ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Consideraciones al momento de evaluar la migración del simulador ns-2 al ns-3

Considerations when evaluating the migration from simulator ns-2 to ns-3

María Luz Almada, Carlos Alberto Talay
mluzalmada@gmail.com, ctalay@uarg.unpa.edu.ar
Instituto de Tecnología Aplicada - Unidad Académica Río Gallegos
Universidad Nacional de la Patagonia Austral
Av. Gregores y Piloto "Lero" Rivera - Río Gallegos - Santa Cruz - Argentina

Recibido: 30/04/2020. Aceptado: 11/09/2020

RESUMEN

En el mundo tecnológico actual, donde las redes están creciendo rápidamente, se exige que los protocolos de red sean robustos, confiables y versátiles. Es por ello que antes de ser propuestos para su uso, deben ser estudiados a fondo, para lo que se requiere utilizar herramientas informáticas que permitan examinar su comportamiento en diferentes entornos. Es así que, con este fin, se utilizan simuladores de redes. Uno de estos simuladores es el ns-2, que ha sido utilizado de manera exhaustiva en ámbitos académicos y científicos por casi dos décadas. Sin embargo, con el lanzamiento de un nuevo simulador, el ns-3, que difiere en muchos aspectos de su antecesor y requiere la reformulación de los programas escritos para la versión anterior, surge el interrogante si la actualización a las nuevas versiones es realmente necesaria.
Es así que, nos preguntamos si el simulador de redes que utilizamos es suficiente para nuestro trabajo o si la nueva versión aportará considerables beneficios en las investigaciones actuales y futuras. Determinar si la inversión de tiempo en el aprendizaje de una nueva herramienta, producirá suficientes mejorías en las tareas que requiere la investigación de redes, es una decisión importante.
Este informe presenta de manera comparativa las dos versiones del simulador. Se describen las ventajas y desventajas, alcances, limitaciones y requisitos para la utilización de cada una de las versiones.

Palabras clave: ns-2; ns-3; Simulación de redes; Herramientas de simulación; Tráfico de red.

ABSTRACT

In today's technological world, where networks are growing rapidly, network protocols are required to be robust, reliable, and evaluated. The protocols must be thoroughly studied, and for this is necessary to use computer tools that allow examining their behavior in different environments. The ns-2 simulator has been extensively used in academic and scientific fields for almost two decades. However, with the launch of a new simulator, the ns-3, which differs in many respects from its predecessor and requires the reformulation of programs written for the previous version, the question arises whether updating to new versions is necessary.
We often ask ourselves if the network simulator we use is sufficient for our work, or if the new version will bring considerable benefits in current and future research. Determining whether spending time learning a new tool will produce enough improvement in the tasks that network research requires is an important decision.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

This report presents the two versions of the simulator comparatively. The advantages and disadvantages, scope, limitations and requirements for the use of each of the versions are described.

Key words: ns-2; ns-3; Network simulation; Simulation tools; Network traffic.

1. INTRODUCCIÓN

En el área de las telecomunicaciones y las redes de datos, poder medir los diferentes parámetros que componen una red resulta de mucha importancia para mejorar su desempeño. Realizar estas mediciones en un ambiente real, además de espacio, tiempo y dinero, resulta imposible llevar a la práctica dado que son muchos los factores que el investigador no podrá controlar. Por este motivo, en los ambientes educativos y de investigación, se recurre al diseño de modelos que simulen el ambiente del mundo real y permitan controlar la mayor cantidad de variables que modificarán los resultados.

En este sentido, se ha utilizado ampliamente el simulador ns-2 para implementar los modelos diseñados y se han hecho muchas modificaciones al mismo desde que comenzó a desarrollarse la primera versión del simulador en 1989. Y aunque el soporte oficial de esta herramienta haya cesado en 2011, luego del lanzamiento de la versión 2.35 (USC/ISI, 2011), es un simulador aún muy utilizado.

Uno de los objetivos del grupo de investigación fue poder formalizar la migración en el uso de la herramienta, de su versión ns-2 a ns-3. Ambas herramientas son incompatibles entre sí, dado que la última versión no es una actualización o extensión de la primera, sino que fue un desarrollo desde cero basado en otro paquete y con grandes diferencias respecto a su antecesora. Es un nuevo simulador y las diferencias se advierten desde el nivel físico y el lenguaje utilizado, en las formas de implementación y modo de uso, y principalmente en el hecho que todos los desarrollos alcanzados con ns-2 deben ser reescritos para que puedan funcionar de manera similar en ns-3, ya que no admite las mismas API que ns-2.

Por estos motivos, el propósito de este trabajo es analizar las prestaciones de cada una de las versiones, las facilidades y dificultades que presenta el cambio planteado, y sentar los puntos fundamentales a tener en cuenta para cualquiera que utilice ns-2 actualmente y desee hacer la migración para el uso del nuevo simulador.

Utilizar ns-2 requiere conocimientos previos en lenguajes de programación y manejo de –al menos– el lenguaje OTcl para realizar simulaciones utilizando elementos de redes existentes en la herramienta, sin la necesidad de modificarla. Por otro lado, cuando las implementaciones requieren módulos inexistentes en la herramienta, ésta debe ser modificada y para ello se deben agregar elementos y clases programadas en C++, y luego, programar el script en OTcl para realizar la simulación. Con esta descripción se intenta exponer la complejidad del simulador y la necesidad de un gran conocimiento previo para su uso.

El texto se encuentra organizado de la siguiente manera: En la Sección 2, se describe el marco histórico en el que surge el protocolo TCP, cómo se originan los problemas que motivan este proyecto de investigación y el entorno en el que surgen los simuladores ns-2 y ns-3. Luego, en el marco conceptual, se definen los conceptos involucrados; y en el marco teórico, se presenta el fundamento teórico que motiva el análisis de las herramientas de simulación involucradas. En la Sección 3 se comentan los trabajos relacionados con este estudio. La Sección 4, se enfoca en describir la metodología que se utiliza en esta investigación, para ello y con el fin de explicar el origen de los datos que debe procesar el software de simulación, se presenta una breve síntesis del enfoque de simulación y, en particular, de cada una de las herramientas utilizadas. Mientras que, en la Sección 5, a partir del diseño y simulación de modelos

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

sencillos, se presentan los diferentes resultados que pueden ofrecer cada uno de los simuladores. En la Sección 6 se discuten estos resultados y se plantea un marco de evaluación para las herramientas, a partir del cual se analizan las ventajas y desventajas de cada una de ellas y el ámbito donde cada una resulta más adecuada. Finalmente, en la Sección 7, se presentan las conclusiones que se pudieron obtener a partir del uso de los dos softwares de simulación y en la Sección 8 se plantean recomendaciones que incluyen posibles trabajos a futuro.

2. MARCO DE REFERENCIA

Marco histórico

Al estudiar un fenómeno, de acuerdo a las posibilidades y necesidades de los investigadores, se ponen de manifiesto diversas técnicas. Mediante la observación es posible examinar directamente algún hecho o fenómeno según se presenta de manera espontánea y naturalmente, recopilando datos de manera sistemática. La experimentación, en cambio, es un método que consiste en reproducir un fenómeno –comúnmente en un laboratorio– en condiciones particulares, eliminando o introduciendo aquellas variables que puedan influir en él, para su estudio. Estas técnicas dependen necesariamente del objeto de estudio. Cuando estudiamos redes informáticas, comunicaciones y protocolos, la observación o experimentación pueden llegar a ser técnicas costosas, poco prácticas e imposibles con las tecnologías disponibles.

El modelado de sistemas se postula como una alternativa para representar un sistema sin tener que implementarlo. Muchas veces los investigadores han utilizado enfoques analíticos para estudiar los sistemas mediante modelos matemáticos y la aplicación de métodos numéricos para obtener una percepción más precisa y extensa sobre el funcionamiento de los mismos. Sin embargo, la complejidad de algunos sistemas no permite las simplificaciones y suposiciones necesarias para aplicar este enfoque. Se hace necesario, entonces, aplicar el enfoque de simulación, recreando el sistema mediante herramientas de simulación.

Así, la simulación intenta modelizar sistemas reales o hipotéticos mediante sistemas informáticos, de forma que su funcionamiento pueda ser estudiado y podamos predecir su comportamiento. La historia y evolución de la simulación por ordenador han ido paralelas a la evolución de la informática. Sus orígenes se remontan a principios del siglo XIX, cuando dos matemáticos John Von Neumann y Stanislaw Ulam tenían el reto de resolver un problema complejo y como los experimentos basados en prueba y error eran muy caros, y resolverlo mediante técnicas analíticas lo hacía aún más complejo, lograron hacer una simulación mediante números aleatorios y distribuciones de probabilidad, a lo que llamaron “método de Montecarlo” (FIB/UPC, 2008). En la actualidad, la simulación se ha convertido en una técnica muy útil para experimentar y desarrollar aplicaciones en distintos ámbitos. Uno de los temas más estudiados es el análisis de rendimiento del protocolo TCP en redes inalámbricas (Trinidad & Talay, 2019).

El protocolo TCP es usado en la mayoría de las comunicaciones de datos actuales. Pero, desarrollado en la década del ‘70 para implementarse en topologías cableadas –predominantes en la época– y adaptado para ser aplicado en las redes inalámbricas a partir de su surgimiento a finales de la década del ’90, comenzó a evidenciar ciertas falencias en su funcionamiento.

En las redes cableadas, las pérdidas de paquetes frecuentemente se deben a una red congestionada. En redes las inalámbricas, en cambio, pueden surgir pérdidas por una multiplicidad de factores y no sólo por la congestión de la red. En este escenario son más frecuentes las pérdidas cuando surgen desconexiones por falta de señal, interferencias, ancho

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

de banda limitado y variable, entre otras causas. Distinguir el origen de las pérdidas ha provocado la introducción de nuevos algoritmos que modifican el protocolo TCP original y se presentan como variantes de TCP.
Cada variante surge para mitigar algunas deficiencias específicas, pero con la resolución de algunos problemas se suelen introducir otros. Por este motivo, a pesar de las mejoras implementadas y todas las propuestas presentadas por las diversas variantes a lo largo de los años, el protocolo continúa manifestando deficiencias en escenarios inalámbricos. En una actualidad donde los dispositivos móviles se multiplican diariamente y deben convivir con otros cableados, los defectos de TCP resultan en un deterioro del rendimiento de la red.
Por ello, resulta necesario continuar con investigaciones que puedan resolver el problema planteado y posibilitar un funcionamiento adecuado teniendo en cuenta los nuevos escenarios.
En el estudio del rendimiento del protocolo TCP en redes inalámbricas, se han utilizado de manera extensiva los simuladores de red. Dado que el eje de este informe son los simuladores de red en el ámbito de la investigación, no se considerarán los detalles específicos de la implementación de cada modelo sino sólo los necesarios para comprender el diseño de los mismos.
Los simuladores de la familia Network Simulator surgieron a partir de 1989 como una variante del simulador de red REAL y han evolucionado constantemente (USC/ISI, 2011). La primera versión, conocida como ns-1, fue desarrollada en el Lawrence Berkeley National Laboratory (LBNL) entre 1995 y 1997, fue el primer simulador para el campo de la educación e investigación (Avneet Kaur, Saluja, Sweta A, Dargad, & Krupali, 2017).
Surgió luego, en 1996, la primera versión del simulador ns-2 que conocemos. Su nueva arquitectura presentaba como novedad el Object Tcl (OTcl), en lugar del lenguaje Tcl que utilizaba el ns-1. Gran parte de las funcionalidades de ns-1 fueron fusionadas en ns-2, manteniendo casi total compatibilidad con las versiones anteriores. Los scripts Tcl desarrollados para ns-1 podían funcionar, generalmente de manera directa, en ns-2 (McCanne, Floyd, & Fall, 2009). Ns-2 luego de muchos años siendo el estándar de facto entre académicos e investigadores, ha posibilitado, por su accesibilidad y flexibilidad, el desarrollo de una cantidad considerable de publicaciones sobre análisis de protocolos.
Si bien, en 2006 fue liberada la primera versión del simulador ns-3, su predecesor continuó siendo mantenido mientras éste se construía y crecía, en parte, portando algunos de los modelos implementados en ns-2. Durante los años que convivieron ambas versiones, también se buscaron mecanismos de integración entre ellos, aún sin éxito. Finalmente, en 2011 se dio fin al mantenimiento de ns-2, aunque se creó una wiki específica para mantener su documentación, su código fuente sigue estando disponible en línea, y se pueden encontrar numerosos tutoriales y algunos foros de usuarios aún activos.

Marco conceptual

La simulación de redes es, sin duda, una de las metodologías de evaluación predominantes en una gran variedad de áreas del conocimiento y, en particular, en el área de redes informáticas. Es ampliamente utilizada para el desarrollo de nuevas arquitecturas de comunicación y protocolos de red (Weingärtner et al., 2009). En los ámbitos de investigación y educativos, donde la mayoría de las actividades se basan en entidades de simulación, y se trabaja con hipótesis y supuestos, las propuestas de trabajo están enunciadas en base a una topología determinada y un flujo de datos al cual está sometido el modelo que planteamos (Torres et al., 2015).
En el estudio de las redes, la simulación permite recrear el comportamiento de un sistema controlando los parámetros y configuraciones que influyen en una red, modelándola de manera arbitraria al especificar el comportamiento de cada uno de los nodos y los canales de

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

comunicación (Weingärtner et al., 2009). Este enfoque da como resultado una extensa lista de datos sobre cada instante de la simulación, lo que resulta abrumador para llegar a conclusiones objetivas.
Una plataforma de simulación de redes está formada por tres componentes: el modelo y los datos de entrada, la ejecución de la simulación, y la salida y el análisis consecuente. En la primera fase del proceso, las tareas son: (1) describir la topología de red; (2) caracterizar cada uno de los tráficos; y (3) definir cada uno de los eventos que tendrá lugar en el escenario de simulación (Marques, Plácido, & Sampaio, 2009; Nikoukaran, Hlupic, & Paul, 1998). Es decir, se plantea en esta fase el modelo de la simulación de red. En la segunda fase, la tarea es realizada íntegramente por la herramienta de simulación, basándose en el modelo diseñado en la primera fase y escrito en el lenguaje de programación correspondiente. Finalmente, en la tercera fase se analizan los resultados obtenidos y se arriba a las conclusiones, aunque no es el eje de este trabajo, es una tarea fundamental en todo trabajo de investigación, las actividades de esta fase fueron examinadas en un trabajo anterior (Almada & Talay, 2019).

Marco teórico

Una red es un conjunto de nodos interconectados mediante enlaces. Para que estos nodos puedan comunicarse es necesario que todos los dispositivos conozcan qué datos enviar, en qué formato, en qué momento y cómo responder a los mensajes, por ejemplo. Básicamente, un protocolo es un acuerdo entre las partes en comunicación sobre cómo se debe llevar a cabo la comunicación (Tanenbaum, 2003).
Uno de los protocolos fundamentales de Internet es el protocolo TCP (Transmission Control Protocol), que opera en la capa de transporte del modelo TCP/IP y se encarga de llevar datos de un extremo a otro de la red. Su popularidad se debe a que es confiable y robusto; orientado a la conexión, implementando mecanismos de control para garantizar que la información llegue de forma completa, ordenada y correcta; y, además, utiliza el estado de la red para modificar la tasa de transmisión de paquetes con el objetivo de optimizar la utilización de recursos. Sin embargo, la problemática de transmisión de datos en topologías mixtas que implican atravesar medios cableados e inalámbricos presenta un grado de complejidad específico que habilita su estudio en forma extensa. Para llevar adelante el análisis de las variantes del protocolo TCP se recurre al ensayo de simulaciones con dos versiones del simulador Network Simulator: ns-2 y ns-3.
Muchos trabajos de investigación han realizado estudios de desempeño de las variantes de protocolo TCP en distintos escenarios y para ello la selección del simulador a utilizar es una tarea básica y fundamental que permite elaborar los informes que se publican en estas investigaciones.
Ns-2 y ns-3 son dos simuladores ricos en funcionalidades y documentación, ampliamente utilizados en ámbitos académicos y de investigación. Son herramientas tan útiles como complejas, por lo que requieren un conocimiento relativamente avanzado sobre su arquitectura, características, y en particular, alcances y limitaciones (Trinidad & Talay, 2019).

3. REVISIÓN BIBLIOGRÁFICA

Existe una gran variedad de herramientas para la simulación de redes, algunas de las más conocidas son OPNET, que actualmente se llama Riverbed Modeler (Riverbed, 2020), OMNeT++ (AAVV, 2020b), JSIM (AAVV, 2020a), y REAL (Keshav, 1988), entre otras. Si bien en el ámbito educativo y, particularmente, en el científico el uso de los simuladores ns-2 y ns-3 es extensivo, otros autores han incluido en sus estudios una amplia lista de alternativas.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Jaganath et al. (2019) presentan un análisis comparativo a manera de estado del arte sobre veinte herramientas de simulación, indicando brevemente las características y limitaciones de cada una. Luego presentan los resultados de su estudio en una matriz donde contrastan para cada una: el tipo de licencia requerido, los sistemas operativos para los que se encuentran disponibles y el soporte respecto a la movilidad de los nodos, el tipo de red (cableadas o inalámbricas), el número de norma IEEE, para qué capa del modelo OSI se utiliza, y el consumo energético que en el caso de ns-2 y ns-3 no aplica. De la misma manera, otros autores como Arvind (2016), Calle et al. (2018) y Gupta et al. (2013) han formulado comparaciones similares nombrando a ns-2 y ns-3 en sus exhaustivas listas de simuladores. Calle et al. (2018) además de resumir las ventajas competitivas de cada una de las herramientas, entregan una serie de pautas para realizar una selección adecuada de las mismas.
Saluja et al. (2017), en cambio, se concentraron en las proximidades y diferencias entre los simuladores ns. Ofrecen una revisión desde ns-1 hasta ns-4, haciendo hincapié en las compatibilidades y problemas entre cada simulador. Ns-1 es la primera versión, sobre la que más tarde se basó ns-2 y ns-4 no es un simulador nuevo en sí, sino que es el diseño de un prototipo para extender ns-3 planteado en una publicación científica (Fan, Bi, Zhou, Zhang, & Yu, 2017).

4. METODOLOGÍA

Herramientas de software y métodos utilizados

Para el desarrollo de la investigación se utilizaron dos simuladores de redes, Network Simulator-2 (ns-2, *v. 2.35*) y Network Simulator-3 (ns-3, *v.3.30.1*). Además del ensayo con ambas herramientas, se recurrió a su documentación oficial, manuales, publicaciones científicas e información disponible en la web, tanto de la comunidad de usuarios como la publicada por los desarrolladores y colaboradores de ambos proyectos.
La metodología en la investigación comienza con la formulación del problema. En esta fase de especificación, el investigador plantea el problema y presenta tanto los objetivos e hipótesis del estudio como las características de la simulación a llevar a cabo.
Luego de reunir estos datos y definir el escenario a simular, se debe reflejar esto en un lenguaje que el simulador pueda comprender, OTcl, Python o C++, de acuerdo a la versión con que se esté trabajando. De esta manera se describe la topología de red, la pila de protocolos y parámetros específicos de cada uno de ellos, entre otras particularidades que pueden definirse.
Finalmente, se llevan a cabo las ejecuciones de simulación en la herramienta de simulación seleccionada, es decir, se ejecutan los scripts diseñados y luego se recopilan los resultados. No obstante, los resultados obtenidos del simulador no se encuentran en un formato fácilmente legible para humanos. Por ello, es necesario realizar un procesamiento precio a las trazas obtenidas por el simulador. Y luego, también de acuerdo al simulador utilizado, se utilizará alguna de las herramientas disponibles que permiten su visualización, análisis, procesamiento y edición.
En este trabajo se realizará un análisis que describirá de manera comparativa los simuladores ns-2 y ns-3, las principales herramientas de simulación utilizadas en el ámbito de investigación en redes informáticas, y que han surgido una como reemplazo de la otra, pero son totalmente incompatibles. El uso de alguna de estos softwares permitirá a los investigadores ejecutar las simulaciones para luego analizar los resultados y documentar las conclusiones. Así, tomando como referencia el marco de trabajo utilizado en (Calle et al.,

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

2018; Velásquez & Gamess, 2009), se define en primera instancia, un conjunto estandarizado de parámetros y criterios que se utilizarán para el análisis y comparación de los dos simuladores.

Parámetros a analizar para elegir un software simulación

A continuación, se presentan parámetros que resultan útiles para el análisis comparativo de las dos herramientas de simulación que ayudarán al lector a visualizar las similitudes y diferencias entre ambas versiones.

En la Sección 4, la Tabla 1 corresponde a una matriz donde para cada uno de los simuladores analizados, se presenta el valor de cada uno de los parámetros aquí descritos. La tabla 1, por lo tanto, se constituye en una herramienta rápida de comparación y selección (Calle et al., 2018).

Primer lanzamiento

Conocer el momento histórico donde surge una herramienta nos permite saber la antigüedad del proyecto y asociado a ello, si el mismo sigue activo, la cantidad de usuarios, actualizaciones y fiabilidad que puede brindar.

Base

Este parámetro indica en que paquete o aplicación se basa la herramienta en cuestión. Es una guía para saber cuál fue el origen o comienzo de cada uno de los simuladores.

Tipo de simulador

Existen diferentes modelos de simulación: simulación de eventos discretos o continuos.

Versión actual y última actualización

La fecha de la última actualización es importante ya que muestra qué tan activo es el proyecto, mientras que el número de versión actual indica sobre qué versión de las herramientas se hizo el análisis en este trabajo.

Plataformas soportadas (GNU/Linux, MS Windows, Otros)

Es indispensable saber de antemano si las herramientas son compatibles con el sistema operativo que el investigador o estudiante utiliza habitualmente. Si bien éste no es un impedimento para utilizarlas (se puede recurrir a máquinas virtuales, por ejemplo), el soporte de la plataforma a la que está familiarizado el usuario será una característica básica a tener en cuenta.

Licencia y derechos de autor

La licencia de software, así como los derechos de autor son características que se deben conocer siempre al elegir una herramienta. Es común que el investigador, debido a las particularidades de sus datos o de sus objetivos de investigación, necesite desarrollar algoritmos de análisis y/o de procesamiento propios. Esta posibilidad de desarrollar algoritmos propios no siempre está incorporada en el software privativo [13].

Soporte.

El soporte de cualquier aplicación es una parte importante en cualquier herramienta informática que se utilice, ya que ante cualquier inconveniente o duda se puede recurrir a soporte oficial.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Dificultad de descarga e instalación

La descarga e instalación de un software es el primer acercamiento a un software nuevo, por lo que la dificultad en hacerlo influirá en gran medida en continuar con su utilización en caso de muchas dificultades, o abandonar antes de utilizarla. También tendrá efecto en la disposición del futuro usuario hacia el software.

Cantidad de documentación disponible

La cantidad de documentación, y exactitud de la misma, es un aspecto valioso a la hora de utilizar una herramienta nueva. Además, la disponibilidad de una página web oficial, foros de usuarios, manuales de uso y recopilación de FAQ (preguntas frecuentes) resultan de suma utilidad para resolver dudas e inconvenientes en el uso de cualquier aplicación.

Curva de aprendizaje requerida

Aprender su utilización y resolver las dificultades que se encuentren de manera adecuada, son también objetivos relacionados con la curva de aprendizaje y pueden determinar que el estudiante o investigador se anime a conocer cada detalle de su funcionamiento y hacerse hábil en el uso de la herramienta, o bien, abandone con frustración su uso.
Este parámetro considera aspectos fundamentales para medir, como conocimientos previos requeridos y uso didáctico de la herramienta que contempla qué tan amigable es la interacción con el usuario.

Presentación (Interfaz gráfica (GUI), Línea de comandos (CLI))

En cuanto a la presentación de la interfaz, no se puede aducir que la interfaz por línea de comandos sea superior a la interfaz gráfica o viceversa, no obstante, la elección de la misma dependerá de la experiencia y preferencia de cada usuario, así como de cada situación de uso. Por ello, al elegir la herramienta adecuada, es un aspecto a tener en cuenta que la misma trabaje con el tipo de interfaz con la que el usuario se sienta más cómodo.

Frentes gráficos disponibles.

Para muchas herramientas diseñadas para uso exclusivo por línea de comandos, se han diseñados otras que proponen una interfaz gráfica para su uso. Esto resulta sumamente práctico para los usuarios menos experimentados para realizar simulaciones básicas y, principalmente, en las primeras etapas de aprendizaje.

Posibilidad de diseñar y modificar los escenarios de red

Aunque para entender el funcionamiento de los protocolos y las redes puede ser suficiente, en el ámbito académico, el uso de escenarios predefinidos. Resulta indispensable para todo investigador que la herramienta seleccionada le permita definir sus propios escenarios.

Lenguaje de programación para modificar el simulador

Si la licencia lo permite, es probable que los investigadores puedan plantearse modificaciones en el código fuente para que el simulador se adapte a sus necesidades, o bien para colaborar con la comunidad de usuarios reparando errores o agregando funcionalidades. Para ello, es importante conocer el lenguaje en el que está implementado el software.

Lenguaje de programación para definir las simulaciones

Resulta esencial para todo investigador poder codificar sus propios programas de simulación, para implementar todos los escenarios que desee. Para realizar esto, debe conocer el lenguaje de scripting que maneja el simulador para definir las simulaciones, y no siempre coincide con el lenguaje en el que está escrita la herramienta.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Contribución de códigos y modelos por parte de la comunidad de usuarios
En relación a la magnitud de la comunidad de usuarios activos y de desarrolladores, debe existir un repositorio con las contribuciones de diferentes códigos y modelos que puedan reducir la carga de trabajo de nuevos usuarios e investigadores.

Formato de salida
De acuerdo a la finalidad de los gráficos o animaciones es de suma importancia estudiar los formatos de salida que ofrece cada simulador, así como también la posibilidad de imprimir los gráficos. Se agregan como sub parámetros los tres resultados tradicionales en los simuladores de redes: animación de los eventos de la simulación, archivos de traza y archivos de rastreo de paquetes.

Compatibilidad con herramientas de procesamiento gráfico
En relación al parámetro anterior, se incluyen en este, las aplicaciones disponibles para ejecutar cada una de las salidas básicas del simulador.

Módulos disponibles
Es importante conocer los módulos incluidos con la herramienta, aunque también pueden surgir muchos otros del aporte de la comunidad de usuarios y desarrolladores.

Interacción con aplicaciones reales
Los simuladores de redes no sólo son utilizados en la investigación, en ocasiones los administradores de red necesitan medir distintos indicadores. En estos casos, la función del simulador es emular condiciones específicas previo a la instalación de una red compleja en el escenario real. Permitir este paso lo más directo posible es una capacidad valiosa en un simulador.

5. RESULTADOS

El proceso de una simulación comienza con la especificación de los requerimientos de la misma, es decir, el diseño del escenario a simular. En el caso de ambos simuladores, la especificación informal del escenario se realiza de la misma manera, se deciden aspectos básicos: la topología a utilizar, la cantidad de paquetes que se enviarán o la duración de la simulación, el tamaño de los paquetes, entre otros detalles técnicos.
El proceso difiere al implementar de los modelos diseñados, con cada uno de los simuladores analizados esta etapa se materializa de manera diferente. Ns-2 requiere que se haga la especificación formal de la simulación en OTcl, la versión orientada a objetos de Tcl. Ns-3, en cambio, plantea la implementación del modelo en C++, el lenguaje del simulador, o de manera opcional, mediante un script de simulación Python (en reemplazo del OTcl que utilizaba la versión anterior).
Tanto Tcl como OTcl son lenguajes poco conocidos, por lo que cualquier estudiante o investigador requerirá aprender su sintaxis para poder realizar un script para ns-2 correctamente. C++ y Python son lenguajes muy conocidos y fáciles de aprender. Para justificar la elección de los desarrolladores en este punto fundamental, se recurre a la aplicación publicada por la revista IEEE Spectrum (Diakopoulous, 2018) que actualiza, de acuerdo a la ponderación de diferentes métricas, la popularidad de los lenguajes de programación. En este ranking, cuya captura se muestra en la Fig. 1, se evidencia la

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

popularidad de los lenguajes C++ y Python –que ocupan los primeros dos lugares–, y el desuso en el que ha caído Tcl (en el puesto número 41).

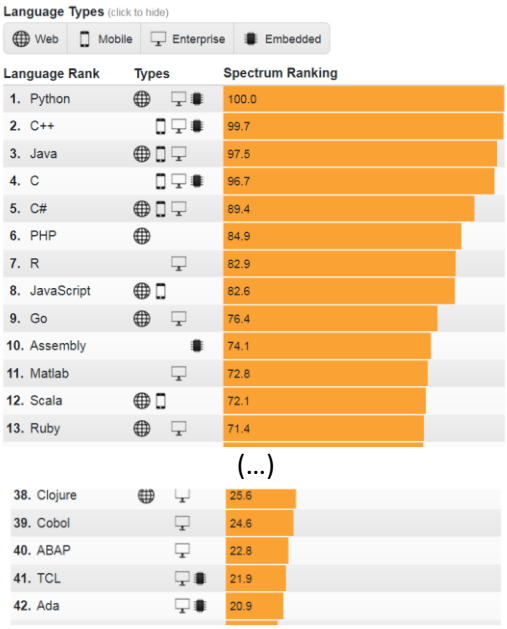

Figura 1. *Captura de la aplicación IEEE Spectrum Top Programming Languages*

Esta decisión fundamental de diseño en el nuevo simulador, que eliminó las posibilidades de compatibilidad con los scripts programados para ns-2, se justifica al observar la popularidad de los lenguajes de scripting que admite ns-3. Los desarrolladores lograron de esta manera, eliminar la complejidad que requería la combinación de dos lenguajes de programación en el uso de la herramienta, manteniendo sólo C++ y agregando Python de manera opcional.

Podemos especificar, entonces, que ns-3 es una biblioteca de C++ que proporciona un conjunto de modelos de simulación de red implementados como objetos C++ y envueltos con Python. Para usar esta biblioteca, normalmente los usuarios escriben una aplicación en C++ (o en Python) que crea ejemplares de un conjunto de modelos de simulación para construir el escenario de simulación deseado, ejecutar el hilo principal y salir de él cuando se complete la simulación.

En cuanto a los módulos disponibles para cada una de las herramientas, ns-2 cuenta con un conjunto amplio aportado por la comunidad mientras que ns-3, por su parte, cuenta con modelos más detallados en muchas de las áreas más populares en investigación actualmente (incluye módulos detallados para LTE y Wi-Fi). Otra potente utilidad es que toda la pila de red de Linux puede ser encapsulada en un nodo de ns-3, se consigue a través de un framework llamado Direct Code Execution (DCE). Además, ns-3 incluye también un planificador en tiempo real que facilita las simulaciones en bucle para interactuar con sistemas reales (Almazán, 2014).

Una característica interesante al momento de generar las pruebas con el simulador es la presentación del mismo. Se conoce que el uso por excelencia de ambos es mediante la línea de comandos y se requiere, por lo tanto, el conocimiento de diferentes lenguajes de programación. Sin embargo, para el ns-2 –probablemente debido a su antigüedad– han sido diseñadas alternativas gráficas para utilizar el simulador o generar los programas de la simulación de manera automática. Se encuentran disponibles algunos frentes gráficos como NS2 Scenarios Generator (NSG2), Visual Network Simulator (VNS), nsBench o Network Simulator by Mouse (NSBM), entre otros. En el caso de ns-3, no existen –hasta el momento en que se escribe este informe– frentes gráficos disponibles.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Es necesario aclarar en este punto, que el funcionamiento de ambos al momento de generar las pruebas es idéntico: Se ejecuta el simulador mediante la terminal del sistema, se le indican cuáles son los programas de la simulación a ejecutar y comienza su actividad. Al finalizar su ejecución se podrá comprobar, en los directorios adecuados, la creación de los archivos de salida que se hayan indicado al configurar la simulación.
La salida principal de cualquier simulador de redes debe ser un archivo de traza que indique el detalle de los resultados de la simulación. Esta salida tiene formato .tr y se puede obtener mediante ambas versiones del simulador. Una segunda salida que se obtiene de estos simuladores es un fichero de animación que permite visualizar la topología, distinguir los tipos de paquetes, nodos y enlaces, asignándoles diferentes íconos, colores, formas y etiquetas. Ambos simuladores incluyen la posibilidad de generar este tipo de archivo, pero con diferentes formatos para aplicaciones específicas. Mientras que ns-2 utiliza la extensión .nam, un archivo que se ejecuta mediante una aplicación que lleva el mismo nombre, ns-3 genera archivos xml que pueden ser leídos por la herramienta netanim. En ambos casos las aplicaciones asociadas suelen ser instaladas junto a cada simulador. Ambas facilitan la interpretación de los resultados y permiten su visualización de manera dinámica. A medida que avanza, se puede hacer el análisis de paquetes ya que al pulsar cada uno de ellos se obtiene el tipo, tamaño y número de secuencia. Nam ofrece, además, opciones para guardar estas animaciones en formato de video MPEG o imágenes GIF, y netanim posee pestañas adicionales con estadísticas y gráficos de la simulación. En la Fig. 2 se puede observar la interfaz y la captura de una animación en curso en nam, en la Fig. 3 se observa la interfaz y la captura de una animación en curso en su análogo para ns-3.

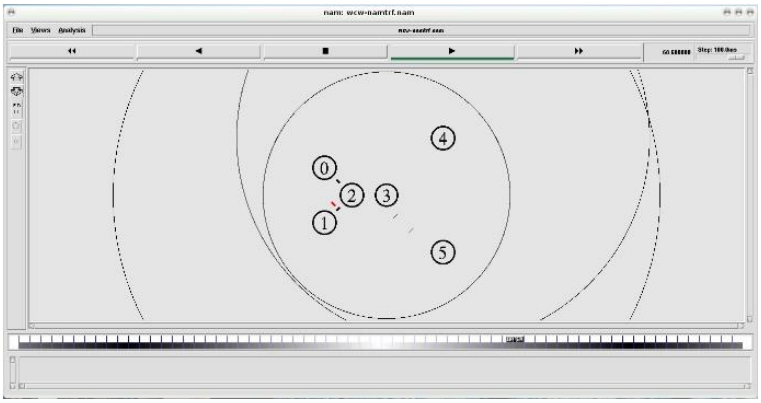

Figura 2. *Network Animator reproduciendo una simulación*

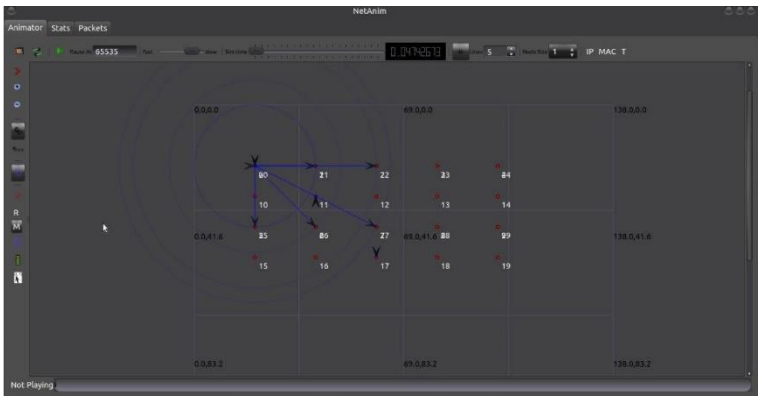

Figura 3. *Netanim reproduciendo una simulación*

Si bien estas herramientas permiten ejecutar sólo un ensayo por vez, resultan un complemento adecuado para los simuladores ya que posibilitan una rápida validación de la simulación

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

realizada a partir de un análisis visual del escenario que debe coincidir con el modelo del sistema que se pretendía configurar. A pesar de las ventajas presentadas que impulsan su uso, en el ámbito de la investigación resulta insuficiente y requiere complementarse con otras herramientas de representación gráfica.

El segundo tipo de resultado que brindan los simuladores es la generación de archivos de traza. Dado que el objetivo de la simulación no es conocer el modelo en sí sino el comportamiento que este tiene ante situaciones particulares, esta es la salida más informativa, que además permite múltiples ejecuciones a la vez, lo que asegura su escalabilidad. Los archivos de traza muestran información relevante de la simulación realizada. A partir de ellos, se pueden obtener algunos parámetros de rendimiento de la red, que permiten estudiar el comportamiento de distintos fenómenos como congestión (Esterhuizen & Aes, 2012; Jacobson, 1988), colisiones y competencia por recursos (González et al., 2019), entre otros. Es tanta la información que puede contener una traza, que habitualmente las herramientas para presentar los datos requieren de un proceso de filtrado previo que prepare estos archivos de acuerdo a los gráficos que se desee obtener.

En ambos simuladores, salvando pequeñas diferencias de acuerdo a si se trata de trazas cableadas o inalámbricas, el formato de las trazas es similar. En la Fig. 4 se muestra, sólo a modo de ejemplo, un extracto de un archivo de traza cableada. Los detalles específicos de cada uno de los datos que contiene y su significado no son relevantes para esta discusión. A partir de la traza se obtienen los parámetros directos de la simulación, y luego, realizando un procesamiento de esos datos mediante los cálculos y filtros adecuados, se pueden calcular los parámetros indirectos que son los que aportan información valiosa para presentar en gráficos, tablas o planillas que permitan a los investigadores hacer el análisis correspondiente.

```
+ 11.028154 3 2 ack 60 ------- 2 1.1.1.0 0.0.1.0 248 614
- 11.028154 3 2 ack 60 ------- 2 1.1.1.0 0.0.1.0 248 614
r 11.028155 3 2 ack 60 ------- 2 1.1.1.0 0.0.1.0 248 614
+ 11.028155 2 1 ack 60 ------- 2 1.1.1.0 0.0.1.0 248 614
- 11.028155 2 1 ack 60 ------- 2 1.1.1.0 0.0.1.0 248 614
```

Figura 4. *Ejemplo de traza cableada como resultado de una simulación en ns-2.*

El tercer tipo de resultado generado por la simulación se incluye únicamente en el simulador ns-3, consiste en archivos de formato pcap, uno por cada paquete de la simulación. Este aporte en la nueva herramienta es sumamente útil porque permite de manera rápida examinar los resultados de la simulación, con herramientas similares a las que se utilizarían en una observación de la realidad. Estos archivos pueden ser estudiados mediante cualquier analizador de paquetes disponible, los más conocidos son Wireshark (Combs & Col., 2020) o tcpdump (Jacobson, Leres, McCanne, & Lawrence Berkeley National Laboratory, University of California, Berkeley, 2019). En la figura 5, se muestra la interfaz de Wireshark al abrir un conjunto de estos archivos generados por el simulador y el uso de algunas de las herramientas adicionales que provee el software, como la generación de gráficos sencillos.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

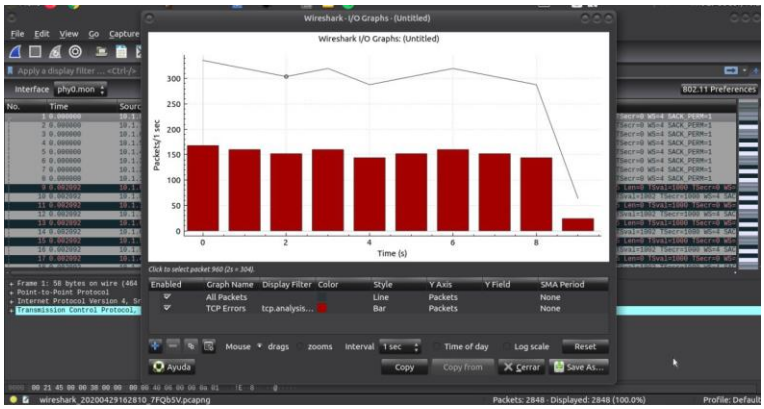

Figura 5. Utilización de herramientas de Wireshark para analizar paquetes de una simulación

En la Figura 6 se ha planteado un esquema que muestra el proceso general de simulación en ambas herramientas. Se puede observar que no hay diferencias considerables en el proceso, sino que difieren los lenguajes y herramientas a utilizar. La generación de pruebas en los simuladores se concreta con los tipos de salida descriptos: (1) los archivos para ejecutar la animación; (2) los archivos de traza para el procesamiento posterior y la ejecución de múltiples tipos de gráficos y análisis; o también, (3) los archivos de rastreo de paquetes, en el caso de ns-3.

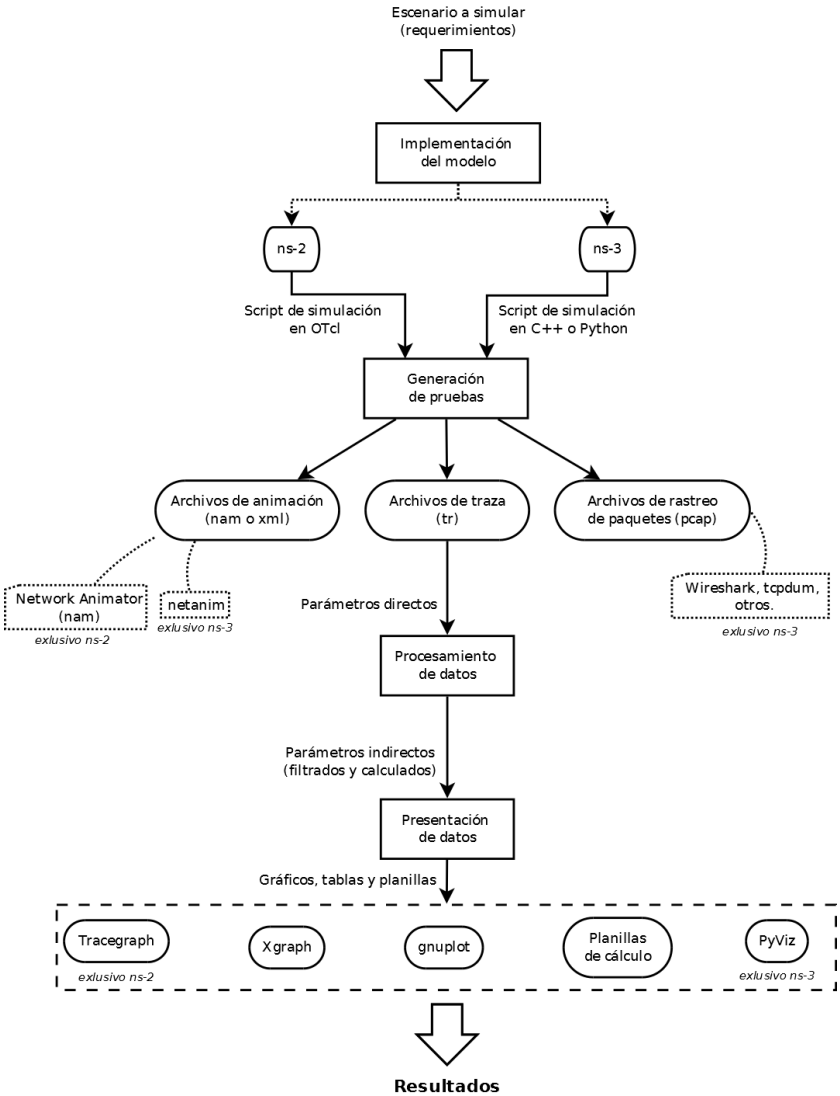

Figura 6. *Estructura del proceso de simulación en ambos simuladores*

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

6. DISCUSIÓN

La última actualización (v.2.35) del simulador ns-2 fue lanzada en noviembre de 2011, y a partir de entonces, el soporte de la herramienta finalizó. La falta de soporte comercial y los recursos limitados para continuar el mantenimiento a largo plazo de un código base que crece continuamente, provocaron el cese de sus actualizaciones.

En contraste a esta situación, aunque el simulador ns-3 tampoco cuenta con soporte comercial, lanza actualizaciones frecuentemente, su última versión estable (v.3.31) fue publicada recientemente (en junio de 2020). La contribución de 48 autores permitió corregir errores y añadir numerosas características (Nsnam, 2020).

Si bien ambas herramientas son usadas principalmente en sistemas Linux o macOS, también existe soporte para BSD y entornos para Windows que pueden compilar código Linux, como Windows Subsystem for Linux (Microsoft, 2019) o Cygwin (Vinschen, Turney, & Blake, 2020). En el caso de ns-3 estos emuladores pueden presentar problemas con algunas librerías de Python, por lo que se recomienda a usuarios de Windows utilizar una máquina virtual Linux para mejor desempeño.

En cuanto a la presentación, ambos carecen de una interfaz gráfica de entrada que facilite la navegación entre las diferentes opciones o parámetros para simulación de escenarios tipo (Rodríguez Pérez, 2012) y el uso por excelencia de ambos se da mediante la terminal del sistema. Si bien ns-2 posee varias herramientas gráficas para utilizar como complemento del simulador, sus funcionalidades y configuraciones son bastante limitadas, y la mayoría de ellas se encuentra en proyectos abandonados o desactualizados, sin soporte oficial ni documentación amplia, por lo que, los comportamientos pueden no coincidir con los esperados en la práctica. Ns-3 no cuenta hasta el momento con ninguna alternativa similar para superar la línea de comandos, al menos para simulaciones básicas. Se espera que a futuro surjan proyectos con este fin.

La arquitectura base de ambos simuladores es diferente, si bien algunos componentes de ns-2 están escritos en C++, contiene otros en OTcl. Esta versión orientada a objetos de Tcl puede resultar poco familiar para la mayoría de los estudiantes y difícil de depurar (Gross, Wehrle, & Güneş, 2010). Así, la principal diferencia entre estos dos simuladores es arquitectónica, ns-3 se aleja de la combinación entre OTcl y C++, y plantea sus modelos íntegramente en C++, con enlaces opcionales en Python. Esto, además de adaptarse mejor a los lenguajes actuales, aporta rendimiento y facilidad en la depuración: los usuarios de ns-3 son libres de escribir sus simulaciones en Python, o en C++.

Poder diseñar escenarios propios para simular es uno de los aspectos básicos que debe cubrir cualquiera de las herramientas. La forma de implementar estos modelos utilizando ns es escribiendo los programas de simulación en un lenguaje que entienda el simulador. En el caso de ns-2, los códigos deben estar programados en OTcl, y no es posible ejecutar simulaciones únicamente utilizando C++. Los scripts de simulación en ns-3 pueden escribirse en C++ o en Python –un lenguaje moderno y fácil de aprender– de manera indistinta.

Si bien ns-2 cuenta, debido a su larga historia, con un conjunto más variado de módulos aportados por la comunidad, ns-3 cuenta con modelos más detallados en muchas de las áreas más populares en investigación actualmente (como WiFi o LTE). Ns-3, a diferencia de su antecesor, brinda facilidades para realizar simulaciones en bucle e interactuar con sistemas reales (Almazán, 2014).

La gran cantidad de módulos que logró ns-2 de parte de la comunidad también generó inconvenientes. Ns-2 no impuso un estándar de codificación, aceptando modelos con pruebas de inconsistencia en el software y falta de verificación de los modelos, así como escasez en el diseño general del sistema (Gross et al., 2010). Esto permitió que la herramienta creciera considerablemente a lo largo de los años, pero llevó a que los usuarios perdieran la confianza

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

en los resultados, que el software fuera menos flexible para su reconfiguración y finalmente, los retos y obstáculos para su mantenimiento fueron tantos que su actualización fue insostenible en el tiempo. Por este motivo, ns-3 impuso un estándar de codificación más riguroso, un proceso de revisión de código y una infraestructura de pruebas. Esto sumado al uso de un único lenguaje de programación, con enlaces a Python u otros en un futuro.

Aunque ambos simuladores están enfocados en el sector educativo como herramienta de simulación para enseñanza e investigación (Calle, Tovar, Castaño-Pino, & Cuéllar, 2018), ns-3 tiene características adicionales que lo hacen valioso en muchos otros ámbitos. Como es compatible con aplicaciones de red reales del sistema operativo GNU/Linux, brinda la posibilidad de desarrollar y probar aplicaciones y funcionalidades en entornos simulados garantizando un comportamiento igual cuando se despliegue la misma en un dispositivo de red real. Asimismo, permite la interacción con dispositivos de red del mundo real (Alexander et al., 2017).

Además de su arquitectura y módulos disponibles, es importante exhibir la manera en que se presentan los resultados de ambos simuladores. Tanto los resultados de ns-2, como los de ns-3, se pueden visualizar de manera similar mediante gnuplot o Xgraph, ya que estas herramientas admiten los archivos de traza de formato tr que utilizan ambos simuladores como formato de salida. Hay otras herramientas de graficación que no han sido actualizadas y son de uso exclusivo para ns-2, como Tracegraph (v2.05) o NetworkAnimator (nam) normalmente instalado por defecto junto a ns-2.

Se encuentra disponible el repositorio de un colaborador que planteaba la adaptación de nam a ns-3 (Lacage, 2009), aunque el proyecto no ha sido integrado a la herramienta y finalmente ha sido abandonado, puede ser de utilidad para usuarios interesados en darle continuidad.

Para uso exclusivo de ns-3, se encuentran disponibles algunos nuevos visualizadores y animadores, como PyViz y NetAnim (PyViz, s. f.; Riley, 2017) y otros en desarrollo actual. Además, dado que ns-3 también genera archivos de rastreo de paquete con formato pcap, también se pueden utilizar otras utilidades para analizar rastreos, como Wireshark (Combs & Col., 2020) o tcpdump (Jacobson et al., 2019).

Los aspectos que varían de forma más significativa entre un simulador y otro se refieren al trabajo involucrado, principalmente, a la facilidad para generar los programas de simulación y a la búsqueda de módulos disponibles que se adapten con las necesidades del investigador. Es ineludible el hecho de que ambos simuladores cuentan con una empinada curva de aprendizaje, de modo que se requiere conocimiento técnico en el área de las redes de comunicación, protocolos y estándares de red. Además, el uso de estas potentes herramientas requiere tener claros los conocimientos básicos en programación y el manejo de diferentes lenguajes de programación para obtener los resultados esperados. El grado de esfuerzo invertido en aprender la utilización de uno de estos softwares, dependerá del ámbito del usuario. Estas son algunas de las observaciones que surgen a partir del uso y estudio de la documentación de los simuladores:

- Se recomienda el uso para ambos simuladores sobre plataformas Linux, ya sea local o virtualmente, especialmente con ns-3 donde se han reportado que los enlaces a Python no funcionan correctamente.
- Si se desea introducir en el uso de estas herramientas, ns-2 tiene una gran cantidad de modelos ya desarrollados por la comunidad, ns-3 por su parte, tiene un completo tutorial incluido en la herramienta con los ejemplos básicos predeterminados para comenzar a aprender.
- Ambas herramientas poseen una documentación muy completa, con actualizaciones para cada versión lanzada. Ns-3 posee además soporte con actualizaciones frecuentes, por lo que se espera que continúe adaptándose al uso en sistemas modernos y simulaciones de nuevas tecnologías.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

- Los programas de simulación para ns-2 deben ser escritos en un lenguaje poco conocido, pero fácil de aprender (OTcl), los de ns-3, en cambio, pueden codificarse mediante C++ (un lenguaje básico y conocido entre programadores) o Python, fácil de aprender y muy popular en muchos ámbitos durante los últimos años.

A continuación, se realiza un compendio de las diferencias más importantes detectadas en el estudio de los simuladores, considerándose los parámetros de comparación definidos en la Sección 3, como última actualización, plataformas soportadas, tipo de licencia, tipo de interfaz –gráfica o por línea de comandos–, el lenguaje utilizado para los programas de simulación personalizados y los posibles formatos de salida, entre otros aspectos. Se realiza la evaluación de cada uno de ellos, en los dos simuladores, con el fin de establecer un cuadro comparativo (Tabla 1) que le permita al usuario visualizar de una manera clara las ventajas y desventajas de elegir una herramienta u otra (Calle et al., 2018).

Tabla 1. *Parametrización de los simuladores ns-2 y ns-3.*

Aspecto		**ns-2**	**ns-3**
Primer lanzamiento		1996	2008
Base		ns-1 y Real-world Simulator	ns-2, GTNets y YANS
Tipo de simulador		Eventos discretos	Eventos discretos
Versión actual y última actualización		2.35 (Noviembre 2011)	3.30.1 (Septiembre 2019)
Plataformas soportadas	GNU/Linux	✓	✓
	MS Windows	Mediante Cygwin.	Mediante Cygwin o WSL.
	Otros	FreeBSD, SunOS, Solaris, otros mediante máquina virtual Linux	FreeBSD, OS X, otros mediante máquina virtual Linux
Licencia y derechos de autor		Software libre. Licencia pública GNU v2.	Software libre. Licencia pública GNU v2.
Soporte		El soporte oficial finalizó en 2011.	No comercial. A través de la lista de correo de usuarios y grupo de usuarios en Google Groups.
Dificultad de descarga e instalación		Alta	Media
Cantidad de documentación disponible		Alta	Alta
Curva de aprendizaje		Alta	Alta
Presentación	Línea de comandos (CLI)	✓	✓
	Interfaz gráfica (GUI)	✗	✗
Frentes gráficos disponibles		Varios, con funcionalidades limitadas: NS2S, VNS, nsBench y NSBM, entre otros.	No existen complementos para añadir interfaz gráfica al simulador hasta el momento.
Posibilidad de diseñar y modificar los escenarios de red		✓	✓
Lenguaje de programación para modificar el simulador		C++	C++
Lenguaje de programación para definir las simulaciones		OTcl	C++ o Python (indistintamente)
Contribución de códigos y modelos por parte de la comunidad de usuarios		Alta	Baja
Implementa estándares de codificación		✗	✓

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

Formato de salida	Archivos de traza (tr)	✓	✓
	Animación de la simulación (nam/xml)	✓	✓
	Rastreo de paquetes (pcap)	✗	✓
Compatibilidad con herramientas de procesamiento gráfico	Gráficos basados en la traza de resultados	Tracegraph, gnuplot, Xgraph	gnuplot, Xgraph, PyViz
	Gráficos animados basados en los eventos de la simulación	nam	netanim
	Otros	•	Wireshark, tcpdump
Módulos disponibles	Redes cableadas	✓	✓
	Redes inalámbricas	✓	✓
	Ad-Hoc	✓	✓
	Redes inalámbricas de sensores	•	✓
Interacción con aplicaciones reales		•	✓
Escalabilidad		Simulación secuencial	Simulación distribuida
Paquetes		Dos regiones distintas: una para el encabezado y la segunda almacena la carga útil de datos.	Único buffer de bytes, se puede agregar de manera opcional una colección de pequeñas etiquetas que contengan metadatos.
Ventajas		- Facilidad para agregar nuevos protocolos. - Se han publicado un gran número de protocolos. - Es el estándar de facto de los simuladores libres, en ámbitos académicos e industriales.	- No es una extensión de ns-2, sino un nuevo simulador. - Soporta virtualización. - Se ha establecido una amplia comunidad de usuarios y continúa creciendo. - Se utilizan lenguajes ampliamente conocidos en la comunidad informática (C++ y Python)
Desventajas		- Soporta sólo dos MAC de protocolos inalámbricos: 802.11 y protocolos de acceso múltiple TDMA. - Necesidad de familiarizarse con el lenguaje de scripting (OTcl). - Arrastra carencias de diseño que han hecho necesarios muchos parches y extensiones.	- Los enlaces a Python no funcionan en Cygwin. - Los scripts desarrollados para ns-2 no funcionan en ns-3.

7. CONCLUSIONES

Para examinar protocolos de red y estudiar el desempeño de TCP en redes inalámbricas –o en combinación con redes cableadas– resulta fundamental el uso de un simulador confiable y robusto que permita llevar a cabo pruebas sin que esto requiera el despliegue de toda la red en la realidad y que además aporte la posibilidad de aislar los eventos necesarios para controlar

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

lo que sucede en la red simulada. De esta manera, se puede lograr una comprensión íntegra del fenómeno estudiado.

En el ámbito de la investigación, y también en el académico, resulta fundamental seleccionar la herramienta adecuada y poder utilizarla apropiadamente. Por ello, elegir el software de simulación que se comenzará a utilizar es una tarea necesaria y un estudio comparativo entre los dos simuladores de mayor prestigio en el área resulta de gran ayuda al momento de realizar un análisis.

Mediante la experimentación con ambos simuladores, se advirtió que la curva de aprendizaje de ambos es similar, lleva mucho tiempo y dedicación poder utilizar los simuladores para explotar sus características de manera efectiva. Sin embargo, la posibilidad de acceder a tutoriales oficiales o aprender con modelos ya diseñados por otros usuarios y la amplia documentación que ambos simuladores tienen es un factor que acelera el proceso.

Así como ambas versiones del simulador son herramientas de software totalmente diferentes desde su base arquitectónica, las herramientas complementarias que se utilizan en la investigación, también difieren. En trabajos anteriores (Almada & Talay, 2019; González et al., 2018) se habían estudiado las herramientas de procesamiento gráfico necesarias para analizar los resultados del simulador ns-2, al comenzar a utilizar el nuevo simulador, se advierte que otras herramientas gráficas tendrán que ser estudiadas para analizar los resultados ya que la mayoría de ellas tampoco son compatibles con ns-3.

Además, la imposibilidad de establecer una "traducción directa" de los modelos diseñados para el simulador ns-2 y deseen migrar al nuevo simulador, es un punto desfavorable, dado que se utilizan lenguajes completamente diferentes para su uso. Por ello, es necesario reescribir los códigos de todos los modelos y escenarios que funcionaban en ns-2 para que lo hagan también en ns-3.

De manera que la elección y el dominio de un simulador debe adaptarse a las necesidades planteadas por la tarea a realizar, y dependerá, en gran medida, de la experiencia del usuario con otros sistemas similares, el manejo de los lenguajes de programación, y el objetivo particular de du estudio.

Para ámbitos de investigación donde se aspira a publicar en revistas y congresos científicos, utilizar una herramienta que dejó de tener soporte hace casi diez años quita un poco de valor y credibilidad a los resultados del trabajo.

En cambio, si el objetivo es su uso meramente académico para entender el funcionamiento de los protocolos y estudiar ciertos comportamientos aislados, utilizar una u otra herramienta será indistinto. Si el estudiante aún no tiene conocimientos básicos de ninguna de ellas, se recomendará que invierta su tiempo y dedicación a aprender a utilizar el simulador más actual, que además de brindar mayor cobertura en tutoriales actuales y compatibilidad con todos los sistemas modernos, se espera que, siguiendo los pasos de su antecesor, se convierta en el nuevo simulador de facto, con una amplia comunidad activa de usuarios y artículos publicados.

Si el estudiante ya conoce el funcionamiento de ns-2, quiere continuar utilizándolo y puede manejarlo con relativa fluidez, podrá hacerlo sin demasiados inconvenientes. La comunidad de usuarios que pueden resolver dudas respecto de ns-2 y los modelos publicados para utilizar son amplios. Aprender el uso de una nueva herramienta es un proceso complejo y extenso, por lo que quizás volver a recorrer la pronunciada curva de aprendizaje que tienen estos simuladores no resulte en beneficios extra que continuar con el uso del antiguo simulador.

Si bien, como grupo de investigación, aún continuamos utilizando el simulador ns-2 como soporte de las publicaciones y trabajos actuales, nos encontramos en un camino de aprendizaje para descubrir todas las potencialidades del simulador ns-3. Entendemos que asimilar los conocimientos y habilidades que su uso requiere llevará tiempo, pero se verá compensado en nuevas posibilidades para la investigación y ahorrará esfuerzos a futuro.

ICT-UNPA-245-2020
ISSN: 1852-4516
Aprobado por Resolución N° 0441/20-R-UNPA
http://doi.org/10.22305/ict-unpa.v12.n2.732

8. RECOMENDACIONES

En el transcurso de una investigación surge la necesidad de realizar pruebas y observaciones, cuando examinar los hechos en la realidad no es posible porque resulta costoso y no es posible aislar todos los eventos involucrados, se puede recurrir a sistemas informáticos que simulen los eventos. Los simuladores de eventos discretos que se han analizado en este trabajo son dos versiones incompatibles entre sí, planteadas una como reemplazo de la otra.
Continuar investigando y estudiando las potencialidades del nuevo simulador es una tarea que llevará tiempo y dedicación por parte de este equipo de investigación, pero a mediano plazo permitirá ejecutar simulaciones que el simulador ns-2 ya no sea capaz de afrontar. Asimismo, realizar publicaciones con versiones actualizadas, nos permitirá comunicar con mayor confianza los resultados obtenidos y las conclusiones consecuentes. Como se ha mencionado, las herramientas de simulación pueden tener usos variados según el uso sea académico o de investigación.
Si bien, gran parte de los trabajos de este equipo han sido utilizando el simulador con el que tenemos más experiencia, debe destacarse la necesidad de migrar a un nuevo simulador, actualizado y robusto como ns-3. También, queda la posibilidad de optimizar trabajos anteriores incorporando el uso del nuevo simulador, por ejemplo, en el informe (Almada & Talay, 2019) se podría expandir la lista de herramientas y modificarla para adaptarse a las nuevas necesidades que plantea el uso de ns-3 en el ámbito de investigación de redes de computadoras.
Además, se pueden utilizar como guía los consejos dados en la discusión para decidir en qué caso sería más adecuado la elección de un simulador u otro. Con este objetivo, el marco de evaluación aquí planteado resultaría una herramienta de comparación y selección válida.

9. AGRADECIMIENTOS

Agradecemos a la Unidad Académica Río Gallegos, de la Universidad Nacional de la Patagonia Austral, por el financiamiento de este proyecto de investigación. Al Ing. Diego Rodríguez Herlein, la Lic. Claudia González, al Sr. Franco Trinidad y a la Sra. Marycarmen Díaz Labrador por la colaboración.

REFERENCIAS

AAVV. (2020a). *JSim*.
AAVV. (2020b). *omnetpp*. Recuperado de https://github.com/omnetpp/omnetpp
ALMADA M., & TALAY C. (2019). Procesamiento gráfico de datos obtenidos en la simulación de redes. Análisis comparativo de herramientas como complemento del simulador NS-2. *Informes Científicos Técnicos - UNPA, 11,* 1-14. https://doi.org/10.22305/ict-unpa.v11.n3.794
ALMAZÁN D. B. (2014). *Modelado de la norma 802.11n en ns-3* (Universidad de Sevilla). Recuperado de https://bit.ly/32bW8np
AVNEET KAUR, SALUJA, SWETA A, DARGAD, & KRUPALI, M. (2017). A Detailed Analogy of Network Simulators – NS1 , NS2 , NS3 and NS4. *International Journal on Future Revolution in Computer Science & Communication Engineering*, *3*(12), 291-295.

CALLE M. A., TOVAR J. D., CASTAÑO-PINO Y. J., & CUÉLLAR J. C. (2018). Comparación de parámetros para una selección apropiada de herramientas de simulación de redes. *Información Tecnológica*, *29*(6), 253-266. https://doi.org/10.4067/S0718-07642018000600253

COMBS G., & COL. (2020). *Wireshark*. Recuperado de https://www.wireshark.org/

DIAKOPOULOUS N. (2018). Interactive: The Top Programming Languages 2018. Recuperado de IEEE Spectrum website: https://bit.ly/308nJ6j

ESTERHUIZEN A., & AES K. (2012). TCP Congestion Control Comparison. *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*. Recuperado de https://bit.ly/2SlFDyf

FAN C., BI J., ZHOU Y., ZHANG C., & YU H. (2017). NS4: A P4-driven network simulator. *SIGCOMM Posters and Demos 2017 - Proceedings of the 2017 SIGCOMM Posters and Demos, Part of SIGCOMM 2017*, 105-107. https://doi.org/10.1145/3123878.3132002

FIB/UPC. (2008). La simulación por ordenador. *Retroinformática. El pasado del futuro*. Recuperado de https://www.fib.upc.edu/retro-informatica/avui/simulacio.html

GONZÁLEZ C. N., RODRÍGUEZ H. D. R., TALAY C. A., TRINIDAD F. A., & ALMADA M. L. (2019). Análisis de rendimiento y equidad en TCP. *XXI Workshop de Investigadores en Ciencias de la Computación (WICC 2019)*, 106-109. San Juan: Editorial UNSJ.

GONZÁLEZ C. N., TRINIDAD F. A., ALMADA M. L., TALAY C. A., RODRÍGUEZ H. D. R., & MARRONE L. A. (2018). Herramientas para el procesamiento gráfico de datos en la simulación de redes. *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología (TE&ET)*.

GROSS J., WEHRLE K., & GÜNEŞ M. (2010). *Modeling and Tools for Network Simulation*. https://doi.org/10.1007/978-3-642-12331-3

JACOBSON V. (1988). Congestion avoidance and control. *ACM SIGCOMM Computer Communication Review*, *18*(4), 314-329. https://doi.org/10.1145/52325.52356

JACOBSON V., LERES C., MCCANNE S., & LAWRENCE BERKELEY NATIONAL Laboratory, UNIVERSITY OF CALIFORNIA, BERKELEY, C. (2019). *tcpdump - dump traffic on a network*. Recuperado de https://www.tcpdump.org/

KESHAV S. (1988). *REAL: A Network Simulator*. Recuperado de https://dl.acm.org/doi/book/10.5555/894210

LACAGE M. (2009). *nam (ns-2 network animator) for ns-3*. Recuperado de http://www.nsnam.org/contributed/ns-3-nam.tar.bz2

MARQUES E. M. D., PLÁCIDO R. A. S. A., & SAMPAIO P. N. M. (2009). Visual Network Simulator (VNS): A GUI to QoS simulation for the ns-2 simulator. *2009 IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2009*, 342-349. https://doi.org/10.1109/AICCSA.2009.5069346

MCCANNE S., FLOYD S., & FALL K. (2009). LBNL's Network Research Group. Recuperado de https://ee.lbl.gov/ns/

MICROSOFT. (2019). *WSL2*. Recuperado de https://docs.microsoft.com/es-es/windows/wsl/wsl2-index

NIKOUKARAN J., HLUPIC V., & PAUL R. J. (1998). Criteria for simulation software evaluation. *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)*, *1*, 399-406 vol.1. https://doi.org/10.1109/WSC.1998.745014

NSNAM. (2020). ns-3.31 released. Recuperado de https://bit.ly/2WgpI7n

PYVIZ. (s. f.). Recuperado de https://www.nsnam.org/wiki/PyViz

RILEY G. F. (2017). *NetAnim*. Recuperado de http://code.nsnam.org/netanim

RIVERBED. (2020). Riverbed. Recuperado de https://www.riverbed.com/mx/products/steelcentral/steelcentral-riverbed-modeler.html#

RODRÍGUEZ PÉREZ R. (2012). *Interfaz gráfica de usuario para la simulación y enseñanza de Sistemas de cola mediante ns-2* (Universidad de Cantabria; Vol. 2). Recuperado de https://repositorio.unican.es/xmlui/bitstream/handle/10902/1061/348234.pdf

TANENBAUM A. S. (2003). *Redes de computadoras* (4ta ed.). Recuperado de https://books.google.es/books?id=WWD-4oF9hjEC

TORRES J. A., FIGUEROA D. A., & DÍAZ J. (2015). *Herramientas de software de simulación para redes de Comunicaciones*. Recuperado de https://bit.ly/2SoxF7j

TRINIDAD F., & TALAY C. (2019). Consideraciones metodológicas para la investigación con simuladores de red. *Informes Científicos Técnicos - UNPA*, *11*(3), 211-235. https://doi.org/10.22305/ict-unpa.v11.n3.803

USC/ISI. (2011). ns: Change History. Recuperado de The Network Simulator website: https://www.isi.edu/nsnam/ns/CHANGES.html

VELÁSQUEZ K., & GAMESS E. (2009). Análisis comparativo de herramientas de evaluación de desempeño en redes de computadores. En E. de Computación (Ed.), *Lecturas en Ciencias de la Computación* (1.ª ed.). Caracas: Universidad Central de Venezuela.

VINSCHEN C., TURNEY J., & BLAKE E. (2020). *Cygwin*. Recuperado de https://www.cygwin.com/

WEINGÄRTNER E., VOM LEHN H., & WEHRLE K. (2009). A performance comparison of recent network simulators. *2009 IEEE International Conference on Communications*, 1-5. https://doi.org/10.1109/ICC.2009.5198657