JITEL 2017
XIII Jornadas de Ingeniería Telemática
27-29 Septiembre 2017, Valencia (España)

Actas de las XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia (España), 27-29 de Septiembre de 2017.

ISBN: 978-84-9048-595-8
DOI: http://dx.doi.org/10.4995/JITEL2017.2017.6502

Definición de Testbeds Virtualizados Utilizando Perfiles de Actividad de Red

David Muelas, Javier Ramos, Jorge E. López de Vergara
HPCN, Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior, Universidad Autónoma de Madrid
Francisco Tomás y Valiente, 11, 28049 Madrid, España
Email: {dav.muelas, javier.ramos, jorge.lopez_vergara}@uam.es

***Resumen*—Un problema recurrente para los profesionales de la Ingeniería Telemática es la escasez de despliegues de tecnologías emergentes y las restricciones de acceso a redes operativas. Por ello, en este trabajo presentamos un método para la generación automática de carga siguiendo perfiles de actividad que facilita la replicación del comportamiento típico de una red. Este método se basa en un nodo de control que configura agentes de generación de tráfico, para aprovechar las capacidades de las plataformas de virtualización de red. Evaluamos esta propuesta en un caso de estudio que considera un despliegue de Voz sobre IP (*Voice over IP*, VoIP) en un servidor de propósito general, usando Mininet como entorno de virtualización ligera. Los resultados muestran que el método propuesto replica fidedignamente la dinámica de red especificada, y que los recursos físicos consumidos permiten su uso en equipamiento de coste reducido.**

***Palabras Clave*—redes virtualizadas, redes definidas por software, experimentación, testbeds, evaluación de prestaciones, VoIP, Mininet.**

I. INTRODUCCIÓN

Los avances en las tecnologías de telecomunicaciones han permitido una evolución muy rápida de los elementos y comportamientos presentes en las redes actuales [1]. Como consecuencia, las herramientas clásicas de simulación, como OMNeT++ o ns-3, no son particularmente aptas para la innovación en entornos emergentes, como los orientados a la Internet de las Cosas (*Internet of Things*, IoT) o a las Redes Definidas por Software (*Software Defined Networks*, SDN) [2]. Por ello, desde el punto de vista de la evaluación de nuevos protocolos, metodologías y elementos de red, se hace necesario desarrollar nuevos enfoques que se adapten a estos entornos reales.

Las oportunidades que ofrece la Virtualización de Funciones de Red (*Network Function Virtualization*, NFV) [3], [4] ha atraído la atención de la comunidad dedicada a la investigación en Ingeniería Telemática por la flexibilidad que ofrecen los despliegues de red virtualizados. No obstante, pese a la profusión de soluciones para encarar diversos retos de la gestión y operación de redes de comunicaciones, su aplicación a la definición de *testbeds* realistas y de bajo coste no ha sido particularmente amplia. Este tipo de plataformas resultan totalmente necesarias a la hora de garantizar tanto la repetibilidad de los experimentos como la disponibilidad de infraestructura de pruebas para la investigación de entornos emergentes, tales como las redes inalámbricas de sensores [5].

Además, el acceso a entornos de red operativos es en muchos casos muy restringido, debido a la importancia que tienen estas infraestructuras. Esto dificulta la evaluación de nuevas soluciones con comportamientos que representen la realidad, de forma que muchas veces la detección de problemas que no aparecen en evaluaciones sintéticas es inviable —por ejemplo, los errores de marcado de tiempo reportados en [6] que afectan a motores de captura de altas prestaciones aparecen cuando la tasa del tráfico disminuye.

Por todo ello, en este trabajo exploramos la definición por software de *testbeds* virtualizados con el fin de facilitar la experimentación en entornos emergentes de red. Para ello, definimos un método para automatizar la acción de agentes dentro de una red de comunicaciones, replicando una dinámica de red determinada. Además, evaluamos Mininet [7], [8] como entorno de virtualización ligera, proporcionando una caracterización de métricas de rendimiento, lo cual define los escenarios y situaciones donde esta herramienta puede ser usada. Nuestro trabajo sigue la filosofía con la que fue desarrollada Mininet, que surgió como una herramienta que facilitara la experimentación en entornos emergentes de red [9], [10], [11].

Para responder a la cuestión de si es posible emular despliegues que representen un amplio espectro de escenario reales en condiciones controladas y con un coste reducido, en este trabajo *(i)* presentamos una metodología y arquitectura para la generación automática de carga de red siguiendo perfiles de actividad, para facilitar la definición de *testbeds* virtualizados; *(ii)* describimos la

implementación de esta arquitectura en Mininet, para reducir el coste de aplicación y maximizar su versatilidad; y *(iii)* comprobamos la viabilidad de la solución tanto en términos del consumo de recursos físicos en diversas topologías con baja y alta actividad de red, como de las características de la carga generada.

El resto del artículo se estructura del siguiente modo. La Sección II recopila diversos trabajos relacionados con el nuestro, motivando las características y decisiones de diseño de nuestra propuesta. Posteriormente, en la Sección III se describe nuestro método de generación de carga, primero en su versión más general y luego en un caso particular aplicado al estudio de despliegues de Voz sobre IP (*Voice over IP*, VoIP). En la Sección IV se incluyen los resultados obtenidos en un entorno virtualizado con Mininet, mostrando la viabilidad de este enfoque para la definición de *testbeds*. Finalmente, la Sección V recopila las principales conclusiones de este trabajo, y plantea las líneas de trabajo futuro que estamos empezando a explorar.

II. TRABAJO RELACIONADO

En esta sección presentamos una selección de resultados previos que motivan nuestro trabajo. Todos ellos ilustran la necesidad de mejorar la definición de *testbeds* virtualizados, por la importancia que tienen durante la evaluación de nuevas herramientas y métodos en un amplio abanico de entornos de red emergentes.

En [12], [13], [14] se presentan metodologías para la generación de tráfico sintético según los parámetros extraídos de tráfico real. Particularmente, los autores de [12] definen un método para extraer las características del tráfico de red, y generar conexiones que presentan las mismas características estadísticas. Por su parte, en [13], se ofrece una revisión más exhaustiva de la literatura relativa a generadores de carga de red, motivando las decisiones de diseño de una arquitectura que comparte ciertos rasgos con la solución que presentamos en este trabajo, aunque se restringe a la generación de flujos de datos realistas. Más recientemente, [14] incluye un análisis más próximo al nuestro, analizando las características de flujos de datos procedentes de fuentes específicas.

La evaluación de estas soluciones permite comprobar que las características del tráfico sintético son indistinguibles estadísticamente hablando de las del tráfico original. No obstante, nuestra propuesta se centra en replicar la dinámica de la actividad de la red en términos de conexiones activas, con el fin de definir comportamientos complejos. Este aspecto facilita la evaluación de nuevos protocolos y herramientas, y además permite que el tráfico sintético pueda ser generado en base a la actividad observada de distintas aplicaciones reales. Por ello, nuestra propuesta se podría utilizar para extender esos trabajos, con el fin de enriquecer el comportamiento dinámico de la carga generada.

Por su parte, en [15], [16] se discute la adecuación de la virtualización de elementos de red para la evaluación de aplicaciones multimedia. Estos trabajos se pueden ver como un antecedente directo de nuestra solución, aunque las características tecnológicas de las soluciones planteadas en ellos están lejos de los últimos desarrollos de virtualización de redes. Estos factores constriñen, por tanto, la aplicabilidad del sistema propuesto en la actualidad, aunque es un trabajo que motiva los desarrollos en la línea de la solución que proponemos.

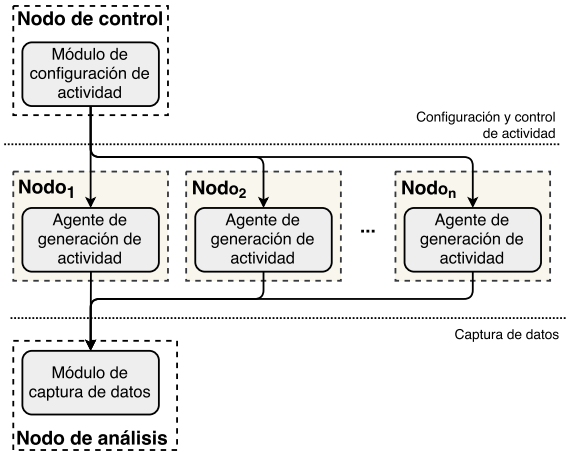


Fig. 1. Esquema de la arquitectura propuesta.

El uso de Mininet como plataforma de emulación de redes para validación y experimentación es una de las motivaciones iniciales de su desarrollo. Experiencias previas, como la recogida en [17], ya muestran su idoneidad para el estudio de redes en operación, siempre y cuando se acepte cierta pérdida de precisión en los resultados obtenidos [1], [18]. Por otro lado, trabajos como [10], [11] muestran la utilidad de esta herramienta a la hora de experimentar sobre entornos emergentes de red. Todos estos resultados previos motivan la exploración de las posibilidades que ofrecen los *testbeds* virtualizados para facilitar el acceso a entornos controlados y que representen de manera fidedigna los futuros despliegues de red.

III. GENERACIÓN DE CARGA A PARTIR LA DINÁMICA DE LA ACTIVIDAD

La metodología que proponemos para generar de forma automática carga de red realista se basa en el uso de agentes que emulen la actividad de usuarios que son controlados para replicar el comportamiento agregado de una red. La carga generada consiste en paquetes de red con el fin de que la solución sea útil en el mayor número de casos posible.

A. *Definición general del método*

La Fig. 1 representa la arquitectura de nuestra solución, formada por un nodo de control que configura y que activa agentes de generación de actividad (tráfico) en el resto de nodos. Finalmente, para observar el comportamiento global agregado, se incluye otro nodo que captura y analiza el tráfico generado. Este diseño permite que asumamos sin pérdida de generalidad que tanto la configuración como la generación de actividad se refieren a únicamente una aplicación. En otro caso, se puede replicar la arquitectura propuesta por cada aplicación que se quiera incluir en el *testbed* y agregar posteriormente el tráfico resultante.

Para caracterizar el comportamiento de la dinámica de red, es necesario que exista un patrón más o menos estable de la actividad de red. De hecho, cambios sostenidos en estos valores pueden ser indicativos de cambios de uso en la red [19], lo que entra en contradicción con la caracterización del comportamiento dinámico de la red. Por ello, para aplicar nuestro método requerimos que las siguientes hipótesis se cumplan:

- Existe una línea base para la evolución temporal típica del número de conexiones activas en la red [20].
- Existe un proceso característico para la aparición de nuevas conexiones por unidad de tiempo que se puede ajustar utilizando su esperanza.
- La distribución de la duración de la conexión no cambia con el tiempo.

A partir de la primera hipótesis, se sigue que es posible definir una línea base de alta dimensionalidad, basada en una función $F(t), t \in \mathbb{T}$, con $\mathbb{T} \subset \mathbb{R}$ un compacto correspondiente al dominio temporal de la dinámica. Esta línea base se puede definir a partir del análisis de la dinámica de una una red real [20] con el fin de replicar su comportamiento; o para acomodarse a una situación controlada como en el caso de estudio propuesto.

Por otro lado, como en este caso se están modelando las conexiones activas tal y como las vería un elemento de red que recibiese todo el tráfico, en este sistema no aparece ningún efecto de encolado —y por lo tanto, la duración de las conexiones y el tiempo que están activas coincide. Además, de la tercera hipótesis se sigue que la esperanza de la duración de las conexiones (W) debe ser constante.

A partir de $F(t)$ y W, se quiere definir una aproximación de la esperanza del proceso de nuevas conexiones para poder modular la actividad del *testbed* y ajustarla a la dinámica esperada. Siguiendo la demostración de la Ley de Little [21], utilizamos una descomposición del compacto $\mathbb{T}$ en una sucesión de compactos $\{\mathbb{T}_i\}$ tales que su unión es $\mathbb{T}$ y que son disjuntos dos a dos. Ahora, como W es constante es posible definir el número esperado de nuevas conexiones en base a la expresión de la Ec. 1:

$$\lambda(t) = \frac{\sum_{t \in \mathbb{T}_i} F(t)}{W}, \ \forall t \in \mathbb{T}_i \tag{1}$$

Por su parte, el proceso de nuevas conexiones $A(t), t \in \mathbb{T}$ se ajusta de modo que se cumpla la Ec. 2:

$$\mathbb{E}[A(t)] = \lambda(t), \forall t \in \mathbb{T} \tag{2}$$

Finalmente, el nodo de control configura un número aleatorio de nuevas conexiones entre los *hosts* presentes en la topología virtual, generado según el proceso ajustado en cada unidad de tiempo. Después, las nuevas conexiones son activadas, establecidas y mantenidas de forma autónoma por los nodos de generación de carga.

B. Adaptación para generación de tráfico de VoIP

Para adaptar el método general previo al caso de generación de tráfico de VoIP, vamos a considerar el modelo clásico de telefonía en el que el número de nuevas conexiones sigue una distribución de Poisson, de modo que el número de nuevas conexiones $A(t)$ en cada instante t cumple la Ec. 3:

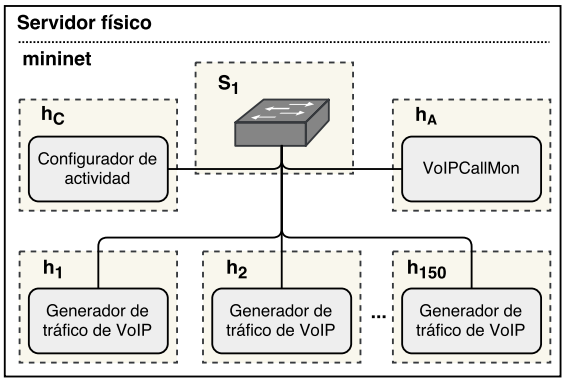


Fig. 2. Topología virtual definida para la evaluación del método de generación de carga.

$$A(t) \sim \text{Poi}(\lambda(t)), t \in \mathbb{T} \tag{3}$$

y la duración de cada conexión k sigue una distribución exponencial, de modo que se cumple la Ec. 4:

$$Dur(\mathcal{C}) \sim \text{Exp}(1/W(t)), t \in \mathbb{T} \tag{4}$$

para todas las conexiones $\mathcal{C}$ iniciadas en el instante t.

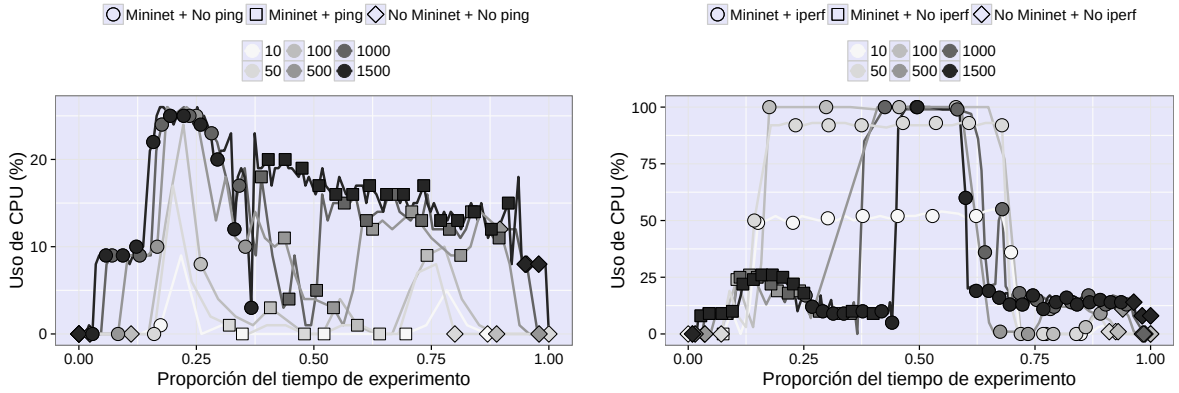
En lo referente a las conexiones, deben generarse dos flujos de datos por cada sentido de una llamada —uno correspondiente a la señalización y otro para los datos multimedia. La combinación de protocolos de señalización y transporte multimedia ha sido seleccionada para representar entornos habituales tanto en redes empresariales como domésticas. En particular, los protocolos de señalización utilizados son el Protocolo de Inicio de Sesión (*Session Initiation Protocol*, SIP) y el Protocolo de Control de Llamada Skinny (*Skinny Call Control Protocol*, SCCP). El protocolo para transporte de datos multimedia es el Protocolo de Transporte de Tiempo Real (*Real-time Transport Protocol*, RTP), utilizando la definición de carga correspondiente al códec G.711.

IV. RESULTADOS EXPERIMENTALES

A. Definición de los experimentos

A continuación se ofrece una descripción completa del *hardware* y la topología virtual considerada durante la realización de nuestros experimentos. Asimismo, el software empleado para la generación de carga se encuentra disponible bajo petición para cualquier persona interesada en su uso o modificación.

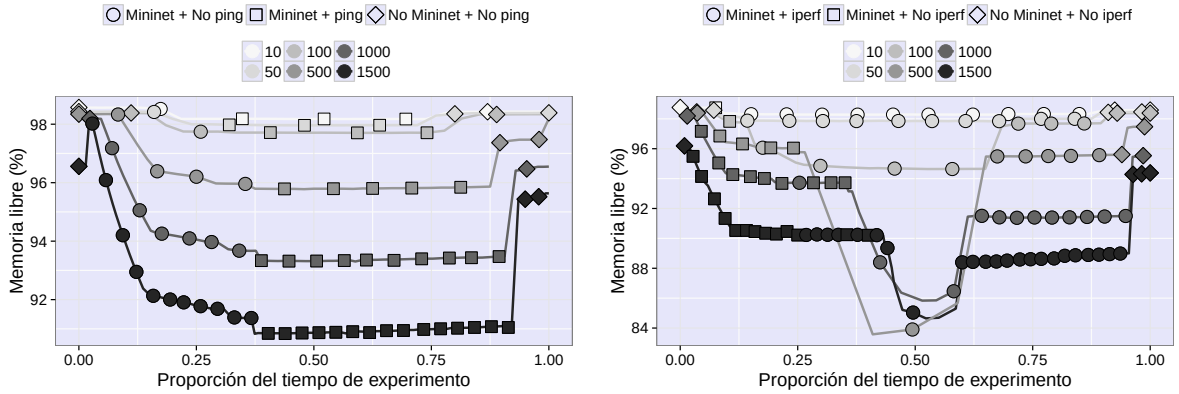
Todos los experimentos han sido ejecutados en un servidor de propósito general equipado con dos procesadores Intel Xeon E5-2620 (6 *cores* por procesador) con una frecuencia de 2.10 GHz y 32 GB de memoria RAM. Para evitar efectos no controlados producidos por la virtualización de *hardware*, las características de *Hyper-Threading* han sido desactivadas. El sistema operativo de este servidor se corresponde con una distribución Ubuntu 14.04.1 instalada con un *kernel* de Linux versión 4.4.0-45. Las pruebas se han ejecutado usando la versión 2.2.2



(a) Entorno de baja carga de red

(b) Entorno de alta carga de red

Fig. 3. Uso de CPU en base al número de nodos de generación de carga, 10 *switches*.



(a) Entorno de baja carga de red

(b) Entorno de alta carga de red

Fig. 4. Uso de memoria en base al número de nodos de generación de carga, 10 *switches*.

de Mininet descargada desde el repositorio Git de la herramienta[1] y utilizando la configuración por defecto.

Mininet utiliza *namespaces* de red para virtualizar los distintos elementos que conforman una topología —*hosts*, *switches*, etc.. En nuestro caso consideraremos Open vSwitch (OvS) como elemento de interconexión de todos los *hosts* presentes en las topologías propuestas. Por ello, como primer conjunto de resultados, analizamos la ocupación de recursos físicos del servidor a medida que se incrementa el número de *hosts* con una topología lineal de 10 *switches* interconectándolos. Con ello, caracterizamos el comportamiento del *testbed* en términos de recursos, asegurando que este escenario no presenta cuellos de botella o limitaciones que impidan la generación de carga.

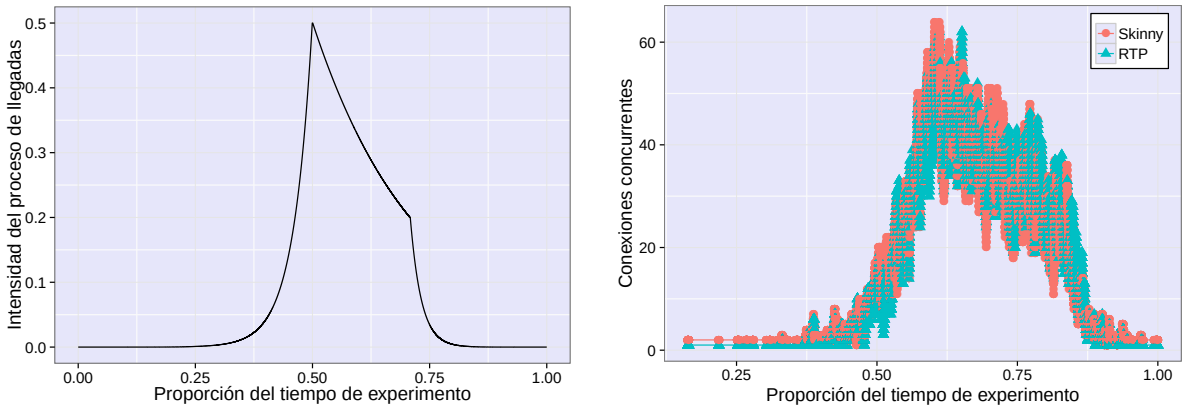
Por otro lado, la topología definida con Mininet correspondiente a las pruebas de generación de carga se muestra en la Fig. 2, indicando los enlaces (sin ningún tipo de limitación) establecidos entre los distintos elementos de red implicados. Con el fin de simular una Red de Área Local (*Local Area Network*, LAN) de terminales de telefonía, se utilizan 150 *hosts* de Mininet $\{h_i\}, i = 1 \ldots 150$, controlados por un *host* que orquesta el número de conexiones establecidas, h_C, y que se interconectan a través de un único *switch* virtual, s_1. Para el análisis de tráfico, se introduce otro *host* h_A que recibe todo el tráfico, que ejecuta una instancia de la herramienta de análisis de tráfico VoIPCallMon [22] —de hecho, el *testbed* se utilizó para evaluar el comportamiento de la herramienta antes de su despliegue en un entorno operativo.

[1] git://github.com/mininet/mininet

B. Comportamiento del entorno de virtualización

En primer lugar, caracterizamos el uso de recursos físicos que supone utilizar Mininet, para obtener evidencias de su viabilidad como plataforma de *testbeds* virtualizados. La Fig. 3 incluye los resultados de consumo de CPU, mientras que la Fig. 4 refleja el consumo de memoria. Ambos recursos son monitorizados en entornos de baja y alta carga de red (definidos usando `ping` e `iperf`, respectivamente), y variando el número de *hosts* activos en la topología emulada desde 10 hasta 1500. De esta forma, barremos todas las posibles situaciones de interés para el caso de estudio que nos proponemos.

En el caso del uso de CPU, se observa que durante la creación de la topología virtual ((Mininet + No ping) no se consume más del 25% de la CPU del equipo. La mayor parte de este consumo se debe a la creación e interconexión de los *switches*. Cuando se usa `ping`, la carga de CPU disminuye substancialmente, ya que la carga de red que genera es un muy limitada —1 paquete ICMP cada segundo. En el caso de `iperf`, la carga de CPU crece hasta el 100% en algunos casos como consecuencia de las elevadas tasas de transmisión y recepción (∼10



(a) Intensidad del modelo de nuevas conexiones.

(b) Dinámica de conexiones de señalización y tráfico multimedia.

Fig. 5. Caracterización de la carga generada por el sistema.

Gb/s). Además, el efecto de incorporar nuevos *hosts* es más notable que en el caso anterior, por el incremento asociado del número de instancias de `iperf` en ejecución.

Los resultados de consumo de memoria muestran resultados similares. Durante la creación de la topología de Mininet se observa que, a mayor número de *hosts*, mayor consumo de memoria. No obstante, el consumo de memoria no resulta excesivo, ya que para 1500 *hosts* la memoria utilizada no representa más del 6% del total disponible. Analizando las diferencias entre los entornos de baja y alta carga, se observa que utilizando `ping` apenas se incrementa el uso de memoria al lanzar los procesos una vez ha sido creada la topología. Por el contrario, cuando se crean los procesos de `iperf`, se incrementa sustancialmente el consumo de memoria, suponiendo hasta un consumo extra del 12% en el peor de los casos.

Las conclusiones que se extraen de estos experimentos son: (i) que la carga de CPU es altamente dependiente del tipo de proceso de red que ejecutemos en los *hosts* de Mininet y (ii) que el consumo de memoria depende en gran medida del número de *hosts* usado en el escenario. Además, podemos observar que, en términos de memoria, emular una red compleja con 1500 *hosts* y 10 *switches* no consume más de un 20% de la memoria de la máquina.

C. Carga generada

Tras contar con evidencia suficiente de la viabilidad de emulación del despliegue, estudiamos el comportamiento de una red de teléfonos con soporte para VoIP emulada, incluyendo 150 terminales. En nuestro caso, el objetivo es estudiar la funcionalidad y estabilidad de VoIPCallMon a la hora de monitorizar una red local de VoIP. Por ello, fijamos como requisito del *testbed* que se tenga una concurrencia de 60 llamadas en el período de máxima actividad y que muestre la dinámica de actividad típica de una red empresarial, marcada por actividad en horario laboral y hora más cargada alrededor de las 12 del mediodía.

Para ello, fundamentándonos en experiencias previas de monitorización, prefijamos un perfil de concurrencia diario basado en cuatro puntos temporales que lo parametrizan:

- *Hora de comienzo de actividad:* 6 de la mañana.
- *Hora de finalización de actividad:* 8 de la tarde.
- *Hora de máxima actividad (H1):* 12 del mediodía.
- *Transición de actividad de media tarde a cierre (H2):* 5 de la tarde.

y dos adicionales que nos dan la concurrencia en H1 y H2. Posteriormente, definimos una función a trozos que sigue ese perfil, y la transformamos para obtener el patrón de nuevas conexiones (llamadas) según muestra la Fig. 5(a).

Utilizando el método descrito en la Sección III, generamos actividad que replique este perfil para un día de actividad, y analizamos el tráfico en el nodo que recibe el agregado ejecutando VoIPCallMon. La generación de tráfico se ha acometido utilizando la librería de Python Scapy, por su versatilidad a la hora de conformar paquetes de tráfico. La utilización de esta librería se fundamenta en que SCCP es un protocolo propio de teléfonos de VoIP de Cisco. Esto ocasiona que sea la alternativa más simple para poder evaluar un sistema de monitorización con soporte para estos terminales, si no se tiene acceso a un despliegue real. Por otro lado, y tal y como se ha mencionado anteriormente, las llamadas generadas utilizan G.711 como códec para el audio transmitiendo un paquete por cada 20ms de muestras. En base a medidas empíricas, se ha seleccionado 100s como duración media, ya que este valor representa lo esperable en entornos de oficina.

Las series temporales de conexiones activas detectadas por esta herramienta se muestra en la Fig. 5(b), mostrando el buen ajuste a los requisitos para la dinámica de evaluación. Además, al simular distintas trayectorias sobre el dominio $\mathbb{T}$, se observa que la actividad en instantes equivalentes en distintas realizaciones de la dinámica presenta un comportamiento estable adaptado a los parámetros antes indicados —la Fig. 6 ilustra esta idea sobre 30s del período de más actividad de 40 trayectorias, mostrando gráficos de cajas para cada día y la función de medias. La variación de la función de medias viene dada por la varianza del proceso de generación de nuevas conexiones —igual a la media, por seguir una distribución de Poisson.

V. CONCLUSIONES Y TRABAJO FUTURO

Este trabajo presenta evidencias de la viabilidad de la definición de *testbeds* para entornos emergentes de red usando Mininet como plataforma de virtualización ligera.

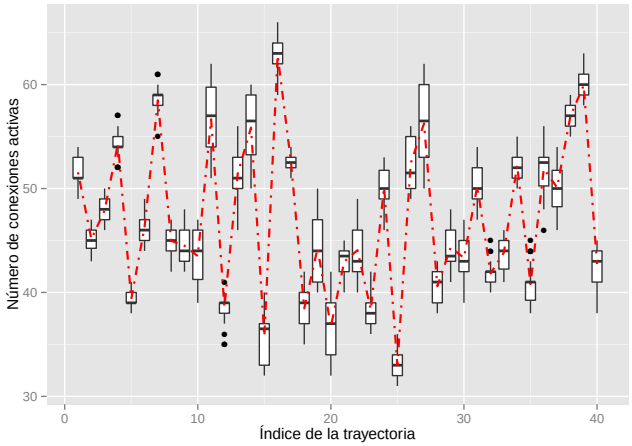


Fig. 6. Evolución de las conexiones en 30s del período de máxima actividad, 40 trayectorias. Cada diagrama de caja muestra los valores para un mismo día, y la línea roja discontinua la función de medias.

Hemos caracterizado Mininet en términos de consumo físico de recursos, analizando la carga de CPU y uso de memoria en entornos de baja y alta actividad de red. Nuestro resultados indican que es posible desplegar topologías con más de 1000 elementos de red en servidores de propósito general, siendo la carga de CPU muy dependiente de la actividad de los nodos.Con estos resultados, comprobamos la viabilidad del uso de esta plataforma para un caso de estudio que ilustra un método de generación de carga capaz de replicar perfiles de actividad no estacionarios. Este caso de estudio plantea la emulación de un despliegue empresarial de VoIP, mostrando que nuestro método genera tráfico según el perfil prefijado.

Como trabajo futuro, planteamos la extensión de la metodología propuesta a otro tipo de tráfico para facilitar, por ejemplo, la evaluación de mecanismos de transmisión de vídeo desde dispositivos empotrados, el funcionamiento de redes de sensores, o la introducción de métodos de validación de parámetros de calidad. Asimismo, estamos estudiando el comportamiento de esta plataforma introduciendo penalizaciones en los enlaces desplegados, mediante el uso del comando `tc` de Linux; y las posibilidades que ofrecen OvS y el uso de OpenFlow.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y Competitividad y el Fondo Europeo de Desarrollo Regional a través de los proyectos TRÁFICA (MINECO / FEDER TEC2015-69417-C2-1-R) y RACING DRONES (MINECO / FEDER RTC-2016-4744-7).

REFERENCIAS

[1] D. Pediaditakis, C. Rotsos, and A. W. Moore, "Faithful Reproduction of Network Experiments," in *Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ser. ANCS '14, 2014, pp. 41–52.

[2] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 27–51, 2015.

[3] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[4] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-Art and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 236–262, 2016.

[5] J. Horneber and A. Hergenröder, "A Survey on Testbeds and Experimentation Environments for Wireless Sensor Networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 1820–1838, 2014.

[6] V. Moreno, P. M. S. del Río, J. Ramos, J. J. Garnica, and J. L. García-Dorado, "Batch to the Future: Analyzing Timestamp Accuracy of High-Performance Packet I/O Engines," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1888–1891, 2012.

[7] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for Software-Defined Networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010, pp. 19:1–19:6.

[8] J. Yan and D. Jin, "VT-Mininet: Virtual-time-enabled Mininet for Scalable and Accurate Software-Define Network Emulation," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, ser. SOSR '15, 2015, pp. 27:1–27:7.

[9] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, "Reproducible Network Experiments Using Container-based Emulation," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '12, 2012, pp. 253–264.

[10] B. Lantz and B. O'Connor, "A Mininet-based Virtual Testbed for Distributed SDN Development," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 365–366, 2015.

[11] L. Baldesi and L. Maccari, "NePA TesT: network protocol and application testing toolchain for community networks," in *2016 12th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, 2016, pp. 1–8.

[12] M. C. Weigle, P. Adurthi, F. Hernández-Campos, K. Jeffay, and F. D. Smith, "Tmix: A Tool for Generating Realistic TCP Application Workloads in Ns-2," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 3, pp. 65–76, 2006.

[13] A. Botta, A. Dainotti, and A. Pescapé, "A tool for the generation of realistic network workload for emerging networking scenarios," *Computer Networks*, vol. 56, no. 15, pp. 3531 – 3547, 2012.

[14] P. Rygielski, V. Simko, F. Sittner, D. Aschenbrenner, S. Kounev, and K. Schilling, "Automated Extraction of Network Traffic Models Suitable for Performance Simulation," in *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*, ser. ICPE '16, 2016, pp. 27–35.

[15] C. Bachmeir, P. Tabery, S. Uzumcu, and E. Steinbach, "A scalable virtual programmable real-time testbed for rapid multimedia service creation and evaluation," in *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on*, vol. 3, 2003, pp. III–257–60 vol.3.

[16] W. Fuertes and J. E. López de Vergara, "An emulation of vod services using virtual network environments," *Electronic Communications of the EASST*, vol. 17, 2009.

[17] M. Raza, S. Chowdhury, and W. Robertson, "SDN based emulation of an academic networking testbed," in *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2016, pp. 1–6.

[18] J. M. Jimenez, J. O. R. Martínez, A. Rego, A. Dilendra, and J. Lloret, "Study of multimedia delivery over software defined networks," in *Network Protocols and Algorithms*, vol. 7, no. 4. Macrothink Institute, 2015, pp. 37–62.

[19] F. Mata, J. L. García-Dorado, and J. Aracil, "Detection of traffic changes in large-scale backbone networks: The case of the Spanish academic network," *Computer Networks*, vol. 56, no. 2, pp. 686 – 702, 2012.

[20] D. Muelas, J. E. López de Vergara, J. R. Berrendero, J. Ramos, and J. Aracil, "Facing Network Management Challenges with Functional Data Analysis: Techniques & Opportunities," *Mobile Networks and Applications*, pp. 1–13, 2016.

[21] J. D. Little, "Little's Law as Viewed on Its 50th Anniversary," *Operations Research*, vol. 59, no. 3, pp. 536–549, 2011.

[22] J. L. García-Dorado, P. M. Santiago del Río, J. Ramos, D. Muelas, V. Moreno, J. E. López de Vergara, and J. Aracil, "Low-cost and high-performance: VoIP monitoring and full-data retention at multi-Gb/s rates using commodity hardware," *International Journal of Network Management*, vol. 24, no. 3, pp. 181–199, 2014.