Cienc Tecn UTEQ (2020) 13(1) p 89-96 ISSN 1390-4051; e-ISSN 1390-4043 https://doi.org/10.18779/cyt.v13i1.356

FNCS: Propuesta de una plataforma de gestión de dispositivos de red basados en RouterOS

FNCS: Proposal of a Platform for the management of network devices based on RouterOS

Núñez-Agurto Daniel[1], Benavides-Astudillo Eduardo[1], Salazar Armijos Diego Ricardo[1], Milton Temístocles Andrade Salazar[1]

[1]*Universidad de las Fuerzas Armadas ESPE, adnunez1@espe.edu.ec, debenavides@espe.edu.ec*

Rec.: 19.03.2019. Acept.: 21.1.2019.

Publicado el 30 de juio de 2020

Resumen

En las organizaciones, contar con la disponibilidad de los servicios de red inalámbrica es imprescindible. Sin embargo, esta disponibilidad se ve afectada debido principalmente a dos factores: la necesidad de un experto en la configuración de los equipos y al desplazamiento de dicho experto al sitio, para que este configure cada uno de los equipos que necesitan dicha configuración. Por otra parte, la empresa Mirotik dispone de un Sistema Operativo (SO) para equipos Routerboard, denominado RouterOS (RouterBoard Operating System). Este SO soporta la comunicación mediante los protocolos telnet, ssh, pero las aplicaciones que utilizan estos protocolos por lo general no tienen una interfaz gráfica, además existen herramientas propietarias gráficas como winbox y webfig para configurar estos equipos, pero esto trae consigo otro tipo de problemas como son: configuración manual de equipo en equipo, mayor uso del CPU de los dispositivos, y consumo de más ancho de banda para poder gestionarlos. Para solucionar estos problemas, ha sido desarrollado una aplicación con interfaz web, denominada Fast Network Config System (FNCS), la cual permite gestionar y configurar de forma gráfica, remota y centralizada los dispositivos con RouterOS, de manera mucho más eficiente que las soluciones disponibles. Para desarrollar la aplicación, se aprovechó la API-Mikrotik que está integrada en RouterOS. Posteriormente, se comparó la velocidad en la configuración de la herramienta propuesta FNCS versus Winbox y ssh, y se comprobó que la solución FNCS es hasta 200 veces más rápida.

Palabras clave*:* Mikrotik, RouterOS, API-Mikrotik, virtualización.

Abstract

In organizations, having the availability of network services is essential. However, this availability is affected mainly due to two factors: the need for an expert in the configuration of the equipment and the movement of this expert to the site, so that it configures each of the devices that need such configuration. On the other hand, Mirotik has an Operating System (OS) for Routerboard equipment, called RouterOS (RouterBoard Operating System). This OS supports communication through the protocols telnet, ssh, but the applications that use these protocols generally do not have a graphical interface, there are also proprietary graphic tools such as winbox and webfig to configure these devices, but this brings with it other problems such as: manual configuration of device in device, greater use of the CPU of the devices, and consumption of more bandwidth to manage them. To solve these problems, has been developed an application with web interface, called Fast Network Config System (FNCS), which allows to manage and configure graphically, remotely and centrally the devices with RouterOS, much more efficiently than the solutions available. To develop the application, we took advantage of the API-Mikrotik that is integrated in RouterOS. Subsequently, the speed was compared in the configuration of the proposed tool FNCS versus winbox and ssh, and it was found that the solution FNCS is up to 200 times faster.

keywords: Mikrotik, RouterOS, API-Mikrotik, virtualization.

Introducción

En las organizaciones, contar con la disponibilidad de los servicios de red es de suma importancia, esto implica que las organizaciones destinen gran parte del personal de tecnologías de la información (TI) a resolver problemas de infraestructura, sobre todo a nivel de acceso. El personal que se dedica a este tipo de soporte, debe tener un alto grado de habilidades en la gestión de dispositivos de red. En la mayoría de los casos, las configuraciones se vuelven complejas, debido a la cantidad de instrucciones que se ejecutan en los dispositivos, provocando que en ocasiones se cometan errores en las configuraciones y, por lo tanto, el detectarlos y corregirlos puede llevar más tiempo que la misma configuración.

La empresa Mikrotik, ha desarrollado una placa base llamada Routerboard, esta placa base permite desarrollar diferentes equipos routers, access point (AP), switchs con la integración interfaces ethernet y wireless, ademas integra RouterOS (Mikrotik, 2015). Existen herramientas desarrolladas por la empresa mikrotik y herramientas tradicionales, que permiten la configuración de equipos con RouterOS. Entre las herramientas propietarias tenemos winbox, que tiene una interfaz sencilla para realizar la administración y webfig, una herramienta basada en la web, por lo tanto, no necesita ninguna instalación (Mikrotik, 2018a). Existen otras herramientas tradicionales que RouterOS admite en su configuración como telnet y ssh, pero que no tienen una interfaz gráfica para realizar la gestión de su SO (Galbraith, Dyke, & Bright, 2007). Si bien las herramientas propietarias permiten la configuración visual de los dispositivos, esto trae consigo otro tipo de problemas como son: configuración manual de equipo en equipo, mayor uso del CPU de los dispositivos, y más capacidad de ancho de banda para gestionarlos remotamente. Sin embargo, RouterOS ofrece una alternativa que permite desarrollar aplicaciones a medida del usuario para su administración y control; la cual se conoce como API-Mikrotik (Mikrotik, 2018b). Esta alternativa, permite que las organizaciones integren aplicaciones a medida, como un módulo adicional en sus aplicaciones en la mesa de soporte.

El principal objetivo del presente trabajo es agilizar la gestión de dispositivos RouterOS, mediante el diseño e implementación de una aplicación, que permita la gestión centralizada de dispositivos, aprovechando la capacidad de su comunicación a través de la API-Mikrotik (Application Programmable Interface). La aplicación desarrollada se denominada FNCS (Network Config System), y permite realizar la configuración y auditoria de dispositivos basados en RouterOS de forma gráfica, remota y centralizada. El resto de este documento se ha organizado de la siguiente manera: En la Sección 2, se explica la metodología, en la Sección 3 se realiza una descripción de la aplicación y su funcionamiento después de haberla probado en un entorno virtual, en la Sección 4, se muestran los resultados de rendimiento del sistema analizados por los autores y finalmente, en la Sección 5 se exponen las conclusiones del trabajo, así como trabajos futuros.

En cuanto a trabajos relacionados con la gestión de dispositivos con RouterOS, podemos mencionar que He & Cao (2010) proponen la implementación de un servidor PPPoE basado en RouterOS, abandonando las vias convencionales para construir de alto rendimiento, así como la dependencia del router con hardware de gama alta, pudiendo ser implementadas en grandes y medianas empresas. En el trabajo de los autores (Cheng, Wu, Routeros, Security, & Virus, 2010), se hacen pruebas implementando PPPoE (Point-to-Point Protocol over Ethernet) sobre RouterOS, con el objetivo de controlar los problemas de seguridad inmersos en las gestión de la red, la implementación la realizan el campus del Instituto de Huangshi.

Cueva, Pozo, & Iturralde (2017) desarrollan e implementan un programa multiplataforma, llamado ENDS (Easy Network Designer Software), que permite realizar la virtualización de la red y la configuración remota de los parámetros básicos del equipo MikroTik con RouterOS, a través de una interfaz gráfica. El software fue escrito en Java y utiliza una base de datos MySQL donde se guardan los proyectos.

Por su parte, Dolnák & Litvik (2016), realizan pruebas complejas de conmutación por error y balanceo de carga en routers MikroTik con RouterOS, basados en estándares y utilizando la tecnología OpenVPN. Iswadi, Adriman, & Munadi (2019) se centran en los algoritmos de gestión de ancho de banda PCQ - HTB de conmutación adaptativa con RouterOS y analizan las debilidades y ventajas de estos dos algoritmos, para proponer mecanismos de conmutación adaptativos, con el propósito de organizar el uso del ancho de banda.

Materiales y métodos

Sistema operativo RouterOS

Una de las características importantes de RouterOS, es que permite virtualizar una PC como si fuese un router, es decir con todas sus características; firewall, access point, bandwidth managment, hotspot Gateway y servidor VPN. Además, dispone de algunos puertos y protocolos, que son utilizados por determinados servicios. En el Cuadro 1 se observa el listado de los servicios con sus respectivos puertos y protocolos (Mikrotik, 2018c).

Cuadro 1. Servicios RouterOS

Nombre	Puerto
telnet	23
ftp	21
www	80
ssh	22
www-ssl	443
api	8728
WinBox	8291

De los servicios disponibles para la administración de RouterOS, se escogió la API. Esto, debido a que permite desarrollar aplicaciones a medida, y sobre todo, porque se pueden integrar al software de la mesa de soporte.

Infraestructura de virtualización

Los objetos que componen la infraestructura de virtualización, son los servidores y las estaciones de trabajo que se implementan en forma de máquinas virtuales y en las que posteriormente se instalan SO (Fomin, 2017). De manera general se puede indicar que, para desarrollar cualquier trabajo práctico o para llevar a cabo un experimento, es recomendable realizarlo bajo un ambiente controlado, es decir con un sistema de virtualización.

En Horalek, Matyska, & Sobeslav, 2013, se recomienda utilizar un entorno con VirtualBox si el propósito es migrar a otro entorno de host, además, VirtualBox en su versión open source dispone de características para desplegar un entorno virtual sobre anfitrión. Las principales características que tiene Virtual Box son: clonación de sistemas operativos, permite ejecutar múltiples sistemas operativos simultáneamente. Estas características resultan útiles para la instalación del sistema operativo RouterOS y desarrollar el entorno de experimentación que se busca (ORACLE, 2018).

API Mikrotik

La API-MikroTik permite desarrollar aplicaciones para comunicarse con RouterOS con el propósito de: recopilar información, ajustar la configuración y administrar los equipos que tengan este SO. Por defecto, este servicio se encuentra deshabilitado dentro de la configuración, el puerto que utiliza para establecer la comunicación es el tcp 8798. El protocolo de comunicación que utiliza la API, debe contener el comando como una primera palabra seguida de palabras sin ningún orden en particular, el final de la oración debe estar marcada por una palabra de longitud cero, de manera que, cuando el dispositivo recibe la sentencia completa, esta es evaluada y ejecutada, después se forma y envía una respuesta. En el Cuadro 2, podemos observar codificación de palabras según el protocolo descrito (Mikrotik, 2018a).

Funcionamiento de la aplicación

La aplicación instalada en un servidor web, permite que los usuarios se conecten remotamente con sus respectivas credenciales y puedan acceder a las diferentes funcionalidades de la aplicación: administración, configuración de nuevos dispositivos en la red, auditoria de la configuración y estado de los equipos.

Para gestionar un dispositivo de la red, se debe ingresar en la base de datos de la aplicación, la información: nombre, ubicación, ip-wan, subred-lan. Luego, cuando la información se encuentre en la base de datos, podremos ejecutar la configuración, validación y auditoria de los dispositivos. Con esta información, se estructuran los comandos necesarios para ser enviados mediante la conexión API. En la Figura 1 se muestra el entorno de red virtual y el servidor web en el que está alojada la aplicación FNCS. Además, por medio del entorno virtual, se pueden simular varios dispositivos con RouterOS.

Arquitectura de la aplicación

Al momento de realizar la aplicación FNCS se diseñaron dos elementos: El primer elemento es la aplicación web, encargada de registrar y gestionar la información en la base de datos. El segundo elemento es la aplicación web que interactúa mediante la API-Mikrotik con los dispositivos de red.

En la Figura 2 se puede observar la arquitectura de la aplicación desarrollada. Para gestionar la conexión

Cuadro 2. Codificación de palabras en la API Mikrotik

Value of length	#of bytes	Encoding
0 <= len <= 0x7F	1	len, lowest byte
0x80 <= len <= 0x3FFF	2	len | 0x8000, two lower bytes
0x4000 <= len <= 0x1FFFFF	3	len | 0xC00000, three lower bytes
0x200000 len <= 0xFFFFFFF	4	len | 0xE0000000
len >= 0x10000000	5	0xF0 and len as four bytes

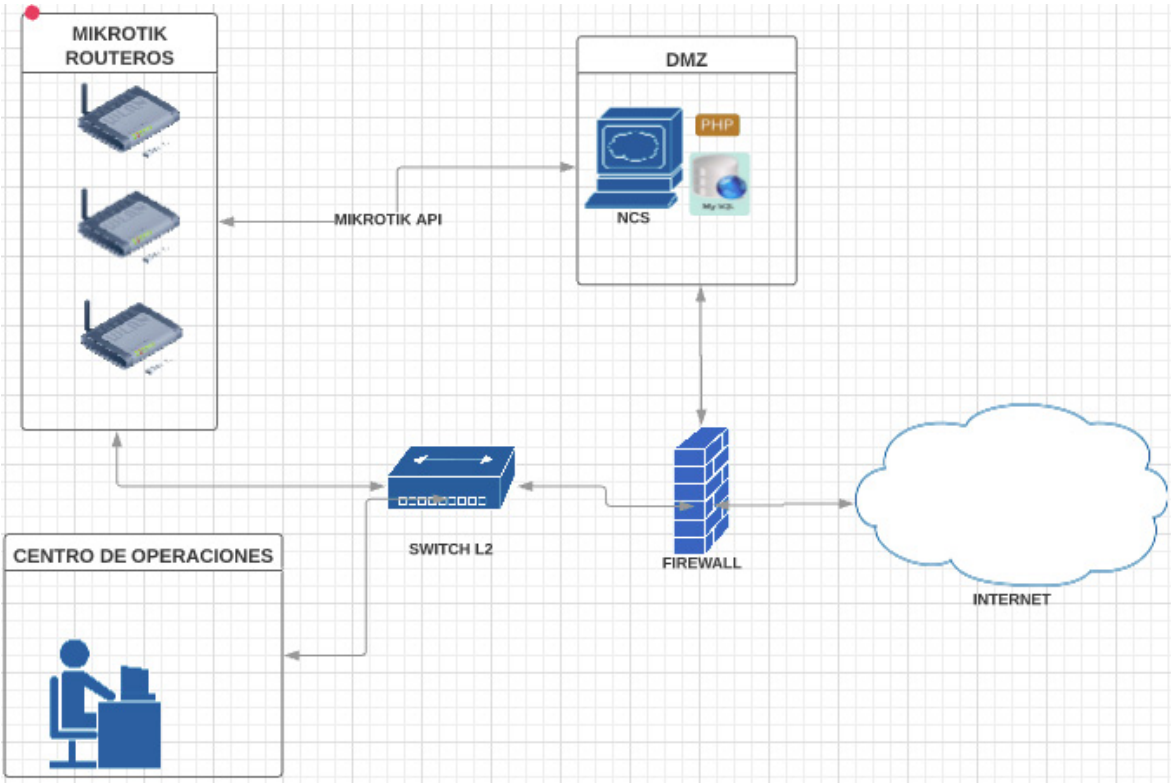

Figura 1. Entorno de red virtual

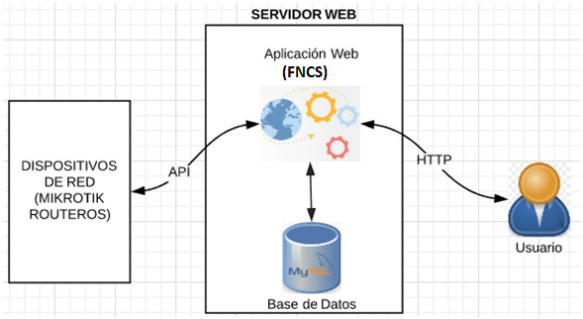

Figura 2. Arquitectura de FNCS.

con la API Mikrotik, se utilizó una librería desarrollada por (Barnes, 2015) en PHP, que permite enviar y recibir comandos a dispositivos que tienen RouterOS.

Plataforma de Desarrollo de la Aplicación

Se escogieron dos herramientas debido a que según Axmark & Widenius (2018) y PHP Group (2017), tienen la posibilidad de adaptar nuevos requerimientos de manera dinámica. Además, se utilizó la metodología XP (Extreme Programming), que permite concentrarse especialmente en el código de la aplicación (Bougroun, Zeaaraoui, & Bouchentouf, 2014). Por las razones antes descritas, la aplicación web fue desarrollada con el lenguaje de programación PHP y la permanencia de los datos de los dispositivos de red, se almacenan en una base de datos MySQL. El tiempo estimado para el desarrollo de este estudio fue de tres meses, divididos en tres fases: La primera fase, consistió en analizar el funcionamiento de la API-Mikrotik y como interactuar con los dispositivos en un entorno virtual con RouterOS, la segunda fase radicó en realizar el modelo de la base de datos en MySQL-workbech (Oracle Corporation, 2018), la tercera fase consistió en desarrollar la aplicación. En esta tercera fase, se utilizó la metodología Test-Driven Development (TDD) y además, se utilizó una herramienta denominada cypres (Mann & Kent, 2018), la cual realiza un test de front-end del código (Zhai, Hu, Tang, Ma, & Chen, 2014).

(Fast Network Config System)

Parámetros de configuración

Cuando se inicia la aplicación, se despliega una pantalla de inicio de sesión, donde se pedirá que se ingrese usuario y clave del sistema. En la Figura 3, se puede observar la pantalla de inicio de sesión.

Para poder gestionar los dispositivos de red, la aplicación pide que se registre el campus donde se encuentran instalados físicamente los dispositivos. Se puede considerar como un campus, una agrupación o un lugar en el cual los dispositivos RouterOS, comparten una ubicación física. En la Figura 4, se puede observar el formulario para registro del campus. Luego, es

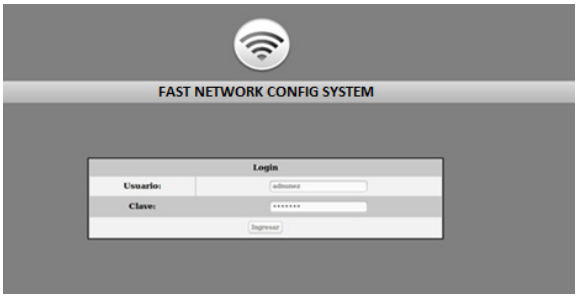

Figura 3. Pantalla de inicio de sesión

Figura 4. Registro de campus

necesario ingresar la información general de los dispositivos de la red, como se puede observar en la Figura5. Esta información es muy importante, porque será consultada posteriormente para construir los comandos de las configuraciones y, además, permitirá obtener información del dispositivo RouterOS, como la IP de administración, con la finalidad de poder gestionarlo remotamente.

Configuración remota

En la Figura 6, se puede observar el proceso de configuración remota de un dispositivo con RouterOS desde la aplicación. Primero se selecciona el campus al que debe pertenecer el dispositivo, luego, se selecciona el dispositivo que se desea configurar y, posteriormente se presiona Ejecutar. Una vez ejecutado el proceso, aparece en la parte inferior de la pantalla el resultado de la ejecución de configuración, junto con las instancias que se configuraron; el algoritmo de este proceso se lo puede revisar en el Cuadro 3.

Validación remota de los dispositivos

Es importante que un gestor de equipos de red, valide que los dispositivos se encuentren funcionando en condiciones normales, tal y como lo hiciera un técnico experto en el área, por esta razón, FNCS analiza los principales indicadores de los dispositivos. En la Figura 7, podemos observar el estado físico de la interfaz ethernet, del firmware y el software de un dispositivo; el algoritmo de este proceso se lo puede observar en el Cuadro 4.

Auditoria remota de los dispositivos

Cuando se gestiona una red, y sobre todo cuando esta red es grande, es necesario conocer si los dispositivos se encuentran actualizados con la última versión de firmware, software y conocer el uptime. Si esta actividad, fuese realizada de forma manual por una persona con conocimientos en el area, tardaría horas y posiblemente habría inconsistencias en la información recopilada. Con FNCS, el poder auditar la red de un campus puede resultar sumamente fácil y en cuestión de unos pocos segundos, tendremos la información de toda la red. En la Figura 8, se observa la información de la auditoria automática realizada en los dispositivos simulados de un campus y en el Cuadro 5 se muestra el algoritmo que realiza es proceso de auditoría.

Pruebas de rendimiento y resultados

Para realizar las pruebas de validación, se configuraron diez dispositivos con RouterOS en un entorno virtual. Luego se ejecutaron tres actividades comunes en la gestión de una red. Estas son: configuración, verificación, y auditoria de los dispositivos. Estas tareas se realizaron con tres herramientas diferentes con la finalidad de poder compararlas. Primero, con la aplicación propuesta FNCS, luego con una herramienta propietaria WinBox, y por último con una la herramienta tradicional ssh. La finalidad de estas pruebas fue el conocer si había diferencias categóricas en su funcionamiento.

Figura 5. Ingreso de dispositivos a la aplicación

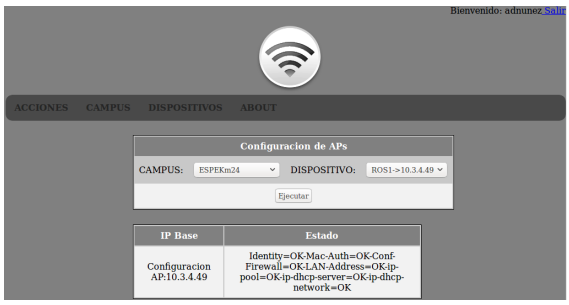

Figura 6. Ejecución de la configuración

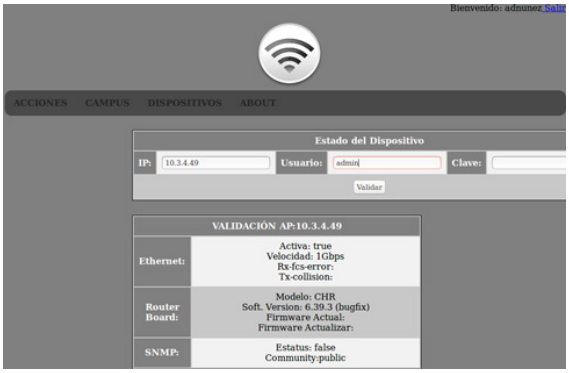

Figura 7. Validación de estado del dispositivo

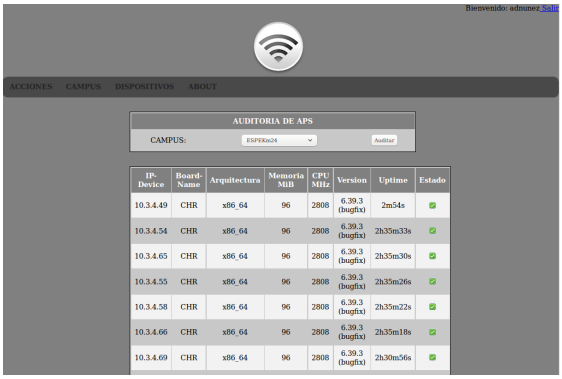

Figura 8. Auditoria de los dispositivos

Cuadro 3. Algoritmo de configuración de dispositivos RouterOS

Algorithm 1 Configuración dispositivos RouterOS

Require: Class RouterOS Variable $API = newrouterosapi.class()$;
Require: Connection Variable Mysql $conn = mysql - connection()$;
Require: Connection Variable dat;
Require: Config Name $conf - name$; bool
Require: Config Ethernet $conf - eth$; bool
Require: Config DHCP $conf - dhcp$; bool
Require: Config Firewall $conf - fw$; bool
Validar conexion a la base de datos
if $!conn$ **then**
  Realizar consulta a la base de datos y asignar a variable dat
  if $API- > connect(dat['ip','routerosuser','routerospasswd','routerosport'])$ **then**
    Configura Nombre Dispositivo $router - name = config - name()$
    Configura Interfaces Ethernet $conf - eth = config - ethernet()$
    Configura DHCP $conf - dhcp = config - dhcp()$
    Configura Firewall $conf - fw = config - firewall()$
    Cerrar conexión $API- > disconnect()$
    if $config - name$ $config - eth$ AND $conf - dhcp$ AND $conf - fw$ **then**
      Dispositivo configurado correctamente.
    else
      No se configuró el dispositivo correctamente.
    end if
  else
    No se puede realizar la conexión
  end if
end if

Cuadro 5. Algoritmo de auditoria de dispositivos RouterOS

Algorithm 3 Auditoria de dispositivos RouterOS

Require: Class RouterOS Variable $API = newrouterosapi.class()$;
Require: Connection Variable Mysql $conn = mysql - connection()$;
Require: Connection Variable dat;
Require: IP Address ip; string
Require: Device Name $dname$; string
Require: Hardware Arquitecture $arquitecture$; string
Require: Hardware Memory $meminfo$; string
Require: CPU Frecuency $cpufreq$; string
Require: Device Uptime $uptime$; string
Require: Device Status $status$; bool
Validar conexion a la base de datos
if $!conn$ **then**
  Realizar consulta a la base de datos y asignar a variable dat
  while dat **do**
    if $API- > connect(dat['ip','routerosuser','routerospasswd','routerosport']$ **then**
      Asignar dirección IP dispositivo $ip = dat['ip']$
      Consultar nombre dispositivo $dname = show - name()$
      Consultar aquitectura $arquitecture = show - arquitectura()$
      Consultar memoria dispositivo $meminfo = show - meminfo()$
      Consultar frecuencia CPU dispositivo $cpuinfo = show - cpuinfo()$
      Consultar software dispositivo $cpufreq = show - cpufreq()$
      Consultar uptimee dispositivo $uptime = show - uptime()$
      Validar resultados del dispositivo y asignar a $estado$
      Cerrar conexión $API- > disconnect()$
      if $estado$ **then**
        Funcionando correctamente:
      else
        Advertencia revisar dispositivo.
      end if
    else
      No se puede realizar la conexión
    end if
  end while
end if

Cuadro 4. Algoritmo de validación de dispositivos RouterOS

Algorithm 2 Valida estado dispositivos RouterOS

Require: Class RouterOS Variable $API = newrouterosapi.class()$;
Require: Connection Variable Mysql $conn = mysql - connection()$;
Require: Connection Variable dat;
Require: Validate Ethernet $val - eth$; bool
Require: Validate routerBoard $val - rb$; bool
Validar conexion a la base de datos
if $!conn$ **then**
  Realizar consulta a la base de datos y asignar a variable dat
  if $API- > connect(dat['ip','routerosuser','routerospasswd','routerosport'])$ **then**
    Valida estado interfaz Ethernet $val - eth = config - ethernet()$
    Valida Configuración DHCP $val - dhcp = config - dhcp()$
    Validad Configuración Firewall $conf - fw = config - firewall()$
    Cerrar conexión $API- > disconnect()$
    if $config - eth$ AND $conf - dhcp$ AND $conf - fw$ **then**
      Dipositivo no presenta novedades:
    else
      Dipositivo presenta novedades.
    end if
  else
    No se puede realizar la conexión
  end if
end if

Pruebas en la configuración de dispositivos RouterOS

En la Figura 9, se pueden apreciar que los resultados de la configuración de los dispositivos con la aplicación FNCS fueron en promedio de 0.34 segundos, en WinBox de 151.4 segundos y en ssh de 220 segundos.

Pruebas de validación del estado de dispositivos RouterOS

Cuando analizamos el estado de los dispositivos con las tres herramientas se obtuvieron los siguientes resultados en promedio: con la aplicación FNCS 0.34 segundos, con WinBox 21.4 segundos y con ssh de 17.1. Estos resultados los podemos observar en la Figura 10.

Pruebas de auditoría en los dispositivos

Se debe considerar que, para realizar esta prueba, nuestra aplicación propuesta tiene una gran ventaja, que es el poder ejecutar la auditoría prácticamente simultáneamente en los diez dispositivos, lo cual se pudo realizar en únicamente 0.26 segundos, mientras que, en los otros dos programas evaluados, no se pudieron realizar auditorías simultaneas, sino que hubo que realizarlas en equipo por equipo. Así, WinBox realizó la ejecución en 214.4 segundos y ssh en 160.9 segundos en promedio. Estos resultados se pueden apreciar en la Figura 11.

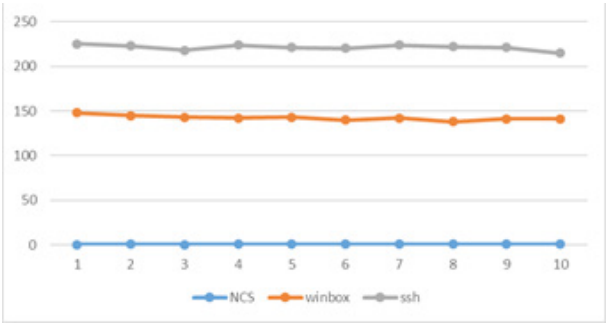

Figura 9. Tiempo de configuración

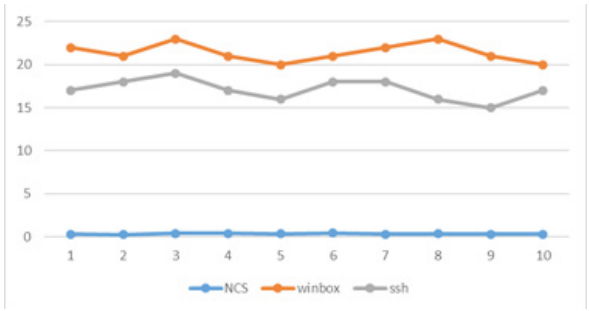

Figura 10. Tiempo de validación

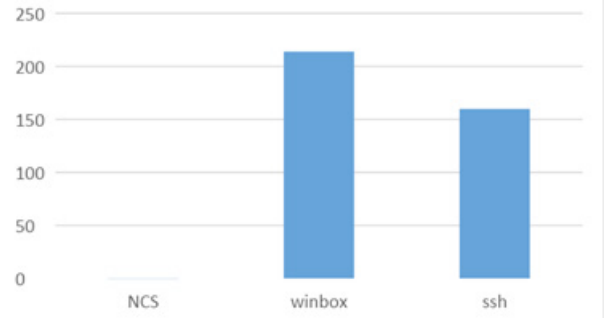

Figura 11. Tiempo de ejecución de auditoria

Conclusiones

El objetivo de la presente investigación fue el de desarrollar una aplicación web que permita gestionar los dispositivos de red en una organización, de una manera eficiente, a través de una herramienta de comunicación diferente a las propietarias o convencionales, pero con mejores características de rendimiento en su velocidad y manejo.

Se comprobó cuantitativamente que, el tiempo que toma el realizar actividades convencionales de administración de dispositivos de red con la aplicación FNCS, fue extremadamente bajo, en comparación con un técnico experto utilizando herramientas propietarias como WinBox y herramientas tradicionales como ssh.

FNCS resultó muy fácil de usar y, sobre todo, no se necesita contar con un técnico experto para ejecutar las diferentes tareas. Además, la permanencia de la información de los dispositivos, contribuye al desarrollo de las líneas de comandos de forma dinámica, permitiendo que el tiempo de ejecución se reduzca.

Un beneficio destacado en esta investigación, fue el haber podido desarrollar una aplicación a medida, que permita gestionar dispositivos con RouterOS mediante el uso de una API del fabricante. Estos resultados pueden ser utilizados como una línea base que pueda ser integrada en los sistemas de gestión de red de las organizaciones que utilicen dispositivos con RouterOS.

Un potencial estudio futuro podría centrarse en el desarrollo de una aplicación que analice el estado de las conexiones de los usuarios y que permita tomar decisiones en tiempo real en su conexión, para mejorar la calidad de servicio.

Bibliografía

Axmark, D. ., & Widenius, M. (2018). MySQL : MySQL 5.7 Reference Manual. Retrieved February 18, 2018, from https://dev.mysql.com/doc/refman/5.7/en/

Barnes, N. (2015). API PHP class - MikroTik Wiki. Retrieved February 18, 2018, from https://wiki.mikrotik.com/wiki/API_PHP_class

Bougroun, Z., Zeaaraoui, A., & Bouchentouf, T. (2014). The projection of the specific practices of the third level of CMMI model in agile methods: Scrum, XP and Kanban. In *2014 Third IEEE International Colloquium in Information Science and Technology (CIST)* (pp. 174–179). IEEE. https://doi.org/10.1109/CIST.2014.7016614

Cheng, J., Wu, H., Routeros, K., Security, N., & Virus, A. R. P. (2010). The Application Of The PPPOE For Network Security Management Using RouterOS,(Iccda), 569–571.

Cueva, H., Pozo, F., & Iturralde, D. (2017). Cross-platform network visualization software for MikroTik devices. *Proceedings of the 2016 IEEE ANDESCON, ANDESCON 2016*. https://doi.org/10.1109/ANDESCON.2016.7836222

D. Iswadi, R. Adriman and R. Munadi. (2019). "Adaptive Switching PCQ-HTB Algorithms for Bandwidth Management in RouterOS". IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), Banda Aceh, Indonesia, pp. 61-65.

Fomin, S. S. (2017). Technology and Practice of ICT Disciplines Maintenance, 193–195.

Galbraith, J., Dyke, J. Van, & Bright, J. (2007). Secure Shell Public Key Subsystem. https://doi.org/10.17487/RFC4819

He, J. H., & Cao, Y. (2010). Research and analysis the PPPoE server technology based on the routeros. *ICCASM 2010 - 2010 International Conference on Computer Application and System Modeling, Proceedings*, *15*(Iccasm), 209–

211. https://doi.org/10.1109/ICCASM.2010.5622586

Horalek, J., Matyska, J., & Sobeslav, V. (2013). Performance Comparison of Selected Virtualization Platforms, 327–332.

I. Dolnák and J. Litvik. (2016). "Failover and load balancing solutions for remote access VPNs based on open standards,". International Conference on Emerging eLearning Technologies and Applications (ICETA), Vysoke Tatry, 2016, pp. 55-58.

Mann, B., & Kent, R. (2018). About Cypress.io | JavaScript e2e Testing. Retrieved March 30, 2018, from https://www.cypress.io

Mikrotik. (2015). Manual:System/SSH client - MikroTik Wiki. Retrieved June 4, 2018, from https://wiki.mikrotik.com/wiki/Manual:System/SSH_client

Mikrotik. (2018a). Manual:Winbox - MikroTik Wiki. Retrieved May 28, 2018, from https://wiki.mikrotik.com/wiki/Manual:Winbox

Mikrotik. (2018b). Manual:API - MikroTik Wiki. Retrieved March 30, 2018, from https://wiki.mikrotik.com/wiki/Manual:API

Mikrotik. (2018c). RouterOS. Retrieved February 18, 2018, from https://wiki.mikrotik.com/wiki/Manual:TOC

ORACLE. (2018). Oracle VM VirtualBox. Retrieved May 28, 2018, from https://www.virtualbox.org/

Oracle Corporation. (2018). MySQL :: MySQL Workbench. Retrieved March 30, 2018, from https://www.mysql.com/products/workbench/

PHP Group. (2017). PHP: Choosing an API - Manual. Retrieved February 18, 2018, from http://php.net/manual/en/mysqlinfo.api.choosing.php

Zhai, J., Hu, J., Tang, X., Ma, X., & Chen, W. (2014). CYPRESS: Combining Static and Dynamic Analysis for Top-Down Communication Trace Compression. *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*, *2015–Janua*(January), 143–153. https://doi.org/10.1109/SC.2014.17