

ISSN: 1390-0129

Revista Politécnica



ISSN: 1390-0129

Revista Politécnica



Volumen 35 No 1 Febrero 2015

La Revista Politécnica es una publicación semestral de la Escuela Politécnica Nacional que pone al alcance de los sectores académico y productivo los desarrollos científicos y tecnológicos en las áreas de las ingenierías y ciencias.

Está incluida en:

- Latindex: Sistema Regional de información en línea para Revistas Científicas de América Latina, el Caribe, España y Portugal

Se autoriza la reproducción total o parcial de su contenido siempre y cuando se cite la fuente. Los conceptos expresados son de responsabilidad exclusiva de sus autores.

Contactar con la Coordinación de Edición:

danilo.chávez@epn.edu.ec
oscar.camacho@epn.edu.ec
iliana.carrera@epn.edu.ec

Teléfono: (+593) 3976000 ext. 5218

Guía para autores:

http://revistapolitecnica.epn.edu.ec/repo_files/autor_manual.pdf

Determinación de Invariantes en Grandes Centros de Datos Basados en Topología Fat-Tree

Ortiz J.*; Londono J.**; Novillo F.***; Ampuno A.*; Chávez M.*

*Universidad Politécnica Salesiana, Facultad de Ingenierías, Guayaquil, Ecuador
jortiz@ups.edu.ec; {aampuno; mchavezc}@est.ups.edu.ec

** Universidad Pontificia Bolivariana, Facultad de Ingeniería en Tecnologías de la Información y la Comunicación, Medellín, Colombia
jorge.londono@upb.edu.co

*** Escuela Superior Politécnica del Litoral, CIDIS-Facultad de Ingeniería en Electricidad y Computación, Guayaquil, Ecuador
fnovillo@espol.edu.ec

Resumen: Durante los últimos años ha existido un fuerte incremento en el acceso a internet, causando que los centros de datos (DC) deban adaptar dinámicamente su infraestructura de red de cara a enfrentar posibles problemas de congestión, la cual no siempre se da de forma oportuna. Ante esto, nuevas topologías de red se han propuesto en los últimos años, como una forma de brindar mejores condiciones para el manejo de tráfico interno, sin embargo es común que para el estudio de estas mejoras, se necesite recrear el comportamiento de un verdadero DC en modelos de simulación/emulación. Por lo tanto se vuelve esencial validar dichos modelos, de cara a obtener resultados coherentes con la realidad. Esta validación es posible por medio de la identificación de ciertas propiedades que se deducen a partir de las variables y los parámetros que describen la red, y que se mantienen en las topologías de los DC para diversos escenarios y/o configuraciones. Estas propiedades, conocidas como invariantes, son una expresión del funcionamiento de la red en ambientes reales, como por ejemplo la ruta más larga entre dos nodos o el número de enlaces mínimo que deben fallar antes de una pérdida de conectividad en alguno de los nodos de la red. En el presente trabajo se realiza la identificación, formulación y comprobación de dos invariantes para la topología Fat-Tree, utilizando como software emulador a mininet. Las conclusiones muestran resultados concordantes entre lo analítico y lo práctico.

Palabras clave: Invariantes de red, topologías, Fat-tree, simulación, emulación.

Abstract: In recent years there has been a sharp increase in access to internet, causing data centers (DC) should dynamically adapt its network infrastructure to face possible problems of congestion, which is not always given in a timely manner. Given this, new network topologies have been proposed in recent years as a way to provide better conditions for handling internal traffic, however it is common for the study of these improvements the need to recreate the behavior of a real DC in models of simulation/emulation. Therefore it becomes essential to validate these models, in order to obtain consistent results with reality. Such validation is possible through the identification of certain properties which are derived from the variables and parameters that describe the network and are maintained in the DC topologies for different scenarios and/or configurations. These properties, known as invariant, are an expression of the operation of the network in real environments, such as the longest path between two nodes or the minimum number of links that must fail before a loss of connectivity on one of the nodes of the network. In this paper, the identification, formulation and testing of two invariants for the Fat-Tree topology is performed, using MiniNet as the software emulator. The conclusions show good agreement between the analytical and the practical.

Keywords: Network invariants, topologies, Fat-tree, simulation, emulation.

1. INTRODUCCION

Durante los últimos años ha existido un fuerte incremento en el acceso a internet provocado en gran parte por la extensa variedad de servicios que se ofrecen [8]. Este hecho ha causado que los DC se vean obligados a adaptar

dinámicamente su infraestructura de red para soportar el incremento de tráfico generado; de otra manera se podrían provocar problemas de congestión, bajo rendimiento de las aplicaciones, alta tasa de errores y pérdida de paquetes, entre otros [20] [11] [19] [13]. Tal adaptación no siempre se da de forma oportuna y depende en muchos casos de nuevas inversiones en equipos de comunicación y/o enlaces, lo que

supone también un incremento en la complejidad de la red, desde el punto de vista de gestión y configuración.

De cara a resolver estos problemas, en los últimos años nuevas topologías de red para los DC se han presentado [3], de manera que se brinden mejores condiciones para el manejo del tráfico interno, mediante la aplicación de mecanismos que incluyen la posibilidad de enrutamiento a través de rutas paralelas, acortamiento de distancias entre nodos y el aumento de tolerancia ante fallos, entre otros. Por ejemplo VL2 [9] evita el tráfico broadcast ARP y DHCP mediante el uso de servidores de directorio para la resolución de direcciones. Con PortLand [16] se propone un mecanismo de enrutamiento en capa 2 empleando las propiedades de la topología. Beube [10] soporta varias aplicaciones que requieran de ancho de banda intensivo mediante múltiples caminos paralelos cortos entre cada par de hosts.

Sin embargo previo a despliegues de infraestructura de DC, es importante validar dichos escenarios en un ambiente lo más real posible. En este contexto, recrear el comportamiento de los DC, mediante modelos de simulación/emulación [7], permite predecir algunos aspectos cruciales de la operatividad de la red. Debido a esto, se vuelve esencial validar dichos modelos con el fin de que exista coherencia entre los resultados que se obtienen de la simulación/emulación y los resultados que se obtendrían de un DC real. Una técnica que permite validar estos modelos se basa en la determinación de invariantes, la cual consiste en identificar ciertas propiedades de la red, que se deducen a partir de las variables y/o parámetros que la describen, y que se mantienen en las topologías de los DC, para diversos escenarios y/o configuraciones.

Un invariante, así por ejemplo, es la ruta más larga entre dos nodos, el número de enlaces óptimo o el mínimo número de enlaces que deben fallar antes de que un nodo pierda conectividad. Debido a esto, los invariantes se convierten en una muy buena referencia del funcionamiento real de un DC, para los ambientes de simulación/emulación.

Una metodología para la obtención de invariantes implica hacer una revisión analítica de la topología, considerando ambientes con y sin tráfico, con pocos y muchos hosts y dispositivos de red. Finalmente considerar la interacción con los protocolos utilizados en la red, por ejemplo para acceso al medio, enrutamiento y transporte. De esta revisión se obtienen los invariantes candidatos.

En la comunidad científica se han realizado algunos trabajos al respecto, así por ejemplo en [14] se proponen diez invariantes para servidores web de internet, los cuales permiten caracterizar el tráfico HTTP y facilitar el estudio de mejoras en el rendimiento del servidor. La problemática de realizar simulaciones en la red Internet se expone ampliamente en [20] formulando dentro de sus soluciones, la búsqueda de invariantes, y en la que se proponen siete alternativas: una de ellas relacionada a la topología. El tráfico generado por juegos de red en Internet es analizado en [1], determinando tres invariantes para el tiempo entre paquetes (IPT). Una metodología para el diseño de arquitecturas de red tomando en cuenta a los invariantes se propone en [4]; en este caso se los considera puntos fijos que limitan la evolución de las arquitecturas.

Por lo tanto, en el presente trabajo se propone identificar, formular y comprobar, dos invariantes de red para grandes centros de datos, considerando una de las topologías más utilizadas en el análisis del comportamiento del tráfico en DC, como lo es Fat-Tree. En particular esta topología será emulada utilizando software ampliamente aceptado en la comunidad científica, como lo es mininet [6], y mediante el uso de una herramienta de generación de tráfico, se modelará un escenario que permite recrear casos reales de tráfico en un DC. Esto permitirá comprobar el cumplimiento de los invariantes formulados, sobre un escenario emulado. El resto del documento está organizado como se explica a continuación. En la sección II se describirá el escenario de la topología sobre la cual se va a trabajar. En la sección III, dos invariantes serán propuestos de forma analítica. En la sección IV los invariantes serán comprobados experimentalmente, y comparados con los obtenidos analíticamente. Finalmente las conclusiones del artículo son presentadas.

2. MODELADO DEL SISTEMA

2.1 Descripción del escenario

El escenario bajo estudio considera un DC con topología Fat-Tree de Al-Fares [15], una instancia de la topología Clos [21]. Esto debido a que en los últimos años, se ha referenciado su uso en varias propuestas de la comunidad científica.

Fat-Tree está compuesta fundamentalmente por switches de iguales características y que se encuentran organizados jerárquicamente en n capas denominadas de núcleo, de agregación y de frontera. Adicionalmente entre estas dos últimas se forman agrupaciones entre switches denominados Pods (P). Al interior de los P , los switches compuestos por k puertos, utilizan $k/2$ puertos para hacer uplink y $k/2$ puertos para hacer downlink. En la Fig. 1 se puede apreciar una topología Fat-Tree con $n=3$. La capa de núcleo está compuesta por un conjunto de k switches (CSW) con k puertos activos que permiten el tráfico hacia y desde redes externas con el resto del DC, así como tráfico interno entre P . A partir de aquí los hosts (h) se conectarán a los P_K , utilizando los $k/2$ puertos disponibles de los switches de la capa de frontera (ESW). El número máximo de hosts h_{MAX} es igual a $K_3/4$.

Debido a que este trabajo se centra en analizar sólo el tráfico interno de la topología, tres sentidos son posibles para los flujos del DC: entre hosts de un mismo switch, entre hosts de diferente switch pero que pertenecen a un mismo P , y entre hosts de diferente P . El tráfico es de tipo unicast, para todos los pares de hosts que son seleccionados de forma aleatoria, pero basados en una distribución de probabilidad uniforme, pues se asume la comunicación con servicios internos como bases de datos o repositorios de archivos. Por otro lado, los flujos del tráfico son UDP y la distribución de probabilidad de los tiempos entre transmisiones es exponencial; distribución que ha sido validada en la literatura para el caso de un DC [5]. En la Fig. 1 se puede apreciar el flujo $f_{i,j}$ entre h_5 y $h_{(MAX-1)}$.

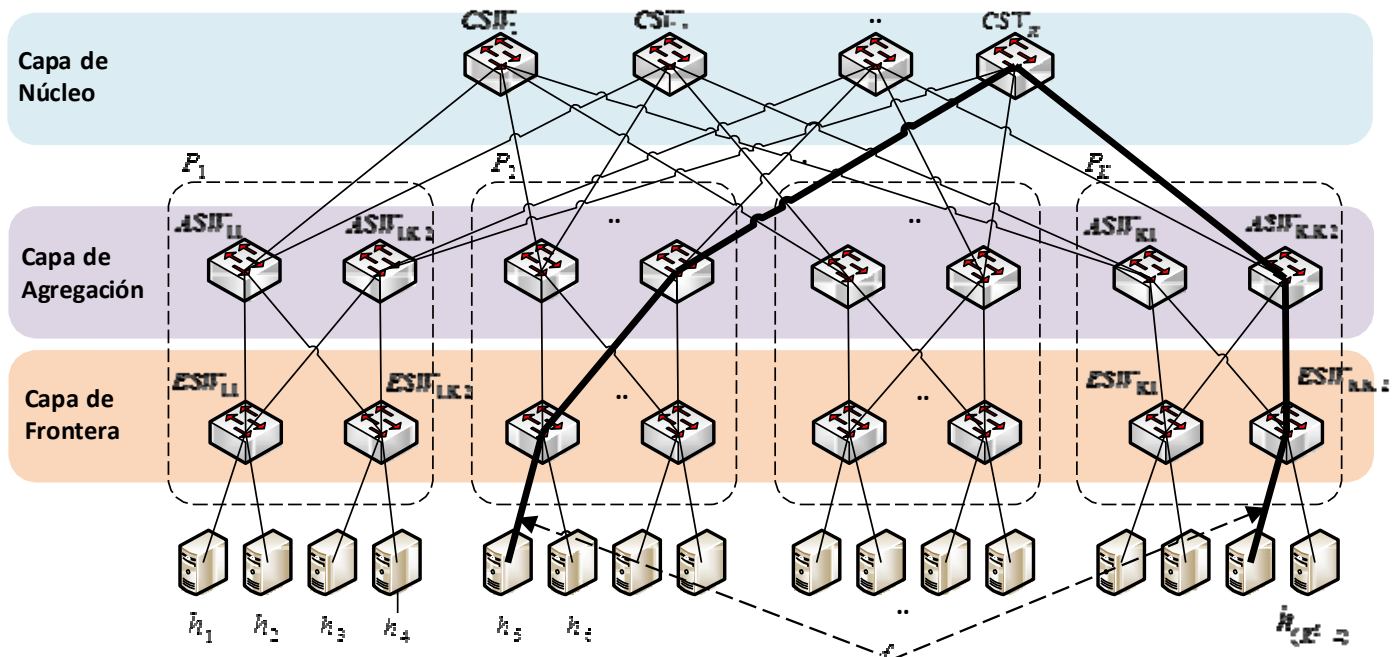


Figura 1. Topología Fat-Tree.

3. FORMULACIÓN DE LOS INVARIANTES

Para formular adecuadamente los invariantes primero es necesario conocer cómo funciona la topología y luego identificar características de la red que puedan ser debidamente cuantificadas y representadas en expresiones matemáticas, ya que en ambientes experimentales, los invariantes validarán la simulación/emulación. Los siguientes invariantes se proponen para un DC con Fat-Tree, sin embargo se fundamentan en conceptos que pueden ser aplicables a otras topologías. Estos invariantes se relacionan con algunos de los parámetros característicos de las topologías [12], y son los siguientes:

3.1 Diámetro de la red

Un invariante conocido es la longitud máxima del camino más corto entre todo par de h de la red, o diámetro de la misma. La utilidad de este invariante radica en conocer el esfuerzo que tienen que hacer los paquetes de un flujo para viajar de extremo a extremo.

En una topología Fat-Tree, la mayor distancia se da cuando se comunican dos h que se encuentran en diferentes P , debido a que se necesita llegar hasta la capa más alta para dar paso al tráfico entre P . De la Fig. 1, para un $f_{5,15}$, el número de saltos requeridos en el tramo ascendente es igual a $n-1$ (no se considera al enlace entre h_5 y $ESW_{2,1}$ como salto), mientras que el número de saltos requeridos en el tramo descendente es igual a $n-1$. Por lo tanto el tramo total para $f_{5,15}$ necesitará de un número total de saltos igual a $2n-1$. Entonces D_{NET} queda definido de la siguiente forma:

$$D_{NET} = 2n - 1 \quad (1)$$

Por otro lado, este invariante no considera que ante un mayor flujo de información, la probabilidad de saturación de los enlaces también aumentará, obligando al mecanismo gestor del enrutamiento de la red, seleccionar rutas alternas que faciliten la libre circulación de los datos. Muchas veces el tomar estas rutas alternas, implica enviar los paquetes por un tramo más largo de extremo a extremo, por lo que el número de saltos para un $f_{i,j}$ también aumenta. En la Fig. 2 se puede apreciar un caso extremo para $f_{5,15}$ en donde todos los posibles caminos redundantes se han saturado, dejando como única alternativa para el encaminamiento de los paquetes, a la señalada en la Fig. 2.

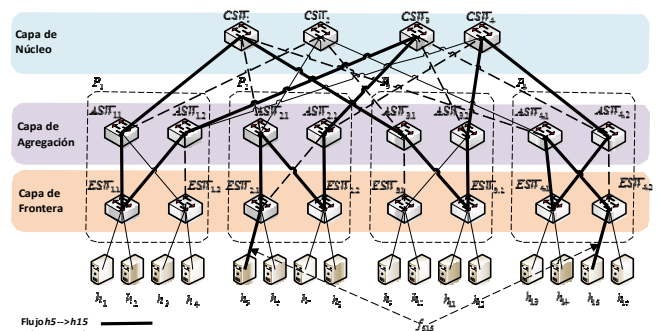


Figura 2. Topología Fat-Tree con enlaces saturados.

Es en este caso que se puede determinar un D'_{NET} que sí considere la saturación de los enlaces. En la Fig. 3 se muestra otra perspectiva de la red, con los ascensos y descensos entre capas que realiza $f_{5,15}$ en su recorrido de extremo a extremo. Los tramos ascendentes o descendentes que atraviesan una capa, tienen un número de saltos igual a $n-2$, mientras que los

tramos ascendentes o descendentes que atraviesan dos capas, tienen un número de saltos igual a $n-1$. A esto hay que adicionar un salto correspondiente a la llegada al h_{15} .

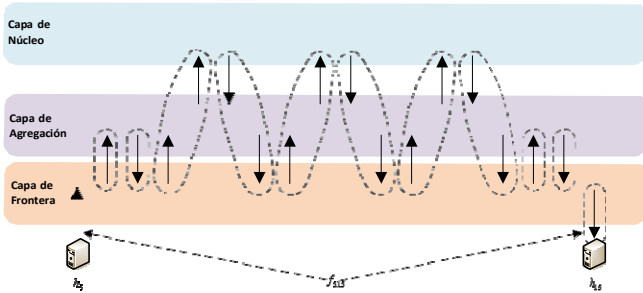


Figura 3. Resumen de saltos entre capas para $f_{5,15}$.

Por lo tanto el invariante D'_{NET} se define de la siguiente forma:

$$\begin{aligned} D'_{NET} &= 2(n-2) + 6(n-1) + 2(n-2) + 1 \\ D'_{NET} &= 10n - 13 \end{aligned} \quad (2)$$

3.2 Longitud de ruta promedio

Otro invariante de utilidad es la longitud de ruta promedio L_{AV} , el cual permite dimensionar el esfuerzo promedio que realizan todos los paquetes durante el desarrollo normal del tráfico de red.

Si se consideran todos los caminos más cortos entre cada uno de los hosts, y se los promedia sobre el número total de caminos, se obtiene la longitud de la ruta promedio L_{AV} de la topología, y cuya expresión matemática se define de la siguiente manera:

$$L_{AV} = \frac{\sum_{h_i=1}^{h_{MAX}-1} \left[\sum_{h_j=h_i+1}^{h_{MAX}} L(h_i, h_j) \right]}{\sum_{n=1}^{n-1} n} \quad (3)$$

Donde $L(h_i, h_j)$, es la longitud mínima o número de saltos mínimo entre dos hosts h_i y h_j . En esta expresión se garantiza que no existan pares de h repetidos. Para el Fat-Tree de la Fig. 1 con $k=4$, el número de saltos mínimo solo puede tomar tres valores: 0 cuando los hosts pertenecen a un mismo switch, 3 cuando los hosts pertenecen a diferentes switches de un mismo P , y 5 cuando los hosts pertenecen a diferentes P .

4. COMPROBACIÓN DE LOS INVARIANTES

En esta sección se describe la forma en como el escenario fue emulado, y luego como se contrastan los resultados experimentales con los analíticos.

4.1 Configuración de la emulación

Para recrear el escenario propuesto se utilizaron varias herramientas; así para emular la topología Fat-Tree se escogió al software mininet [6], pues con la llegada de las redes definidas por software (SDN), es uno de los programas comúnmente utilizado para emular topologías de DC en los últimos años [17]. Para gestionar el enrutamiento, se utiliza el controlador SDN OpenDaylight [18] por la facilidad que brinda en la creación y selección de caminos redundantes por sobre otros controladores. El generador de tráfico D-ITG [2], permitió emular los múltiples flujos de información característica de un DC y mediante el analizador de paquetes Wireshark [22] se capturó el tráfico para la recopilación de resultados. Cada uno de los enlaces fueron ajustados a 5 Mbps y los flujos con paquetes de tamaño constante e igual a 512 Bytes; esto con el fin de que la saturación de los enlaces llegue a ser posible en algún instante.

Para la determinación del DNET experimental, se procedió a generar un flujo con un mayor tamaño de paquete, para facilitar la identificación entre los demás flujos del DC. Para obtener el L_{AV} se generaron flujos entre cada uno de los pares de h hasta completar un número de muestras representativo.

4.2 Análisis de resultados

Luego de que estos invariantes fueron calculados de forma experimental, se obtuvieron los resultados abajo descritos.

Diámetro de la red

Para calcular D_{NET} experimental, se tomaron 100 pares de host al azar y de acuerdo a la traza generada por el f_{ij} , se obtuvieron como máximo número de saltos, los valores de 0, 3 y 5. En la Fig. 4 y en la Fig. 5 se puede apreciar la traza generada por los 10 primeros valores y la estadística del número máximo de saltos para todas las muestras, respectivamente.

C A S O	TRAZA	D N E T
1	$h_1 \rightarrow ESW_{1,1} \rightarrow ASW_{1,2} \rightarrow CSW_3 \rightarrow ASW_{2,2} \rightarrow ESW_{2,1} \rightarrow h_5$	5
2	$h_2 \rightarrow ESW_{1,1} \rightarrow ASW_{1,2} \rightarrow CSW_3 \rightarrow ASW_{3,2} \rightarrow ESW_{3,1} \rightarrow h_{10}$	5
3	$h_4 \rightarrow ESW_{1,2} \rightarrow ASW_{1,1} \rightarrow CSW_2 \rightarrow ASW_{4,1} \rightarrow ESW_{4,1} \rightarrow h_{13}$	5
4	$h_8 \rightarrow ESW_{2,2} \rightarrow ASW_{2,1} \rightarrow CSW_2 \rightarrow ASW_{4,1} \rightarrow ESW_{4,2} \rightarrow h_{15}$	5
5	$h_2 \rightarrow ESW_{1,1} \rightarrow ASW_{1,1} \rightarrow ESW_{1,2} \rightarrow h_4$	3
6	$h_{11} \rightarrow ESW_{3,2} \rightarrow ASW_{3,1} \rightarrow CSW_1 \rightarrow ASW_{4,1} \rightarrow ESW_{4,1} \rightarrow h_{14}$	5
7	$h_{16} \rightarrow ESW_{4,2} \rightarrow ASW_{4,1} \rightarrow CSW_2 \rightarrow ASW_{2,1} \rightarrow ESW_{2,2} \rightarrow h_7$	5
8	$h_5 \rightarrow ESW_{2,1} \rightarrow ASW_{2,1} \rightarrow CSW_1 \rightarrow ASW_{3,1} \rightarrow ESW_{3,1} \rightarrow h_9$	5
9	$h_{10} \rightarrow ESW_{3,1} \rightarrow ASW_{3,1} \rightarrow ESW_{3,2} \rightarrow h_{11}$	3
10	$h_{13} \rightarrow ESW_{4,1} \rightarrow ASW_{4,1} \rightarrow ESW_{4,2} \rightarrow h_{16}$	3

Figura 4. Trazas parciales para D_{NET} .

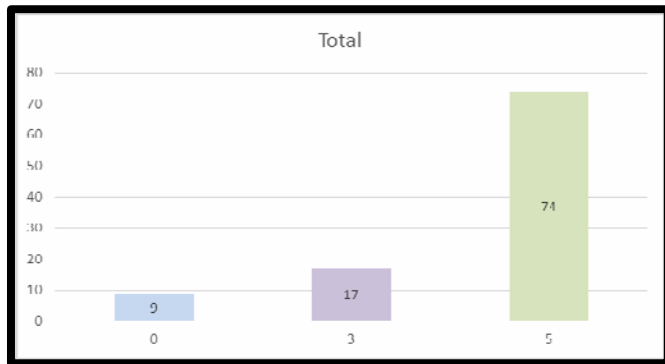


Figura 5. Estadística del máximo número de saltos.

Siendo 5 el valor más alto dentro del número de saltos generados en la topología, el D_{NET} experimental es igual a 5.

Por otro lado para el cálculo del D_{NET} analítico, si se considera la ecuación (1) en la topología emulada con $n=3$, el valor para D_{NET} es igual a:

$$D_{NET} = 2(3) - 1 = 5 \quad (4)$$

Con lo cual el invariante D_{NET} queda comprobado. Cabe indicar que a pesar de que se ha manejado un gran volumen de tráfico, este no ha sido suficiente para saturar los enlaces y llegar a valores mayores y cercanos a un D'_{NET} experimental.

Longitud de ruta promedio

El cálculo del L_{AV} experimental se realizó sobre las 100 muestras anteriormente tomadas. La suma total del número de saltos que utilizaron todos los pares de hosts es igual a 421, cantidad que al dividirla para el número total de muestras (100), resulta en un L_{AV} experimental de 4,21. La interpretación de esto es que en promedio se requieren de 4 saltos para llegar a comunicar un par de hosts cualquiera de la topología.

En cambio, para determinar el L_{AV} analítico en la topología emulada, la ecuación (3) indica que primero se deben sumar el número de saltos de cada uno de los pares de hosts (no repetidos) que componen la topología. Esta suma es igual a 528 y se puede apreciar en la Fig. 6.

HOSTS	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13	H14	H15	H16	Σ
H1		0	3	3	5	5	5	5	5	5	5	5	5	5	5	5	66
H2			3	3	5	5	5	5	5	5	5	5	5	5	5	5	66
H3				0	5	5	5	5	5	5	5	5	5	5	5	5	60
H4					5	5	5	5	5	5	5	5	5	5	5	5	60
H5						0	3	3	5	5	5	5	5	5	5	5	46
H6							3	3	5	5	5	5	5	5	5	5	46
H7								0	5	5	5	5	5	5	5	5	40
H8									5	5	5	5	5	5	5	5	40
H9										0	3	3	5	5	5	5	26
H10											3	3	5	5	5	5	26
H11												0	5	5	5	5	20
H12													5	5	5	5	20
H13														0	3	3	6
H14															3	3	6
H15																0	0
H16																	0
																	528

Figura 6. Número de saltos por cada par de hosts.

Luego este resultado debe ser dividido para el número total de pares considerados en este cálculo, el cual es igual a 120. Por lo tanto el valor analítico para L_{AV} es igual a:

$$L_{AV} = 528 / 120 = 4.4 \quad (5)$$

Lo que implica, al igual que en el valor experimental, de que en promedio también se requiere de 4 saltos para que un par de hosts cualquiera se comuniquen.

5. CONCLUSIONES

En el presente trabajo se ha propuesto el uso de invariantes como método de verificación de modelos concebidos en ambientes simulación/emulación, pues al ser los invariantes una expresión de una red real, permiten establecer la fidelidad del modelo desarrollado para topologías de DC.

Se han propuesto dos invariantes, los cuales han sido demostrados analíticamente, y luego comparados mediante un emulador en condiciones de red similares al de un DC real, dando como resultado respuestas con un buen grado de aproximación.

Este trabajo se ha limitado a utilizar como escenario de un DC, la topología Fat-Tree, sin embargo se puede realizar las mismas verificaciones sobre otras topologías igualmente relevantes, como Bcube, VL2, Dcell, Portland, entre otras. Así mismo se pueden manejar otras distribuciones de probabilidad para la generación del tráfico, así como el uso de tráfico multicast o broadcast, propio de determinados servicios de red.

Es importante como un requisito previo para la determinación de invariantes, el conocimiento profundo de la topología a estudiar, pues de lo contrario los invariantes podrían deducirse incorrectamente.

REFERENCIAS

- [1] Alberto Dainotti, Alessio Botta, Antonio Pescapé, and Giorgio Ventre. 2006. Searching for invariants in network games traffic. In Proceedings

- of the 2006 ACM CoNEXT conference (CoNEXT '06). ACM, New York, NY, USA., Article 43 , 2 pages, 1989.
- [2] Avallone, S., Guadagno, S., Emma, D., Pescapè, A., & Ventre, G. (2004, September). D-ITG distributed internet traffic generator. In Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings. First International Conference on the (pp. 316-317). IEEE.
- [3] Bari, M.F.; Boutaba, R.; Esteves, R.; Granville, L.Z.; Podlesny, M.; Rabbani, M.G.; Qi Zhang; Zhani, M.F., "Data Center Network Virtualization: A Survey," Communications Surveys & Tutorials, IEEE , vol.15, no.2, pp.909,928, Second Quarter 2013
- [4] Bengt Ahlgren, Marcus Brunner, Lars Eggert, Robert Hancock, and Stefan Schmid. 2004. Invariants: a new design methodology for network architectures. In Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture (FDNA '04). ACM, New York, NY, USA, 65-70.
- [5] Benson, T., Akella, A., & Maltz, D. A. (2010, November). Network traffic characteristics of data centers in the wild. In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (pp. 267-280). ACM.
- [6] Bob Lantz, Brandon Heller, and Nick McKeown. 2010. A network in a laptop: rapid prototyping for software-defined networks. In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks (Hotnets-IX). ACM, New York, NY, USA, , Article 19 , 6 pages.
- [7] Breslau, L., Estrin, D., Fall, K., Floyd, S., Heidemann, J., Helmy, A., ... & Yu, H. (2000). Advances in network simulation. *Computer*, 33(5), 59-67.
- [8] Chen, Z., Peng, L., Zhao, S., Zhang, L., & Jing, S. (2014). Feature Selection Toward Optimizing Internet Traffic Behavior Identification. In *Algorithms and Architectures for Parallel Processing* (pp. 631-644). Springer International Publishing.
- [9] Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., ... & Sengupta, S. (2009, August). VL2: a scalable and flexible data center network. In ACM SIGCOMM Computer Communication Review (Vol. 39, No. 4, pp. 51-62). ACM.
- [10] Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., ... & Lu, S. (2009). BCube: a high performance, server-centric network architecture for modular data centers. ACM SIGCOMM Computer Communication Review, 39(4), 63-74.
- [11] Kurup, P. M., & Preethi, J. (2014). Packet Reordering To Improve Data Center Network Using Near Optimal Traffic Engineering.
- [12] Lee, David S., and Jeffrey L. Kalb. "Network Topology Analysis."
- [13] Luan, G. (2014). Buffer Stopping Time Analysis in Data Center Networks.
- [14] Martin F. Arlitt and Carey L. Williamson. 1996. Web server workload characterization: the search for invariants. In Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling of computer systems (SIGMETRICS '96), Blaine D. Gaither (Ed.). ACM, New York, NY, USA, 126-137
- [15] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. 2008. A scalable, commodity data center network architecture. In Proceedings of the ACM SIGCOMM 2008 conference on Data communication (SIGCOMM '08). ACM, New York, NY, USA, 63-74.
- [16] Niranjana Mysore, R., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., ... & Vahdat, A. (2009, August). Portland: a scalable fault-tolerant layer 2 data center network fabric. In *ACM SIGCOMM Computer Communication Review* (Vol. 39, No. 4, pp. 39-50). ACM.
- [17] Nunes, B., Mendonca, M., Nguyen, X., Obraczka, K., & Turletti, T. (2014). A survey of software-defined networking: Past, present, and future of programmable networks.
- [18] Orebaugh, A., Ramirez, G., & Beale, J. (2006). Wireshark & Ethereal network protocol analyzer toolkit. Syngress.
- [19] Park, H. W., Yeo, I. Y., Lee, J. R., & Jang, H. (2014). Study on network architecture of big data center for the efficient control of huge data traffic. *Computer Science and Information Systems*, (00), 67-67.
- [20] Sally Floyd and Vern Paxson. 2001. Difficulties in simulating the internet. *IEEE/ACM Trans. Netw.* 9, 4 (August 2001), 392-403.
- [21] W. Dally and B. Towles, Principles and Practices of Interconnection Networks. Morgan Kaufmann Publishers Inc., 2004.
- [22] WRIGHT, C., MESTERY, K., SHAIKH, A., & BAUCKE, S. OpenDaylight: An Open Source SDN for Your OpenStack Cloud. OpenStack Summit Presentations.

