir el intercambio de datos entre diferentes servidores/ordenadores. (Linuxtopia, 2017)

2.2.4.2. Cliente FTP:

Un cliente FTP emplea el protocolo FTP para conectarse a un servidor FTP para transferir archivos. (Linuxtopia, 2017)

CAPITULO III

METODOLOGÍA

3.1 METODOLOGÍA DE INVESTIGACIÓN

Con el escenario de prueba establecido y la instalación/configuración de las maquinas terminales, se ha definido una metodología de evaluación a seguir de forma que podemos recolectar la capacidades de MPTCP y como optimizar la misma, siendo así el primer paso es para determinar qué protocolo de aplicación genera una mayor cantidad de tráfico TCP entre una maquina(s) cliente(s) y el servidor local el cual se realiza la congestión de una de las redes, con ello se podrá analizar el comportamiento de MPTCP, utilizando uno de los caminos disponibles por el cual se transmiten los datos. El segundo pasó fue instalar y configurar el servidor MultiPath-TCP en el servidor local y el cual también se configuro.

El tamaño de fichero a transmitir fue de 2048Mb que es un tamaño considerable para poder realizar el análisis.

Con respecto al escenario de red, se utilizó tres redes, una de las redes comprende al servidor MultiPath-TCP o también denominado local

server, en el que se encuentra un Router y un Switch de borde hacia internet. Las otras redes son utilizadas para generar los caminos que se han denominado eth0 (camino 1) y eth1 (camino 2) ambos utilizados por MultiPath-TCP, para estas redes se incluyeron Routers R1 y R2 que cumplen la función de dividir las redes y crear caminos MultiPath. Se ha trabajo con la capa 4 (capa de transporte) del modelo OSI.

La activación de funcionalidad de las maquinas se da con los siguientes comandos.

- En la máquina de cliente:

  sudo ip link set dev eth0 multipath on

- En la máquina servidor

  sudo ip link set dev eth1 multipath on

Para la habilitación de TCP en el escenario, específicamente en una de las máquinas utilizamos la habilitación a través del siguiente comando.

sudo sysctl -w net.mptcp.mptcp_enable=0

3.1.1 Diseño del modelo de implementación del prototipo MPTCP en el kernel del sistema operativo Linux

Para la implementación de la comunicación multipath, se ha utilizado paquetes el servidor ftp de MPTCP en la plataforma de Linux, computadoras portátiles, un analizador de paquetes de código abierto denominado software Wireshark, software análisis de tráfico de red basado en cliente – servidor como es ARGUS.

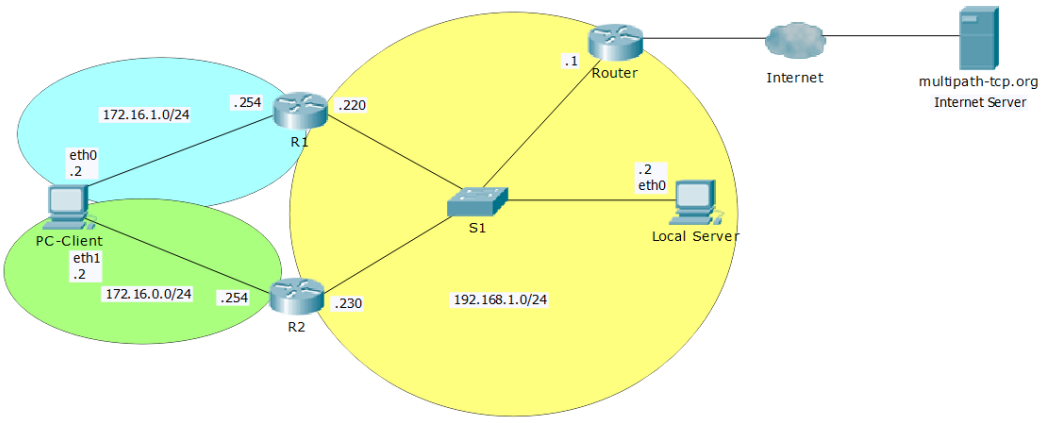
En cuanto al hardware se utilizó equipos CISCO como routers y switch, servidor y computadora portátil.

3.1.1.1 Topología de la red en el laboratorio

En la figura 9, se muestra la topología de red para el análisis de la optimización de la comunicación MultiPath- TCP.

FIGURA 9

TOPOLOGIA DE RED MULTIPATH TCP



Fuente: Elaboración Propia

Se puede observar la red LAN dividida en tres redes 192.168.1.0/24, 172.16.1.0/24 y 172.16.0.0/24; la red 192.168.1.0/24 es la que alberga al servidor MultiPath-TCP, Local Server, que tiene una dirección IP 192.168.1.2 en su interface Ethernet (eth0), en esa misma red se encuentra el Switch S1 y el Router de borde hacia Internet, que tiene una dirección IP 192.168.1.1. La red 172.16.1.0/24 es el primer camino hacia otras redes, que será usado por MultiPath-TCP, en esta red reside la primera interface Ethernet (eth0) del cliente, PC-Client, que tiene la dirección IP 172.16.1.2. La red 172.16.0.0/24 es el segundo camino hacia otras redes, que será usado también por MultiPath-TCP, y en

simultáneo con la primera, en esta red reside la segunda interface Ethernet (eth1) del cliente, PC-Client, que tiene la dirección IP 172.16.0.2. Los Routers R1 y R2 cumplen la función de dividir las redes y crear los caminos para MultiPath-TCP, son necesarios porque MultiPath-TCP usa diferentes caminos en capa de transporte (capa cuatro del modelo OSI) y por lo tanto debe tener diferentes redes o caminos en capa de red (capa tres del modelo OSI), es por ello el uso de dos Routers para crear las redes necesarias. Cada Router tiene dos interfaces, en el caso de R1 tiene una interface para la red 172.16.1.0/24 con la dirección IP 172.16.1.254 que actúa como puerta de enlace de la primera interface Ethernet (eth0) y su otra interface para el lado de la red 192.168.1.0/24 con la dirección IP 192.168.1.220; en el caso de R2 tiene una interface para la red 172.16.0.0/24 con la dirección IP 172.16.0.254 que actúa como puerta de enlace de la segunda interface Ethernet (eth0) y su otra interface para el lado de la red 192.168.1.0/24 con la dirección IP 192.168.1.230.

3.1.2 Instalación de las máquinas (Client - Server)

Luego de tener el escenario de prueba para MTPCP, se pasó a realizar la instalación y configuración de cada una de las máquinas, en nivel de hardware y software.

3.1.2.1 Hardware

Respecto a las máquinas, esta se compone de dos máquinas portátiles con las siguientes especificaciones.

- Maquina Servidor – utiliza un procesador core i3 con una velocidad de 2.13 Ghz, 4 Gb de memoria RAM DDR3, posee una interface de red Fast Ethernet Eth0, esta máquina también posee una interface de red wlan que utiliza la norma IEEE802.11g.

- Maquina Cliente - utiliza un procesador core i5 con una velocidad de 3.8 Ghz, 6 Gb de memoria RAM DDR3, posee una interface de red Fast Ethernet Eth0, esta máquina también posee una interface de red wlan Eth1 que utiliza la norma IEEE802.11g.

En relación de la maquina portátil es una máquina de hardware reciente, y conectadas a los Routers correspondientes para que realice el encaminamiento en el máximo de los 100Mbits/s.

3.1.2.2 Software

Para realizar a instalación del software en cada una de las máquinas, fueron seleccionados con base a las restricciones puestas por los investigadores de la Catholique University de Louvian de Bélgica (UCLouvain, 2017), y también por las restricciones de hardware. Por lo cual, se realizó la instalación en las maquinas portátiles el sistema operativo Ubuntu 14.04 con una arquitectura de 64 bits por ser una máquina de buenos recursos a nivel de hardware.

Con los sistemas operativos correctamente instalado y configurado se en cada una de las máquinas se procedió a realizar las descargas e instalaciones de los paquetes necesarios que permitan el uso adecuado

del protocolo MPTCP (UCLouvain, 2017), cada uno de los paquetes disponibles del repositorio especifico de cada una de sus distribuciones.

La instalación se realiza desde los repositorios de MPTCP. Se ejecuta la línea de comandos en el terminal de Linux. En primer lugar se ejecuta la llave Multipath TCP *(gpg-apt-key)* en cada una de las máquinas para que posteriormente sea posible realizar las descargas de los paquetes MPTCP.

En segundo lugar, fue necesario la creación de un nuevo archivo con el nombre de *mptcp.list* en el directorio */apt/sources.list.d/* en cada máquina y dependiendo a la versión del sistema operativo Linux, en nuestro caso Ubuntu 14.04, luego se agrega la siguiente línea en el fichero creado.

- Para Ubuntu 14.04

deb http://multipath-tcp.org/repos/apt/debian trusty main

En seguida, se ejecuta los dos comandos del repositorio o shell del Linux para que se genere la actualización y así poder instalarse los paquetes base del MultiPath TCP en cada uno de los sistemas.

```
sudo apt-get update

sudo apt-get install linux-mptcp
```

Después de realizar la instalación las maquinas son reiniciada y tenemos instalado el software MultiPath TCP instalado en cada uno de los sistemas.

3.2 CONFIGURACIÓN DE INTERFACES DE RED Y ENRUTAMIENTO

El proceso de encaminamiento dentro de la red se establecieron de acuerdo a las siguientes tablas, para la maquina cliente o la maquina portátil se adicionó un adaptador USB – LAN Rj-45. Al realizar las configuraciones de las interfaces en el entorno gráfico de Ubuntu 14.04, quedó de la siguiente manera.

CUADRO 1

DIRECCIONAMIENTO IP DEL ESCENARIO DE RED

Interfaz	Dirección IP	Mascara de Red	Red	Gateway
mptcp-server-eth0	192.168.1.2	/24	192.168.1.0	192.168.1.1
mptcp-client-eth0	172.16.1.2	/24	172.16.1.0	172.16.1.254
mptcp-client-eth1	172.16.0.2	/24	172.16.0.0	172.16.0.254

Fuente: Elaboración Propia

De acuerdo al cuadro 1, se observa que existen dos interfaces tipo clientes diferentes, la interfaz *eth0* corresponde a la interfaz integrada a la placa madre (fast ethernet) y mientras que la interfaz *eth1* corresponde a la interfaz adicional, agregada mediante un adaptador USB – Rj45. Con la finalidad de realizar la transmisión por diferentes rutas o caminos, en tanto, es una comunicación por dos interfaces Ethernet.

Para el Router se crean tablas de enrutamiento, y para el cual se utiliza las siguientes rutas.

ip rule add from 172.16.0.2 table 1

ip rule add from 172.16.1.2 table 2

Para el otro caso de los Routers se asignó direcciones IP estáticas, con los siguientes comandos dentro del kernel de Ubuntu 14.04.

ip route add 172.16.0.0/24 dev eth1 scope link table 1

ip route add 172.16.1.0/24 dev eth0 scope link table 2

y finalmente hacemos que ambos Routers se comprendan con los siguientes comandos.

ip route add default via 172.16.0.254 dev eth0 table 2

ip route add default scope global nexthop via 10.0.0.1 dev eth0

Para el cliente, se realiza la descarga del servidor FTP de MPTCP. (UCLouvain, 2017)

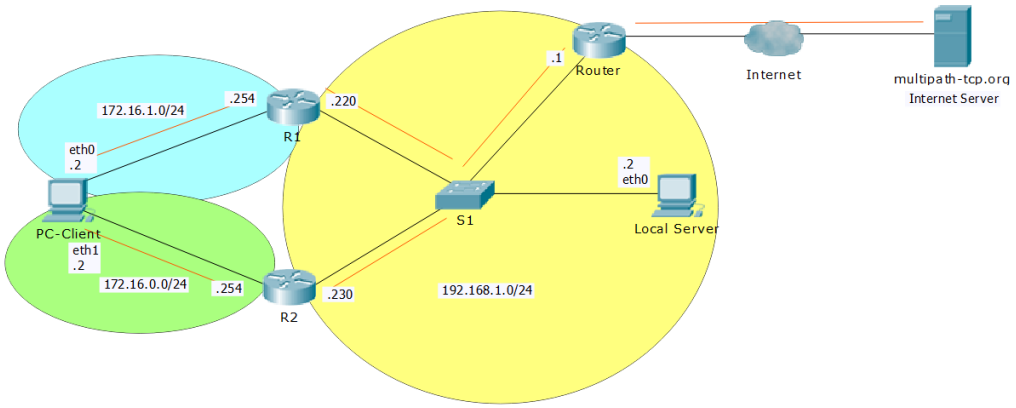
3.2.1 Topología de red con flujo de datos cliente – servidor en internet

En la siguiente topología experimental que se desarrolló se muestra el camino que sigue en el uso de los flujos de los datos.

La línea de color naranja resalta el camino que siguen los dos caminos (MultiPath) desde el cliente, PC-Client, hacia el servidor MultiPath que se ubica en Internet , *multipath-tcp-org;* a diferencia del balanceo de carga, MultiPath no necesariamente iguala el índice de transferencia de datos de los dos caminos, más bien, un camino podría tener mayor índice de transferencia de datos que el otro camino, dependiendo de las condiciones del camino, por ejemplo, el retardo (delay), el ancho de banda (bandwidth), el tráfico en esa red (traffic) y otros factores naturales de una red de computadoras.

FIGURA 10

TOPOLOGÍA DE RED PARA ANÁLISIS DE FLUJO DE DATOS



Fuente: Elaboración Propia

3.2.2 Topología de red con flujo de datos cliente (Pc Client) – servidor local (Local Server)

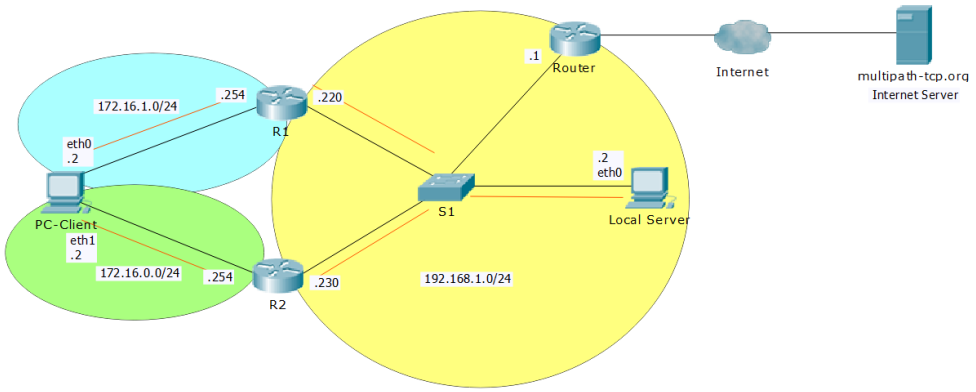
La línea de color naranja resalta el camino que siguen los dos caminos (MultiPath) desde el cliente, PC-Client, hacia el servidor local (Local Server), en la red local cabe resaltar nuevamente que a diferencia del balanceo de carga, MultiPath no necesariamente iguala el índice de transferencia de datos de los dos caminos, más bien, un camino podría tener mayor índice de transferencia de datos que el otro camino, dependiendo de factores que anteriormente ya se describieron.

Cabe resaltar que, si, no ubicamos los dos Routers R1 y R2, no tendremos dos caminos con dos redes distintas, y por ello MultiPath no tiene como elegir más de un camino. También, es necesario resaltar que el servidor de MultiPath tiene sólo una interface en nuestras pruebas por lo cual necesitamos hacer el análisis de flujo de datos desde un cliente con dos interfaces que usen dos caminos hacia un solo destino. Si, usáramos dos interfaces en el cliente y dos interfaces en el servidor, el

cliente usaría las dos interfaces para llegar al servidor, pero este servidor sólo usaría una sola interface aunque las dos estén activas, ya que el cliente debe hacer un flujo de datos hacia un solo destino sin importar cuántos caminos use el cliente.

FIGURA 11

TOPOLOGÍA DE RED DEL FLUJO DE DATOS DE PC-CLIENT HACIA EL SERVIDOR LOCAL



Fuente: Elaboración Propia

3.3 SOFTWARE DE ANÁLISIS DE FLUJO DE DATOS – ARGUS

3.3.1 Software Argus

El software Argus, es un conjunto de herramientas para el análisis de tráfico de red. Está basado en una arquitectura cliente-servidor. Un servidor que será el host o sistema a auditar y un cliente que será el que realice, contra ese host u objetivo, el análisis o auditoría de forma remota. (Alfon, 2010)

3.3.1.1 Proceso de Instalación

Para el proceso de instalación de este software en nuestra versión de Ubuntu 14.04, se utilizó con la versión disponible del repositorio, a partir del kernel de Linux con las siguientes líneas de comando:

- Para el *servidor*

  sudo apt-get install argus-server

Luego se procede a iniciar el servidor del software Argus (-d) en el puerto (-P) 561 y guarda datos (-w) en el archivo *test.argus*

sudo Argus –P 561 –d –w test.argus

- Para el cliente

  sudo apt-get install argus-client

El siguiente paso es, leer (ratop) flujo (-S) desde la dirección IP y puerto del servidor (192.168.1.2:561)

sudo ratop –S 192.168.1.2:561

En seguida, el servidor recibe una solicitud de la dirección IP 192.168.1.220

Y por consiguiente, mientras que el cliente lee los datos, se descarga un archivo del servidor para poder generar el flujo de datos.

FIGURA 12

ENTORNO DE CAPTURA DE DATOS CON ARGUS

```
ratop -S 192.168.1.2:561
Source 127.0.1.1  Version 2.0  16/02/2016 02:30am up 0 days, 00:02:24  Far 60 secs Mar 300 secs
     8 Flows        3 Active        0 Closing           11 Total Records

Microflow Traffic
Flows
Rank        StartTime          Flgs   Type     SrcAddr        Sport   Dir     DstAddr        Dport   SrcPkt   DstPkt   SrcBytes     DstBytes    State
   1 02-16-16 02:27:56.037318  s        tcp   192.168.1.220.36578       ->     192.168.1.2.http       59917    396858    4435989    600832335   CON
   2 02-16-16 02:27:56.044206  *        tcp   192.168.1.230.34509       ->     192.168.1.2.http       31       34        4124       41460       CON
   3 02-16-16 02:27:38.145605           tcp   192.168.1.220.51888       ->     192.168.1.2.561        4        2         336        316         CON
   4 02-16-16 02:28:18.047693       R   igmp   192.168.1.230             ->      224.0.0.22            1        0         54         0           CON
   5 02-16-16 02:28:18.153308       R   igmp     192.168.1.2             ->      224.0.0.22            1        0         54         0           CON
   6 02-16-16 02:28:09.918718       R   igmp     192.168.1.1             ->       224.0.0.1            1        0         50         0           CON
   7 02-16-16 02:28:16.786369       R   igmp   192.168.1.220             ->     224.0.0.251            1        0         46         0           CON
   8 02-16-16 02:28:16.786388       R   igmp   192.168.1.220             ->     224.0.0.252            1        0         46         0           CON
```

Fuente: Elaboración Propia

Luego de observar el flujo de datos, tendemos diferentes tipos de información, y las que se nos muestran por defecto son: StartTime

(Marca de tiempo inicial de la conexión), SrcAddr (Dirección origen), Sport (Puerto origen), DstAddr (Dirección IP destino), Dport (Puerto destino), TotPkts (número total de paquetes del flujo), TotBytes (bytes totales), State (estado de la conexión), Dir (Dirección de flujo), Flgs (Informacion respecto al protocolo. Estado de flujo) y Proto (Protocolo).

El flujo de datos es almacenado el archivo *test.argus* así como lo hemos mencionado anteriormente, luego se cierra sesión del servidor. Para posteriormente observar todos los flujos con las diferentes informaciones capturadas. Para ello usamos los siguientes comandos:

sudo ra –Lo –rn test.argus

Al ejecutar este comando podemos leer (ra) todos los datos (rn) desde el archivo *test.argus,* mostrando los títulos de campo (Lo).

Para poder especificar los campos a mostrar, utilizamos la opción “s”

sudo ra –Lo –s startime srcaddr sport dstaddr dport type srcbyte dstbyte src_bps dst_bps –rn test.argus

y finalmente tenemos la información necesaria, así como se muestra en el cuadro 2 y cuadro 3 para nuestro análisis.

3.4 SOFTWARE DE CAPTURA UNIDADES DE DATOS DE PROTOCOLO

3.4.1 Wireshark

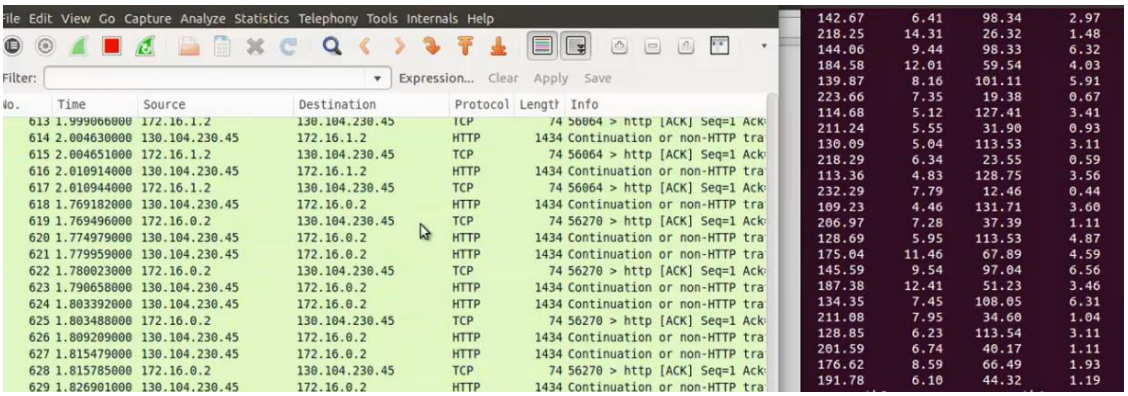
Con todas las configuraciones realizadas, era momento de instalar una herramienta que permitiese almacenar y analizar de un forma precisa los resultados obtenidos en el entorno de red propuesta. La elección de

Wireshark como herramienta para este proyecto de tesis es que tiene muchas herramientas de debe a su versatilidad y permite el análisis al detalle de los flujos de datos recogidos, bien como las peculiaridades introducidas por el MPTCP de los paquetes. Destaca también las características introducidas en el proceso *three-way-handshake.* Además, esta herramienta permite analizar de forma gráfica y estadísticamente el ancho de banda, el número de paquetes transmitidos y perdidas, y el RTT (Round Trip Time) entre el establecimiento inicial de la conexión TCP y los respectivos sub-flujos. En pocas palabras, esta es una herramienta bastante completa y por lo tanto su uso se justifica.

En la figura 13, se puede apreciar la captura de PDUs con la herramienta Wireshark, mientras MultiPath TCP realiza la transmisión por las dos interfaces en paralelo, eth0 y eth1.

FIGURA 13

ENTORNO DE CAPTURA DE PDUs CON WIRESHARK



Fuente: Elaboración Propia

Dentro de nuestra investigación no nos enfocamos en la captura de PDUs, porque, nuestra investigación contempla el análisis del flujo de datos por los diferentes caminos.

CAPITULO IV

RESULTADOS Y DISCUSIÓN

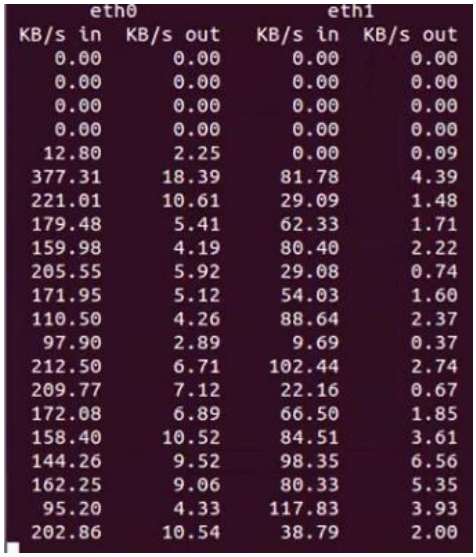
4.1 IMPLEMENTACIÓN DE LA COMUNICACIÓN MULTIPATH TCP

En la figura 14, podemos observar, que los paquetes viajan a través de los dos caminos o interfaces denominados *eth0* y *eth1*, lo cual demuestra que el protocolo Multipath TCP está operativo.

Es uno de los resultados esperados, donde se establece la conexión y la transmisión de información, como se puede observar se está realizando la comunicación en los dos caminos en forma paralela o simultánea.

La interfaz acoplada eth0 recibe la transferencia de hasta 377.31 Kbps como máxima y la interfaz eth1 recibe un tráfico de hasta 117 Kbps, estos valores son variables de acuerdo al balanceo de carga, al descongestionamiento, a la recuperación, throughput que realiza el mismo protocolo MultiPath TCP implementado de acuerdo a nuestro escenario de red creado.

FIGURA 14
OPERATIVIDAD DE LA COMUNICACIÓN MULTIPATH TCP



eth0		eth1	
KB/s in	KB/s out	KB/s in	KB/s out
0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00
0.00	0.00	0.00	0.00
12.80	2.25	0.00	0.09
377.31	18.39	81.78	4.39
221.01	10.61	29.09	1.48
179.48	5.41	62.33	1.71
159.98	4.19	80.40	2.22
205.55	5.92	29.08	0.74
171.95	5.12	54.03	1.60
110.50	4.26	88.64	2.37
97.90	2.89	9.69	0.37
212.50	6.71	102.44	2.74
209.77	7.12	22.16	0.67
172.08	6.89	66.50	1.85
158.40	10.52	84.51	3.61
144.26	9.52	98.35	6.56
162.25	9.06	80.33	5.35
95.20	4.33	117.83	3.93
202.86	10.54	38.79	2.00

Fuente: Elaboración Propia

4.2 ANÁLISIS DEL FLUJO DE DATOS EN LA COMUNICACIÓN MULTIPATH TCP

Para el análisis del modo de operación del protocolo MPTCP y con el fin de probar si esto da lugar a un buen flujo y buen balanceo de carga en comparación de un MultiPath convencional con MultiPath usando sockets basados en nombres, era necesario recoger datos en ambos casos, para después poder comparar los resultados. Por lo tanto, como se señaló anteriormente la herramienta utilizada para recoger los datos fue el Wireshark.

Cabe aclarar que, el flujo de datos es un análisis distinto a la captura de datos, mientras que en la captura de datos se pueden analizar las Unidades de Datos de Protocolos (PDU), mientras que, en un flujo de datos se puede analizar el índice de transferencia de datos o carga, los datos enviados, los datos recibidos, las direcciones IP, puertos, entre otros datos propios de una flujo de datos o transferencia de datos.

El análisis se realiza mediante el uso del software Argus, una herramienta para el análisis de flujo de datos o tráfico de red, esta herramienta es nativa de Linux, por lo que nuestro servidor y cliente son Linux. A continuación se procede a explicar los campos que analizaremos:

- StartTime - El tiempo de inicio del flujo.
- SrcAddr – Dirección IP de Origen.
- Sport – Puerto de origen.
- DstAddr – Dirección IP de destino.
- Dport – Puerto de destino.
- Type – Tipo de protocolo.
- SrcBytes – Bytes enviados desde el origen.
- DstBytes – Bytes enviados desde el destino.
- Src_bps – Bits por segundo en el origen.
- Dst_bps – Bits por segundo en el destino.

4.2.1 Análisis de flujo de datos para la comunicación Multipath TCP

En esta etapa se realiza el análisis de datos para el MultiPath TCP normal, sin el uso de los sockets basados en nombres. Nos concentraremos en los flujos que tienen el campo de tipo de protocolo (type) con el valor TCP. El analizador de flujo de datos sitúa type como TCP y no MPTCP (MultiPath TCP), porque hace el análisis por interface y considera como un TCP convencional. Sin embargo, se notan dos caminos usados hacia el servidor un flujo por la dirección IP de origen 192.168.1.230 y otro flujo con la dirección IP de origen 192.168.1.220,

ambas ocurren simultáneamente como indica el campo del tiempo de inicio del flujo (start time), 02-29-16 12:09:03 p.m. Lo que claramente comprueba que el uso de dos caminos al mismo tiempo se ha dado en el laboratorio.

El puerto de origen es el 47824 que es elegido por el cliente y conocido como un puerto dinámico; el puerto de destino es el 80 que es usado por el servidor web o destino; de esta forma el cliente genera un flujo de datos al descargar un archivo desde el servidor web. Por supuesto este flujo de datos tiene una cantidad de datos enviada desde el servidor al cliente y tiene también un índice de trasferencia de datos.

Para este tema de investigación se realizaron diez análisis, los cuales se encuentran en el anexo 1; en el cuadro 2, se aprecia la información de la captura del primer flujo de datos obtenidos con el software Argus.

CUADRO 2

PRIMER TEST DEL FLUJO DE DATOS MULTIPATH TCP

StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:08:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	755056.18	1078651.69
02-29-16 12:09:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	81078.01
02-29-16 12:09:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	316763.35	35487642.8
02-29-16 12:09:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 12:09:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88

Fuente: Elaboración Propia

4.2.2 Análisis de flujo de datos para Multipath TCP usando sockets basados en nombres

En el siguiente análisis del flujo se hace la misma transferencia de datos que el análisis anterior pero con el uso de sockets basados en nombres

(DNS), el servicio de nombres de dominio, que permite que el cliente ya no use una dirección IP para comunicarse con el servidor, sino que esta vez use un nombre *www.electronicserver.com*, este nombre será traducido a la dirección IP correspondiente por el servidor DNS que reside en el mismo servidor web.

Nos concentraremos en los flujos que tienen el campo de tipo de protocolo (type) con el valor TCP. Se visualizan dos caminos usados hacia el servidor un flujo por la dirección *IP de origen 192.168.1.230* y otro flujo con la dirección *IP de origen 192.168.1.220*, ambas ocurren simultáneamente como indica el campo del tiempo de inicio del flujo (starttime), *02-29-16 13:09:05 p.m.* Lo que claramente comprueba que el uso de dos caminos al mismo tiempo se ha dado en el laboratorio. El *puerto de origen es el 47824* que es elegido por el cliente y conocido como un puerto dinámico; el puerto de destino es el 80 que es usado por el servidor web o destino; de esta forma el cliente genera un flujo de datos al descargar un archivo desde el servidor web. Por supuesto, este flujo de datos tiene una cantidad de datos enviados desde el servidor al cliente y tiene también un índice de trasferencia de datos. Se hicieron diez análisis como se muestra a continuación.

En este análisis se muestra también la consulta DNS, que siempre se muestra al inicio de cada conjunto de flujo de datos y se caracteriza naturalmente por usar el puerto de destino 53 y ser el tipo de protocolo UDP, en el cuadro 3 se muestra sólo el primer flujo de datos obtenidos desde el software Argus.

CUADRO 3

PRIMER TEST DEL FLUJO DE DATOS EN MULTIPATH TCP USANDO SOCKETS BASADOS EN NOMBRES

StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:08:56 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2769.66	71078.11
02-29-16 13:08:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	754056.18	1078651.69
02-29-16 13:09:05 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.3	81078.01
02-29-16 13:09:05 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	325763	36587642.7
02-29-16 13:09:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 13:09:39 p.m.	192.168.1.220	192.168.1.2	2054	60	42	2845694.12	19765605.88

Fuente: Elaboración Propia

4.2.3 Captura de unidad de datos de protocolo utilizando la herramienta Wireshark

A continuación se muestra la captura de PDU o Unidades de Datos de Protocolo, realizado con la herramienta Wireshark. En esta sección se muestran dos PDUs, uno para cada camino, que contienen todos los datos en cada capa del modelo OSI, se resaltan las direcciones IP, los puertos y sobre todo como TCP usa MultiPath.

CASO 1: Ruta o camino 1 de la comunicación MultiPath - TCP

FIGURA 15

CAPTURA DE PDU FUENTE – DESTINO EN LA COMUNICACIÓN MULTIPATH TCP IMPLEMENTADA

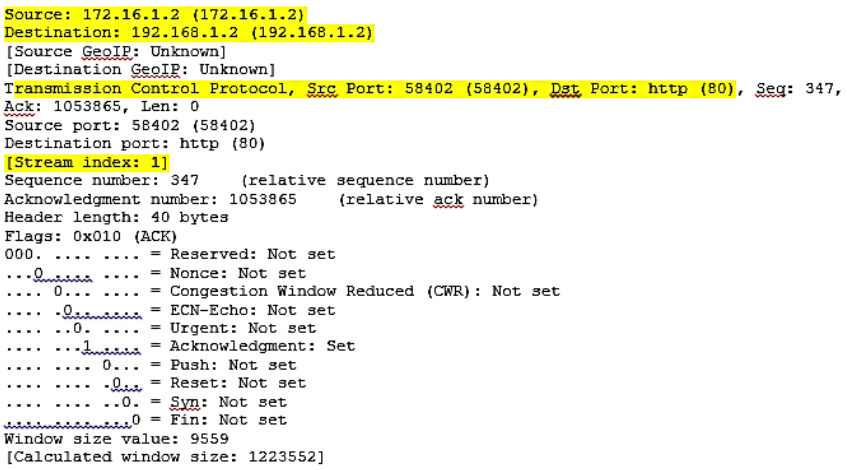
```
No.     Time           Source                Destination           Protocol Length Info
1483 2.179045000    172.16.1.2            192.168.1.2           TCP      74     58402 >
http [ACK] Seq=347 Ack=1053865 Win=1223552 Len=0 TSval=4294957491 TSecr=4294959800

Frame 1483: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Interface id: 0
Encapsulation type: Ethernet (1)
Arrival Time: Feb 15, 2016 22:49:16.312580000 PET
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1455594556.312580000 seconds
[Time delta from previous captured frame: 0.000008000 seconds]
[Time delta from previous displayed frame: 0.000008000 seconds]
[Time since reference or first frame: 2.179045000 seconds]
Frame Number: 1483
Frame Length: 74 bytes (592 bits)
Capture Length: 74 bytes (592 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ip:tcp]
[Coloring Rule Name: HTTP]
[Coloring Rule String: http || tcp.port == 80]
Ethernet II, Src: CadmusCo_0e:b0:4d (08:00:27:0e:b0:4d), Dst: Tp-LinkT_c8:be:7e
(30:b5:c2:c8:be:7e)
```

Fuente: Elaboración Propia

FIGURA 16

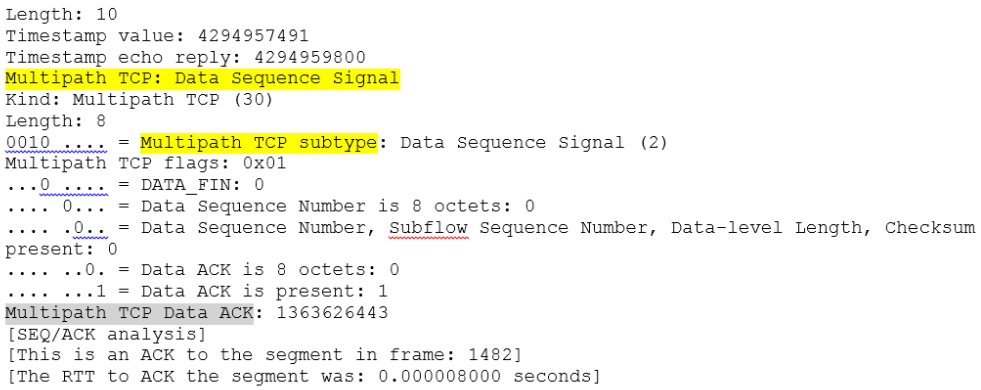
CAPTURA DE PDU EN EL PRIMER CAMINO DE LA COMUNICACIÓN MULTIPATH TCP



Fuente: Elaboración Propia

FIGURA 17

PRIMER CAMINO DE LA COMUNICACIÓN MULTIPATH TCP



Fuente: Elaboración Propia

En el análisis de la captura de PDU se muestra una captura en la primera interface del cliente (primer camino), tiene una dirección de origen 172.16.1.2 y la dirección IP de destino 192.168.1.2, el puerto de origen es el 58402 y el puerto de destino es el 80, el protocolo es TCP con la opción Multipath TCP, lo que confirma el uso de MultiPath (mptcp), cabe resaltar que el dato [Stream index: 1] indica que es el flujo número 1 que hace referencia al primer camino de MultiPath.

CASO 2: Ruta o camino 2 de la comunicación MultiPath TCP

FIGURA 18

CAPTURA DE PDU FUENTE DESTINO DE LA COMUNICACIÓN MULTIPATH TCP

```
No.     Time           Source                Destination           Protocol Length Info
   1486 2.061618000    172.16.0.2            192.168.1.2           TCP      86     42071
> http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=4294957462 TSecr=0 WS=128

Frame 1486: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 1
    Interface id: 1
    Encapsulation type: Ethernet (1)
    Arrival Time: Feb 15, 2016 22:49:16.195153000 PET
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1455594556.195153000 seconds
    [Time delta from previous captured frame: -0.117489000 seconds]
    [Time delta from previous displayed frame: -0.117489000 seconds]
    [Time since reference or first frame: 2.061618000 seconds]
    Frame Number: 1486
    Frame Length: 86 bytes (688 bits)
    Capture Length: 86 bytes (688 bits)
```

Fuente: Elaboración Propia

FIGURA 19

CAPTURA DE PDU EN EL SEGUNDO CAMINO DE LA COMUNICACIÓN MULTIPATH TCP

```
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 42071 (42071), Dst Port: http (80), Seq: 0,
Len: 0
    Source port: 42071 (42071)
    Destination port: http (80)
    [Stream index: 2]
    Sequence number: 0    (relative sequence number)
    Header length: 52 bytes
    Flags: 0x002 (SYN)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...0 .... = Acknowledgment: Not set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..1. = Syn: Set
            [Expert Info (Chat/Sequence): Connection establish request (SYN): server
port http]
```

Fuente: Elaboración Propia

FIGURA 20

CAPTURA DE PDU EN EL SEGUNDO CAMINO DE LA COMUNICACIÓN MULTIPATH TCP

```
Multipath TCP: Join Connection
 Kind: Multipath TCP (30)
     Length: 12
     0001 .... = Multipath TCP subtype: Join Connection (1)
     Multipath TCP flags: 0x00
          .... ...0 = Backup flag: 0
     Multipath TCP Address ID: 2
     Multipath TCP Receiver's Token: 2424259921
     Multipath TCP Sender's Random Number: 2381686962
```

Fuente: Elaboración Propia

En el análisis de la captura de PDU se muestra una captura en la segunda interface del cliente (segundo camino), tiene una dirección de origen 172.16.0.2 y la dirección IP de destino 192.168.1.2, el puerto de origen es el 42071 y el puerto de destino es el 80, el protocolo es TCP con la opción MultiPath TCP, lo que confirma el uso de MultiPath (MPTCP), cabe resaltar que el dato [Stream index: 2] indica que es el flujo número 2 que hace referencia al segundo camino de MultiPath.

4.3 RESULTADOS

Después de analizar el flujo de datos de una comunicación convencional de MultiPath -TCP, de acuerdo al anexo 01, se procede a hallar la media del flujo de datos encontrados en cada camino, la media del primer camino por la dirección 192.168.1.230 es 231918.2085 bps y la media o promedio del segundo camino por la dirección IP 192.168.1.220 es 54627451.01 bps, realizando el proceso de suma de los dos promedios hallamos la media simultáneo es: 54859369.22 bps, que también es *54.86Mbps o 6.86MBps.*

Luego de analizar el flujo de datos para la comunicación de MultiPath utilizando sockets basados en nombres (DNS), de acuerdo al anexo 02, se obtuvo la media de los caminos individuales, la media del primer camino de la dirección IP 192.168.1.230 es 231492.8177 bps y el promedio de segundo camino de la dirección IP 192.168.1.220 es 60654623.85 bps, sumando los dos promedios hallamos el promedio total simultáneo que es 60886116.67 bps, que es también *60.89Mbps o 7.61MBps*.

4.4 PRUEBA ESTADÍSTICA

Para la prueba estadística se tuvo que realizar algunas operaciones para obtener la información necesaria, las cuales eran sumar el flujo de datos de los caminos (eth0 y eth1) y otros casos promediar el flujo de datos de los caminos, así como por ejemplo del cuadro 2 (primer test del flujo de datos MultiPath TCP), se tiene la dirección IP de fuente (SrcBytes) 192.168.1.230 (camino 1) con el puerto 47824 y destino 192.168.1.2 puerto 80, el tipo de protocolo utilizado es TCP, la cantidad de bytes en la fuente es de 20600, la cantidad de bytes en destino 607426 y el flujo de destino es de *2749.65 bps*, de acuerdo a nuestro escenario de red, se realiza la prueba estadística con el flujo de datos de destino (Src_bps), porque es el único valor que varía en todas las pruebas en el mismo camino; considerando la siguiente información para la dirección IP 192.168.1.220 (camino 2), puerto 35908, dirección IP destino 192.168.1.2 puerto 80, protocolo TCP, cantidad de bytes 2373315, cantidad de bytes destino 265887331 y el flujo destino 316763.35 bps. Luego se evalúan sólo los protocolos TCP porque es ahí donde se

realiza el tráfico de red y en ambos caminos, por lo que en esta primera prueba el dato obtenido es la suma del camino 1 (2749.65 bps) y el camino 2 (316763.35 bps) es de 319513.00 bps, así como se muestra en el cuadro 4. Los siguientes datos son encontrados en forma similar y en algunos casos se promedian entre los mismos caminos y se realizan las sumas de ambos caminos. Para verificar mejor el cuadro 4, se utiliza la información obtenida a partir del software ARGUS, que podemos obtener del anexo 1 y el anexo 2.

De acuerdo a los datos obtenidos del flujo en 10 pruebas promedio de ambos caminos, así como se muestra en el cuadro 04:

CUADRO 4

DATOS PROMEDIO DE AMBOS CAMINOS DE LA COMUNICACIÓN MULTIPATH

Promedio de rendimiento MPTCP normal (bps)– Src_bps	Promedio de rendimiento MPTCP utilizando sockets basados en nombres (bps)- Src_bps
319513.00	328512.30
686862.40	699612.40
786334.20	886334.20
120513.00	339513.00
494662.30	606862.40
695234.20	886334.20
188213.00	319513.00
759174.40	786862.40
986334.20	996374.20
319513.00	328413.00

Fuente: Elaboración Propia

Se planteó a nuestras hipótesis como:

H_o= La integración de sockets basados en nombres no optimiza la comunicación MultiPath TCP y

H_1= La integración de sockets basados en nombres optimiza la comunicación MultiPath TCP,

Se utiliza la prueba de normalidad de Shapiro-Wilk P-Valor (comunicación-normal)=0.568 y P-valor (Comunicación - usado sockets)=0.70, que en ambos casos son mayores a un 0.05 por lo cual los datos de la comunicación MultiPath TCP provienen de una distribución normal, así como se puede observar en el cuadro 5.

Las pruebas se realizaron en el software estadístico IBM SPSS v22, y para este proyecto se aplica la prueba t-student, bajo un nivel de significancia de 5% y los tamaños de las muestras son: n_1=10, n_2=10 respectivamente, grados de libertad=9, tenemos que t_c= 3.240 y la significancia=0.010, así como muestra el cuadro 6, por lo tanto, rechazamos la hipótesis nula H_0 y aceptamos H_1: La integración de sockets basados en nombres optimiza la comunicación MultiPath TCP.

CUADRO 5
PRUEBA DE NORMALIDAD

	Shapiro-Wilk		
	Estadístico	gl	Sig.
Promedio de rendimiento normal	0.941	10	0.568
Promedio de rendimiento optimizado	0.857	10	0.070

Fuente: Elaboración Propia

CUADRO 6
PRUEBA DE T-STUDENT

		T	gl	Sig. (bilateral)
Par 1	Promedio de rendimiento normal - Promedio de rendimiento optimizado	3.240	9	0.010

Fuente: Elaboración Propia

CONCLUSIONES

- Existe una diferencia significativa en las medias entre la comunicación MultiPath TCP sin integrar y con la integración de sockets basados en nombres. Por lo cual se concluye que la comunicación MultiPath TCP se optimiza usando sockets basados en nombres. De hecho la comunicación MultiPath TCP sin integrar tiene un índice de transferencia de 54.86Mbps en destino, y es inferior a la comunicación integrada de sockets basados en nombres de 60.89Mbps en destino por lo cual la optimización es en un 10% frente a la comunicación MultiPath TCP normal en destino.

- Es factible la implementación de la comunicación MultiPath TCP (sin nombre), y al realizar las pruebas correspondientes se tuvo un índice de transferencia de 54.86Mps en promedio, las cuales fueron transmitidas por ambos caminos. Cabe destacar que la comunicación MultiPath TCP, es capaz de: utilizar múltiples rutas de red para una sola conexión, utilizar las rutas de red disponibles como el TCP normal, no impedir la conectividad en una ruta en donde funciona TCP normal. Dentro de la topología utilizada, el esquema de control de congestión MPTCP, mide la congestión en cada subflujos y aleja

el tráfico a aquellos con mayor congestión. Finalmente es capaz de soportar hosts multihomed.

- Es factible la integración de sockets basados en nombres a la comunicación Multipath TCP, y se observa que, con el uso de sockets basados en nombres facilita también al ser humano recuerde con facilidad los nombres en vez de las direcciones IP; y sobre todo el uso de DNS para la redundancia de servicios cuando se asigne dos o más direcciones IP a un mismo nombre, y otra ventaja es que, sockets basados en nombres es realiza el balanceo de carga y está relacionado totalmente con la asignación de muchas direcciones IP a un solo nombre de dominio.

RECOMENDACIONES

- Se recomienda la implementación de la comunicación MultiPath en dispositivos móviles, en Smartphone o Tablet, para combinar el uso de las redes de celulares con el de las redes locales, y también es importante que se tenga en cuenta la seguridad cuando se trabaja con el nuevo protocolo, puesto que se pueden generar brechas de inseguridad al ser dinámico con el intercambio de información.

- Se recomienda, la implementación de este protocolo MPTCP integrado con sockets basados con nombres en las instituciones públicas o privadas, para una mejora de los servicios de comunicación entre servidor y cliente.

- Se recomienda realizar trabajos a futuro con direcciones IPv6, y con redes heterogéneas, y esta manera mejorar la congestión de las redes.

BIBLIOGRAFÍA

A. Matsumoto, M. K. (2011). ”TCP Multi-Home Options. *Academic Center for Computing and Media Studies*. Retrieved from http://wit.jssst.or.jp/2003/papers/matsumoto.pdf

Alfon. (2010). *Seguridad y Redes*. Retrieved from https://seguridadyredes.wordpress.com/2010/12/21/argus-auditando-el-trafico-de-red-parte-i/

Barre, S., Paasch, C., & Bonaventure, O. (2014). “Multipath TCP: From Theory to Practice". *Networking*. Retrieved from http://link.springer.com/chapter/10.1007%2F978-3-642-20757-0_35

Beijnum., L. v. (September 2009.). Multipath TCP. *IETF Journal, vol. 5, issue 2,*. Retrieved from http://en.wikipedia.org/wiki/World_IPv6_Day

Dictionary, C. B. (2014). Definition of eyeballs. Cambridge University Press.

Ford, A., Raiciu, C., Handley, M., & Bonaventure, O. (Enero 2013.). TCP Extensions for Multipath Operation with Multiple Addresses. *IETF RFC (6824),* .

Ford, A., Raiciu, C., Handley, M., Barré, S., & Iyengar, J. (Octubre del 2011.). "Coupled Congestion Control for Multipath Transport Protocols". *IETF RFC (6356)*.

Hsieh, H., & Sivakumar, R. (2012). pTCP: An End-to-End Transport Layer Protocol for Striped Connections,. *10th IEEE International Conference on Network Protocols (ICNP'02)*.

Linuxtopia. (2017, 01 18). *Linuxtopia*. Retrieved from http://www.linuxtopia.org/online_books/espaniol/redhat_linux_reference_guide/ch-ftp.html

Mailing, A. (2015, Diciembre 01). Retrieved from http://lists.apple.com/archives/Ipv6-dev/2011/Jul/msg00009.html

Paasch, C. (2014). *Improving Multipath TCP - Thesis Doctoral.* Belgium.

Purushothama, R. (2011). *MULTIPATH COMMUNICATIONS .*

Raiciu, C., Handley, M., & Wiscik, D. (Marzo 2011.). Architectural Guidelines for Multipath TCP Development. *IETF RFC (6182)*. Retrieved from http://tools.ietf.org/pdf/rfc6182.pdf

Ubillos, J., Xu, M., Ming, Z., & Vogt, C. (2011). Name-Based Sockets Architecture draft-ubillos-name-based-sockets-03. *IETF*. Retrieved from http://tools.ietf.org/pdf/draft-ubillos-name-based-sockets-03.pdf

Ubillos., J. (2011). namebasedsockets (Name-based sockets). *Online*. Retrieved from https://github.com/namebasedsockets

UCLouvain. (2017). *Multipath TCP - Linux Kernel Implementation*. Retrieved from http://multipath-tcp.org

Vogt., C. (December 14, 2009.). Simplifying Internet Applications Development With A Name-Based Sockets Interface.

Wing, D., & Yourtchenko, A. (2011). Happy Eyeballs: Trending Towards Success with Dual-Stack Hosts draft-ietf-v6ops-happy-eyeballs-0. *IETF*. Retrieved from http://tools.ietf.org/pdf/draft-ietf-v6ops-happy-eyeballs-01.pdf

wtc@chromium.org. (2015). *chronium*. Retrieved from http://src.chromium.org/viewvc/chrome?view=rev&revision=85934

Yourtchenko., D. W. (2012). Happy Eyeballs: Improving User Experiences with IPv6 and SCTP. *Internet Protocol Journal*.

Zhang, M., J., L., Krishnamurthy, A., Peterson, L., & Wang., R. (2016). A Transport Layer Approach for Improving End-to-End Performance and Robustness Using Redundant Paths".

ANEXOS

ANEXO 1: INFORMACIÓN OBTENIDA CON ARGUS: COMUNICACIÓN MULTIPATH TCP SIN NOMBRES

StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:08:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	755056.18	1078651.69
02-29-16 12:09:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	81078.01
02-29-16 12:09:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	316763.35	35487642.8
02-29-16 12:09:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 12:09:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:11:30 p.m.	192.168.1.220.35918	192.168.1.2.80	tcp	1534044	122984608	610132.74	64948502
02-29-16 12:11:30 p.m.	192.168.1.230.39782	192.168.1.2.80	tcp	17726	549894	9363.53	290474.3
02-29-16 12:13:14 p.m.	192.168.1.2	192.168.1.220	2054	42	60	788732.39	1126760.56
02-29-16 12:13:18 p.m.	192.168.1.220.35921	192.168.1.2.80	tcp	2531305	265888845	750268.4	78808362.53
02-29-16 12:13:18 p.m.	192.168.1.230.32989	192.168.1.2.80	tcp	13360	433316	3960.13	128442.47
02-29-16 12:13:23 p.m.	192.168.1.230	192.168.1.2	2054	120	84	35.56	24.9
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:15:45 p.m.	192.168.1.220.35923	192.168.1.2.80	tcp	3569623	264975817	776634.46	72496315.12
02-29-16 12:15:45 p.m.	192.168.1.230.45245	192.168.1.2.80	tcp	46204	2260640	9699.74	474581.42
02-29-16 12:15:50 p.m.	192.168.1.230	192.168.1.2	2054	60	42	15483870.97	10838709.68
02-29-16 12:15:53 p.m.	192.168.1.220	192.168.1.2	2054	60	42	17142857.14	12000000
02-29-16 12:17:01 p.m.	192.168.1.2	192.168.1.230	2054	42	60	1027522.94	1467889.91
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:19:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	765056.19	978651.66
02-29-16 12:20:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	3749.65	81068.11
02-29-16 12:20:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	116763.32	35487642.8
02-29-16 12:20:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 12:20:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88

StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:22:29 p.m.	192.168.1.2	192.168.1.220	2054	42	60	875335.04	1227621.48
02-29-16 12:22:30 p.m.	192.168.1.220.35918	192.168.1.2.80	tcp	1534044	122984608	410132.76	64948502
02-29-16 12:22:30 p.m.	192.168.1.230.39782	192.168.1.2.80	tcp	17726	549894	9363.25	290474.3
02-29-16 12:23:14 p.m.	192.168.1.2	192.168.1.220	2054	42	60	788732.39	1116760.56
02-29-16 12:23:18 p.m.	192.168.1.220.35921	192.168.1.2.80	tcp	2531305	265888845	565868.41	78808362.53
02-29-16 12:23:18 p.m.	192.168.1.230.32989	192.168.1.2.80	tcp	13360	433316	3960.14	138642.47
02-29-16 12:23:23 p.m.	192.168.1.230	192.168.1.2	2054	120	84	35.56	24.9
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:25:45 p.m.	192.168.1.220.35923	192.168.1.2.80	tcp	3569623	264975817	685634.46	72468515.12
02-29-16 12:25:45 p.m.	192.168.1.230.45245	192.168.1.2.80	tcp	46204	2260640	9599.71	474581.42
02-29-16 12:25:50 p.m.	192.168.1.230	192.168.1.2	2054	60	42	15483870.97	10568709.68
02-29-16 12:25:53 p.m.	192.168.1.220	192.168.1.2	2054	60	42	17125657.14	12002000
02-29-16 12:26:01 p.m.	192.168.1.2	192.168.1.230	2054	42	60	1027526.94	1356889.91
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:28:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	745656.18	1089651.69
02-29-16 12:29:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	80898.01
02-29-16 12:29:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	185463.32	3589642.8
02-29-16 12:29:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 12:29:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28654294.12	19659705.88
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:31:29 p.m.	192.168.1.2	192.168.1.220	2054	42	60	785635.04	1217621.48
02-29-16 12:31:30 p.m.	192.168.1.220.35918	192.168.1.2.80	tcp	1534044	122984608	810156.76	58948502
02-29-16 12:31:30 p.m.	192.168.1.230.39782	192.168.1.2.80	tcp	17726	549894	8963.53	290474.3
02-29-16 12:32:14 p.m.	192.168.1.2	192.168.1.220	2054	42	60	788732.39	1566760.56
02-29-16 12:32:18 p.m.	192.168.1.220.35921	192.168.1.2.80	tcp	2531305	265888845	695268.41	78808366.53
02-29-16 12:32:18 p.m.	192.168.1.230.32989	192.168.1.2.80	tcp	13360	433316	3960.15	128562.47
02-29-16 12:32:23 p.m.	192.168.1.230	192.168.1.2	2054	120	84	35.56	35.9
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:35:45 p.m.	192.168.1.220.35923	192.168.1.2.80	tcp	3569623	264975817	976634.46	72496315.12
02-29-16 12:35:45 p.m.	192.168.1.230.45245	192.168.1.2.80	tcp	46204	2260640	9699.71	474581.42
02-29-16 12:35:50 p.m.	192.168.1.230	192.168.1.2	2054	60	42	15483870.97	10838709.68
02-29-16 12:35:53 p.m.	192.168.1.220	192.168.1.2	2054	60	42	17142857.14	12000000
02-29-16 12:36:01 p.m.	192.168.1.2	192.168.1.230	2054	42	60	1027522.94	1467889.91
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 12:38:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	755056.18	1078651.69
02-29-16 12:39:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	81078.01
02-29-16 12:39:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	316763.32	35487642.8
02-29-16 12:39:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07

02-29-16 12:39:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88
Dst_bps_prom_1	Dst_bps_prom_2	Dst_bps_sum_prom			Src_bps_prom_1:230	Src_bps_prom_1:220	Src_Kbps_sum_prom
231918.2085	54627451.01	54859369.22			6197.576154	555114.1669	561.31
		54.86			6.20	555.11	0.56
Dst_Mbps_sum_prom	Dst_MBps_sum_prom						
54.86Mbps	6.86MBps						

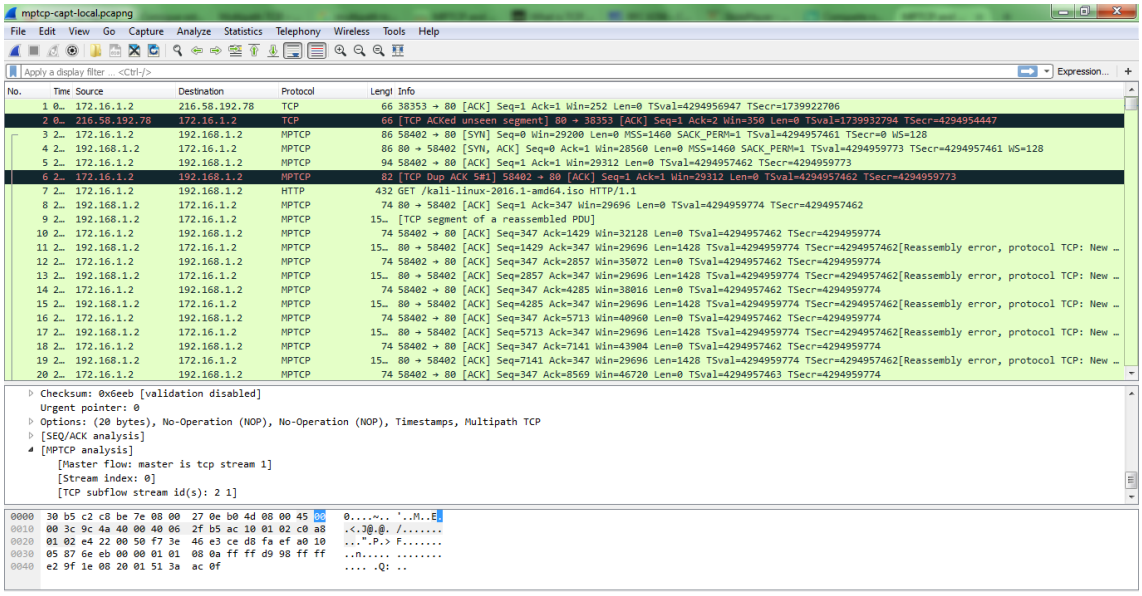
ANEXO 2: INFORMACIÓN ENCONTRADA CON ARGUS: COMUNICACIÓN MULTIPATH USANDO SOCKETS BASADO EN NOMBRES

StartTime	SrcAddr	Sport	DstAddr	Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:08:56 p.m.	192.168.1.230	36844	192.168.1.2.53		udp	40	126	2769.66	71078.11
02-29-16 13:08:59 p.m.	192.168.1.2		192.168.1.220	2054		42	60	754056.18	1078651.69
02-29-16 13:09:05 p.m.	192.168.1.230	47824	192.168.1.2.80		tcp	20600	607426	2749.3	81078.01
02-29-16 13:09:05 p.m	192.168.1.220	35908	192.168.1.2.80		tcp	2373315	265887331	325763	36587642.7
02-29-16 13:09:08 p.m.	192.168.1.230		192.168.1.2	2054		120	84	18.67	13.07
02-29-16 13:09:39 p.m.	192.168.1.220		192.168.1.2	2054		60	42	2845694.12	19765605.88
StartTime	SrcAddr	Sport	DstAddr	Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:11:26 p.m.	192.168.1.230	36844	192.168.1.2.53		udp	42	124	2769.65	72678
02-29-16 13:11:29 p.m.	192.168.1.2		192.168.1.220	2054		42	60	859335.04	1257621.98
02-29-16 13:11:30 p.m.	192.168.1.220	35918	192.168.1.2.80		tcp	1534044	122984608	619832.76	64948502
02-29-16 13:11:30 p.m.	192.168.1.230	39782	192.168.1.2.80		tcp	17726	549894	9763.5	290474.32
02-29-16 13:13:14 p.m.	192.168.1.2		192.168.1.220	2054		42	60	799732.39	1126760.56
02-29-16 13:13:18 p.m.	192.168.1.220	35921	192.168.1.2.80		tcp	2531305	265888845	759668.4	75508362.56
02-29-16 13:13:18 p.m.	192.168.1.230	32989	192.168.1.2.80		tcp	13360	433316	9960.14	128442.47
02-29-16 13:13:25 p.m.	192.168.1.230		192.168.1.2	2054		120	84	46.56	15.9
StartTime	SrcAddr	Sport	DstAddr	Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:15:44 p.m.	192.168.1.230	36844	192.168.1.2.53		udp	40	122	169.6	71378.11
02-29-16 13:15:45 p.m.	192.168.1.220	35923	192.168.1.2.80		tcp	3569623	264975817	876634.46	72496315.12
02-29-16 13:15:45 p.m.	192.168.1.230	45245	192.168.1.2.80		tcp	46204	2260640	9699.74	474581.42
02-29-16 13:15:50 p.m.	192.168.1.230		192.168.1.2	2054		60	42	15483870.97	1056709.68
02-29-16 13:15:53 p.m.	192.168.1.220		192.168.1.2	2054		60	44	17142857.14	12000000
02-29-16 13:17:03 p.m.	192.168.1.2		192.168.1.230	2054		42	62	1012522.94	1567889.91
StartTime	SrcAddr	Sport	DstAddr	Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:19:56 p.m.	192.168.1.230	36844	192.168.1.2.53		udp	40	126	1870.66	62078.01
02-29-16 13:19:59 p.m.	192.168.1.2		192.168.1.220	2054		42	60	755056.18	1078651.69
02-29-16 13:20:03 p.m.	192.168.1.230	47824	192.168.1.2.80		tcp	20600	607426	2749.75	81078.01
02-29-16 13:20:03 p.m	192.168.1.220	35908	192.168.1.2.80		tcp	2373315	265887331	336763.25	35487642.8
02-29-16 13:20:08 p.m.	192.168.1.230		192.168.1.2	2054		120	84	18.67	13.07
02-29-16 13:20:38 p.m.	192.168.1.220		192.168.1.2	2054		60	42	28235294.12	19764705.88
StartTime	SrcAddr	Sport	DstAddr	Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:22:27 p.m.	192.168.1.230	37844	192.168.1.2.53		udp	42	124	1587.6	65878.21
02-29-16 13:22:29 p.m.	192.168.1.2		192.168.1.220	2054		42	60	859335.04	1227621.48
02-29-16 13:22:30 p.m.	192.168.1.220	35918	192.168.1.2.80		tcp	1534044	122984608	600132.76	64948502
02-29-16 13:22:30 p.m.	192.168.1.230	39782	192.168.1.2.80		tcp	17726	549894	9363.54	290474.3

02-29-16 13:23:14 p.m.	192.168.1.2	192.168.1.220	2054	42	60	788732.39	1126760.56
02-29-16 13:23:18 p.m.	192.168.1.220.35921	192.168.1.2.80	tcp	2531305	265888845	600268.36	78808362.53
02-29-16 13:23:18 p.m.	192.168.1.230.32989	192.168.1.2.80	tcp	13360	433316	3960.14	128442.47
02-29-16 13:23:23 p.m.	192.168.1.230	192.168.1.2	2054	120	84	35.56	24.9
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:25:44 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2869.66	72078.01
02-29-16 13:25:45 p.m.	192.168.1.220.35923	192.168.1.2.80	tcp	3569623	264975817	876634.46	72496315.12
02-29-16 13:25:45 p.m.	192.168.1.230.45245	192.168.1.2.80	tcp	46204	2260640	9699.74	474581.42
02-29-16 13:25:50 p.m.	192.168.1.230	192.168.1.2	2054	60	42	15483870.97	10838709.68
02-29-16 13:25:53 p.m.	192.168.1.220	192.168.1.2	2054	60	42	17142857.14	12000000
02-29-16 13:26:03 p.m.	192.168.1.2	192.168.1.230	2054	42	60	1027522.94	1467889.91
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:28:57 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2985.66	72078.11
02-29-16 13:28:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	755056.18	1078651.69
02-29-16 13:29:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	81078.01
02-29-16 13:29:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	316763.35	35487642.8
02-29-16 13:29:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.07
02-29-16 13:29:38 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:31:28 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2668.66	72578
02-29-16 13:31:29 p.m.	192.168.1.2	192.168.1.220	2054	42	60	859335.04	1227621.48
02-29-16 13:31:30 p.m.	192.168.1.220.35918	192.168.1.2.80	tcp	1534044	122984608	810132.76	64948502
02-29-16 13:31:30 p.m.	192.168.1.230.39782	192.168.1.2.80	tcp	17726	549894	9363.5	290474.3
02-29-16 13:32:14 p.m.	192.168.1.2	192.168.1.220	2054	42	60	788732.39	1126760.56
02-29-16 13:32:18 p.m.	192.168.1.220.35921	192.168.1.2.80	tcp	2531305	265888845	750268.41	78808362.53
02-29-16 13:32:18 p.m.	192.168.1.230.32989	192.168.1.2.80	tcp	13360	433316	3960.12	128442.47
02-29-16 13:32:22 p.m.	192.168.1.230	192.168.1.2	2054	120	84	35.56	24.9
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:35:41 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2869.26	62078.11
02-29-16 13:35:45 p.m.	192.168.1.220.35923	192.168.1.2.80	tcp	3569623	264975817	986674.46	72496315.12
02-29-16 13:35:45 p.m.	192.168.1.230.45245	192.168.1.2.80	tcp	46204	2260640	9699.74	474581.42
02-29-16 13:35:50 p.m.	192.168.1.230	192.168.1.2	2054	60	42	15483870.97	10838709.68
02-29-16 13:35:53 p.m.	192.168.1.220	192.168.1.2	2054	60	42	17142857.14	12000000
02-29-16 13:35:59 p.m.	192.168.1.2	192.168.1.230	2054	42	60	1027522.94	1467889.91
StartTime	SrcAddr Sport	DstAddr Dport	Type	SrcBytes	DstBytes	Src_bps	Dst_bps
02-29-16 13:38:58 p.m.	192.168.1.230.36844	192.168.1.2.53	udp	40	126	2869.56	72078.87
02-29-16 13:38:59 p.m.	192.168.1.2	192.168.1.220	2054	42	60	766056.18	1078651.69
02-29-16 13:39:03 p.m.	192.168.1.230.47824	192.168.1.2.80	tcp	20600	607426	2749.65	85678.01
02-29-16 13:39:03 p.m	192.168.1.220.35908	192.168.1.2.80	tcp	2373315	265887331	325663.35	35487642.8
02-29-16 13:39:08 p.m.	192.168.1.230	192.168.1.2	2054	120	84	18.67	13.11

02-29-16 13:39:12 p.m.	192.168.1.220	192.168.1.2	2054	60	42	28235294.12	19764705.88
Dst_bps_prom_1	Dst_bps_prom_2	Dst_bps_sum_prom		Src_bps_prom_1: 230	Src_bps_prom_2: 220	Src_Kbps_sum_prom	
231492.8177	60654623.85	60886116.67		30806.32	60654623.85	#########	
				30806.3	60654623.852	60685.4	
Dst_Mbps_sum_prom	Dst_MBps_sum_prom						
60.89Mbps	7.61MBps						

ANEXO 3: CAPTURA DE PDUS DE LA COMUNICACIÓN MPTCP CON WIRESHARK



ANEXO 4: CASO 1: RUTA O CAMINO 1 DE LA COMUNICACIÓN MULTIPATH TCP

```
No.     Time           Source                Destination           Protocol Length Info
1483 2.179045000    172.16.1.2            192.168.1.2           TCP      74     58402 >
http [ACK] Seq=347 Ack=1053865 Win=1223552 Len=0 TSval=4294957491 TSecr=4294959800

Frame 1483: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Interface id: 0
Encapsulation type: Ethernet (1)
Arrival Time: Feb 15, 2016 22:49:16.312580000 PET
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1455594556.312580000 seconds
[Time delta from previous captured frame: 0.000008000 seconds]
[Time delta from previous displayed frame: 0.000008000 seconds]
[Time since reference or first frame: 2.179045000 seconds]
Frame Number: 1483
Frame Length: 74 bytes (592 bits)
Capture Length: 74 bytes (592 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ip:tcp]
[Coloring Rule Name: HTTP]
[Coloring Rule String: http || tcp.port == 80]
Ethernet II, Src: CadmusCo_0e:b0:4d (08:00:27:0e:b0:4d), Dst: Tp-LinkT_c8:be:7e
(30:b5:c2:c8:be:7e)
Destination: Tp-LinkT_c8:be:7e (30:b5:c2:c8:be:7e)
Address: Tp-LinkT_c8:be:7e (30:b5:c2:c8:be:7e)
.... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
.... ...0 .... .... .... .... = IG bit: Individual address (unicast)
Source: CadmusCo_0e:b0:4d (08:00:27:0e:b0:4d)
Address: CadmusCo_0e:b0:4d (08:00:27:0e:b0:4d)
.... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
.... ...0 .... .... .... .... = IG bit: Individual address (unicast)
Type: IP (0x0800)
Internet Protocol Version 4, Src: 172.16.1.2 (172.16.1.2), Dst: 192.168.1.2
(192.168.1.2)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-
Capable Transport))
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
Total Length: 60
Identification: 0x9f01 (40705)
Flags: 0x02 (Don't Fragment)
0... .... = Reserved bit: Not set
.1.. .... = Don't fragment: Set
..0. .... = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: TCP (6)
Header checksum: 0x2cfe [validation disabled]
[Good: False]
[Bad: False]
Source: 172.16.1.2 (172.16.1.2)
Destination: 192.168.1.2 (192.168.1.2)
[Source GeoIP: Unknown]
[Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 58402 (58402), Dst Port: http (80), Seq: 347,
Ack: 1053865, Len: 0
Source port: 58402 (58402)
Destination port: http (80)
[Stream index: 1]
Sequence number: 347    (relative sequence number)
Acknowledgment number: 1053865    (relative ack number)
Header length: 40 bytes
Flags: 0x010 (ACK)
000. .... .... = Reserved: Not set
...0 .... .... = Nonce: Not set
.... 0... .... = Congestion Window Reduced (CWR): Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
.... ...1 .... = Acknowledgment: Set
```

```
.... .... 0... = Push: Not set
.... .... .0.. = Reset: Not set
.... .... ..0. = Syn: Not set
.... .... ...0 = Fin: Not set
Window size value: 9559
[Calculated window size: 1223552]
[Window size scaling factor: 128]
Checksum: 0x6eeb [validation disabled]
[Good Checksum: False]
[Bad Checksum: False]
Options: (20 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps, Multipath TCP
No-Operation (NOP)
Type: 1
0... .... = Copy on fragmentation: No
.00. .... = Class: Control (0)
...0 0001 = Number: No-Operation (NOP) (1)
No-Operation (NOP)
Type: 1
0... .... = Copy on fragmentation: No
.00. .... = Class: Control (0)
...0 0001 = Number: No-Operation (NOP) (1)
Timestamps: TSval 4294957491, TSecr 4294959800
Kind: Timestamp (8)
Length: 10
Timestamp value: 4294957491
Timestamp echo reply: 4294959800
Multipath TCP: Data Sequence Signal
Kind: Multipath TCP (30)
Length: 8
0010 .... = Multipath TCP subtype: Data Sequence Signal (2)
Multipath TCP flags: 0x01
...0 .... = DATA_FIN: 0
.... 0... = Data Sequence Number is 8 octets: 0
.... .0.. = Data Sequence Number, Subflow Sequence Number, Data-level Length, Checksum
present: 0
.... ..0. = Data ACK is 8 octets: 0
.... ...1 = Data ACK is present: 1
Multipath TCP Data ACK: 1363626443
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 1482]
[The RTT to ACK the segment was: 0.000008000 seconds]
```

ANEXO 5: CASO 2, RUTA O CAMINO 2 DE LA COMUNICACIÓN MULTIPATH TCP

```
No.     Time           Source                Destination           Protocol Length Info
   1486 2.061618000    172.16.0.2            192.168.1.2           TCP      86     42071
> http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=4294957462 TSecr=0 WS=128

Frame 1486: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 1
    Interface id: 1
    Encapsulation type: Ethernet (1)
    Arrival Time: Feb 15, 2016 22:49:16.195153000 PET
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1455594556.195153000 seconds
    [Time delta from previous captured frame: -0.117489000 seconds]
    [Time delta from previous displayed frame: -0.117489000 seconds]
    [Time since reference or first frame: 2.061618000 seconds]
    Frame Number: 1486
    Frame Length: 86 bytes (688 bits)
    Capture Length: 86 bytes (688 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ip:tcp]
    [Coloring Rule Name: HTTP]
    [Coloring Rule String: http || tcp.port == 80]
Ethernet II, Src: Micronpc_0b:24:21 (00:02:0d:0b:24:21), Dst: RealtekS_53:44:58
(00:e0:4c:53:44:58)
    Destination: RealtekS_53:44:58 (00:e0:4c:53:44:58)
        Address: RealtekS_53:44:58 (00:e0:4c:53:44:58)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory
default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Source: Micronpc_0b:24:21 (00:02:0d:0b:24:21)
        Address: Micronpc_0b:24:21 (00:02:0d:0b:24:21)
        .... ..0. .... .... .... .... = LG bit: Globally unique address (factory
default)
        .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 172.16.0.2 (172.16.0.2), Dst: 192.168.1.2(192.168.1.2)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not
ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable
Transport) (0x00)
    Total Length: 72
    Identification: 0xfde4 (64996)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0xcf0e [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 172.16.0.2 (172.16.0.2)
    Destination: 192.168.1.2 (192.168.1.2)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 42071 (42071), Dst Port: http (80), Seq: 0,
Len: 0
    Source port: 42071 (42071)
    Destination port: http (80)
    [Stream index: 2]
    Sequence number: 0    (relative sequence number)
    Header length: 52 bytes
    Flags: 0x002 (SYN)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
```

```
            .... ...0 .... = Acknowledgment: Not set
            .... .... 0... = Push: Not set
            .... .... .0.. = Reset: Not set
            .... .... ..1. = Syn: Set
                [Expert Info (Chat/Sequence): Connection establish request (SYN): server
port http]
                    [Message: Connection establish request (SYN): server port http]
                    [Severity level: Chat]
                    [Group: Sequence]
            .... .... ...0 = Fin: Not set
        Window size value: 29200
        [Calculated window size: 29200]
        Checksum: 0x9107 [validation disabled]
            [Good Checksum: False]
            [Bad Checksum: False]
        Options: (32 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation
(NOP), Window scale, Multipath TCP
            Maximum segment size: 1460 bytes
                Kind: MSS size (2)
                Length: 4
                MSS Value: 1460
            TCP SACK Permitted Option: True
                Kind: SACK Permission (4)
                Length: 2
            Timestamps: TSval 4294957462, TSecr 0
                Kind: Timestamp (8)
                Length: 10
                Timestamp value: 4294957462
                Timestamp echo reply: 0
            No-Operation (NOP)
                Type: 1
                    0... .... = Copy on fragmentation: No
                    .00. .... = Class: Control (0)
                    ...0 0001 = Number: No-Operation (NOP) (1)
            Window scale: 7 (multiply by 128)
                Kind: Window Scale (3)
                Length: 3
                Shift count: 7
                [Multiplier: 128]
            Multipath TCP: Join Connection
             Kind: Multipath TCP (30)
                Length: 12
                0001 .... = Multipath TCP subtype: Join Connection (1)
                Multipath TCP flags: 0x00
                    .... ...0 = Backup flag: 0
                Multipath TCP Address ID: 2
                Multipath TCP Receiver's Token: 2424259921
                Multipath TCP Sender's Random Number: 2381686962
```