ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Extracción de reportes automatizados de consumo de recursos de red

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Debbie Daniella Donoso Calle

Jefferson Ivan Vega Sarango

GUAYAQUIL - ECUADOR

Año: 2022

DEDICATORIA

DEBBIE DONOSO

El presente proyecto lo dedico principalmente a Dios que me dio la sabiduría para poder encontrar las respuestas oportunas a los inconvenientes que se presentaron en el camino. Así también, lo dedico a mis padres Luisa Calle y May Donoso, quienes fueron pilar fundamental en la elaboración de este proyecto, pues su apoyo ha sido incondicional, de la misma manera lo dedico a mi novio Joel Blacio quien me acompañó en todo el proceso y apoyó incondicionalmente.

JEFFERSON VEGA

El presente proyecto lo dedico a Dios, mi familia, a mi papá Marcelo Vega y mi mamá Martha Sarango que brindaron todo su apoyo incondicional y confianza para que cumpla mis objetivos.

AGRADECIMIENTOS

DEBBIE DONOSO

Mi más sincero agradecimiento al Ing. Steven Santillán y al Ing. Rayner Durango por su guía en el proceso de este proyecto. De la misma manera, un agradecimiento para el Ing. Washington Velásquez y al Dr. José Córdova quienes nos acompañaron en todo el camino. Adicional, un agradecimiento especial para mis padres, mi familia y novio por su apoyo incondicional durante todos los años de carrera y amanecidas, por lo cual pude llegar a culminar con éxito mis estudios.

JEFFERSON VEGA

Mi más sincero agradecimiento a todos los compañeros que formaron parte de este proceso. Mención especial a los ingenieros que brindaron su guía y soporte en el desarrollo del proyecto. A mi amiga Debbie Donoso por su apoyo incondicional durante todo el trabajo. A mi familia por todo lo brindado desde el inicio en la carrera, a mi mascota por acompañarme en largas jornadas de estudio.

DECLARACIÓN EXPRESA

"Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Debbie Daniella Donoso Calle, Jefferson Ivan Vega Sarango y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Firmado electrónicamente por:
DEBBIE DANIELLA
DONOSO CALLE

Debbie Daniella Donoso Calle

Jefferson Ivan Vega Sarango

EVALUADORES

Firmado electrónicamente por:
WASHINGTON ADRIAN
VELASQUEZ VARGAS

Washington Velásquez, PhD.

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
JOSE EDUARDO
CORDOVA GARCIA

José Córdova, PhD.

PROFESOR TUTOR

RESUMEN

El presente proyecto realiza el análisis de la necesidad de un monitoreo de los laboratorios de la carrera de ingeniería en Telemática de ESPOL, esto debido a que, en la actualidad el uso de dispositivos electrónicos y plataformas es creciente, por lo que el manejo de los recursos TI es esencial para la seguridad de los datos y una adecuada inspección de los recursos que están siendo consumidos por los usuarios. Se realizó la implementación de un sistema de monitoreo de recursos TI usando una raspberry pi, softwares como Android Studio, Kali Linux y FireBase, implementando scripts de bash / Python para la recolección y tratado de los datos en Kali Linux instalado en la raspberry pi que actuó como un man in the middle, así también se utilizó una base de datos NoSQL para un mejor manejo de la información obtenida con las herramientas de Ethical Hacking. Al realizar la implementación del sistema de monitoreo, se observó un incremento en el control de las sesiones de laboratorio, pues los encargados pudieron observar la cantidad de hosts conectados, sus puertos abiertos, el tiempo de conexión, el ancho de banda consumido y generar un reporte semanal / mensual de acuerdo con sus necesidades. El presente proyecto permite realizar un monitoreo de los recursos TI en tiempo real, colaborando de esta manera a un mayor control de la seguridad de los datos de la institución, así también brinda la opción de implementar nuevas opciones de acuerdo con los requerimientos del cliente.

Palabras Clave: Script Bash, Python, Raspberry, Android Studio, Ethical Hacking

ABSTRACT

This project analyzes the need for monitoring of the laboratories of the ESPOL Telematics engineering degree, this is due to the fact that, currently, the use of electronic devices and platforms is increasing, so the management of IT resources is essential for data security and proper inspection of the resources that are being consumed by users. The implementation of an IT resource monitoring system was carried out using a raspberry pi, software such as Android Studio, Kali Linux and FireBase, implementing bash / Python scripts for the collection and processing of data in Kali Linux installed on the raspberry pi that acted as a man in the middle, so a NoSQL database was also used for better handling of the information obtained with the Ethical Hacking tools. When carrying out the implementation of the monitoring system, an increase in the control of the laboratory sessions was observed, since the managers were able to observe the number of connected hosts, their open ports, the connection time, the consumed bandwidth and generate a weekly / monthly report according to your needs. This project allows real-time monitoring of IT resources, thus contributing to greater control of the institution's data security, as well as providing the option of implementing new options according to customer requirements.

Keywords: Script Bash, Python, Raspberry, Android Studio, Ethical Hacking

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
NMAP	Network Mapper
TI	Tecnología de la Información
IOT	Internet Of Things
PC	Personal Computer
RAM	Random Access Memory
SNORT	Intrusion Detection System
WiFi	Wireless Fidelity
MQTT	Message Queue Telemetry Transport
MySQL	My Structured Query Language
NoSQL	Not Only Structured Query Language
PHP	Hypertext Pre-Processor
CPU	Central Processing Unit
SSH	Secure SHell
Http	Hypertext Transfer Protocol
Sh	Shell
Bash	Bourne-Again Shell
Py	Python
IP	Internet Protocol
MAC	Media Access Control
API	Application Programming Interfaces
CSV	Comma Separated Values
OSPF	Open Shortest Path First
SNMP	Simple Network Management Protocol
VnStat	Monitor de Tráfico de Red Ligero

SIMBOLOGÍA

KiB	kibibyte
Kbps	Kilibits por segundo
MB	Megabytes
GB	Gigabytes
GiB	Gibibytes
MiB	Mebibyte

ÍNDICE DE FIGURAS

ÍNDICE DE CODIGOS DE PROGRAMA

CAPÍTULO 1

1. INTRODUCCIÓN

La revolución digital se propagó de forma masiva en la década del 2000, alcanzando a finales de 2005 una población de Internet que alcanzó los mil millones, llevando un incremento extraordinario anualmente y superando los dos mil millones en 2012 de tal forma que se generalizaba su dimensionamiento e interconectividad entre dispositivos[1]. Sin embargo, no se consideraba para esta etapa una forma de controlar o visualizar los servicios en red de los sistemas operativos en las estaciones de trabajo, puesto que no se contaba con las herramientas de desarrollo y los conocimientos limitados para su implementación, derivando al desequilibrio para ciertas áreas que requerian de una mayor operatividad y disponibilidad para realizar tareas de mayor exigencia en consumo de recursos.

La creación de topologías de red y el diseño de una arquitectura robusta influyen en la detección adecuada de errores, al monitorear los elementos y características que conforman una red de dispositivos u ordenadores para ofrecer a los usuarios de estos equipos una calidad de servicio mejorada tomando en consideración el comportamiento y uso de recursos durante jornadas de alta demanda y siendo capaz de mostrar el rendimiento, comportamiento y consumo de recursos en un período determinado mediante el análisis y recolección de tráfico en una infraestructura establecida.

Actualmente, el control y monitoreo de los laboratorios de una universidad, escuela, colegio e instituciones que proporcionan recursos TI, son de gran relevancia para el correcto manejo de los mismos, pues existe una creciente demanda en el uso de dispositivos electrónicos y estaciones de trabajo [2], lo cual conlleva a un uso considerable de plataformas, redes sociales y aplicativos de terceros que pueden disminuir la atención de la actividad principal que se debe realizar durante una sesión.

Además, de reducir las capacidades de los recursos de red del área en donde los usuarios se encuentren realizando actividades académicas, por lo cual se torna imprescindible la implementación de un sistema para administrar de forma automatizada la gestión de recursos en cada período determinado.

Considerando la perspectiva que se determinará mediante el monitoreo y su relevancia para el cumplimiento de los objetivos, a través de métricas de rendimiento, características de la topología alámbrica o inalámbrica o en su efecto, contabilizar y visualizar de forma gráfica la cantidad de dispositivos conectados y el consumo de red. Llevando a cabo un monitoreo activo, que pudiera ocasionar mayor tráfico en la red, o un monitoreo pasivo con la implementación de diversas técnicas y con el beneficio de herramientas de software libre lo cual garantice un mejor funcionamiento de los dos dispositivos de red [3].

El presente trabajo comprende para el primer capítulo la instauración de la problemática que antecede a la investigación, así mismo se describe la justificación para la ejecución del proyecto en cada contexto definido, para luego definir los objetivos a lograr en cada ciclo a su vez las limitaciones que se ha presentado durante el desarrollo junto con los alcances que complementan cada instrucción y la respectiva investigación fundamentalmente basada en las características que se desplegará en relación a las tecnologías implementadas tanto a nivel de hardware y software que dispondrá de forma más detallada en la metodología.

1.1 PROBLEMÁTICA

Los laboratorios de la carrera de ingeniería en Telemática - ESPOL, han intentado mejorar el control de los recursos utilizados en cada sesión de clases poniendo la plataforma IoT creada en semestres anteriores a disposición de los jefes de laboratorio actuales. Sin embargo, en la actualidad se ha incrementado la importancia de la implementación de las tecnologías de información para el monitoreo de recursos.

En [4] indica que varias instituciones están dependiendo de las Tecnologías de la Información (TI) para tener como resultado una mejor eficiencia y eficacia de sus operaciones, y en ciertos casos las TI son implementadas como supervivencia; es decir, para tener un correcto manejo de los recursos. En este escenario está comprobado

que las instituciones/organizaciones grandes invierten más en TI, pero un sistema de monitoreo de recursos TI es costoso, pues los softwares confiables para realizar este tipo de control los tienen a disposición empresas reconocidas a nivel mundial. [5] [1] [2]

1.2 JUSTIFICACIÓN

La presente investigación se enfoca en el estudio del consumo de tráfico y recursos de red que cada host tiene en los laboratorios de redes de datos y sistemas telemáticos de la ESPOL, ya que debido al uso de recursos de red que cada usuario de laboratorio tenga, el tráfico de red se ve afectado. Este trabajo permite mostrar un estado de cuenta al final de un periodo determinado, así como tener un módulo de administración para los jefes de laboratorio, para determinar la cantidad de recursos utilizados por cada host en determinados puertos, conocer que aplicación/es utilizan los usuarios, conocer cuál es la red más utilizada de las redes del área de telemática y tener una mejor operación de los laboratorios para conocer cómo se utilizan los recursos.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

- Crear un sistema de monitoreo que permita tener un control sobre los recursos TI inalámbricos de los laboratorios de redes de datos y sistemas telemáticos utilizando software de uso libre.

1.3.2 OBJETIVOS ESPECÍFICOS

- Conocer la cantidad de tráfico de red que se utiliza durante una sesión de laboratorio mediante la obtención de la cantidad de personas que se conecten a la red.
- Implementar un aplicativo de monitoreo utilizando un marco de referencia para el respectivo control de los laboratorios.

[1] https://geekflare.com/es/network-monitoring-software/
[2] https://www.paessler.com/prtg/prtg-network-monitor

- Evaluar escenarios de prueba en los laboratorios de redes de datos y sistemas telemáticos para la obtención de reportes de recursos TI.

1.4 ALCANCE Y LIMITACIONES

1.4.1 ALCANCE

El alcance del estudio se limita a obtener datos de los recursos TI de la parte inalámbrica de los laboratorios de sistemas telemáticos y redes de datos de la ESPOL, los cuales se presentan en un aplicativo que esta conectado a la plataforma IoT desarrollada por los estudiantes Pilco y García. [6]

Las pruebas para obtener dichos datos se realizarán únicamente con las personas que hagan uso de los laboratorios mencionados anteriormente, que se encuentren en horarios de clases o que realicen prestamos de las PC ubicadas en estos lugares. Este periodo de prueba tendrá la misma duración que el presente término académico.

Además, el sistema de monitoreo que se propone crear está orientado para el uso exclusivo de los jefes de laboratorio, con el fin de que puedan obtener un reporte de la información más importante de los recursos IT.

1.4.2 LIMITACIONES

Las limitaciones para la realización de esta investigación son:

- **1)** El periodo de prueba de la recolección de los datos del router de cada laboratorio es muy corto para determinar con precisión un reporte adecuado.
- **2)** El alto costo del microcontrolador a utilizarse dentro del diseño de la implementación del presente proyecto.
- **3)** Falta de información actualizada sobre temas relacionados al monitoreo de recursos IT inalámbricos al momento del estudio.
- **4)** Acceso restringido a las redes alámbricas de los laboratorios en donde se implementará la plataforma de monitoreo.

- **5)** El tamaño de la muestra para realizar las pruebas correspondientes de la presente tesis es pequeño, pues se tomarán en cuenta solo las personas que hagan uso de los laboratorios de sistemas telemáticos y redes de datos.

- **6)** Poca información sobre software/hardware para realizar sniffers o un proxy en un router inalámbrico y obtener información relevante.

1.5 MARCO TEÓRICO

Existen diversos softwares para realizar un monitoreo eficiente de recursos TI pero todos estos, en su gran mayoría no son de uso libre, pues disponen de diversas opciones de control, además se debe mencionar que para poder elegir un sistema eficiente se deben de tener en cuenta diversas variables [7].

En [8] se indica que al seleccionar una infraestructura de monitoreo se necesita tener en cuenta varios factores para encontrar la combinación perfecta de acuerdo con las necesidades que se tengan. Como primer paso, se debe seleccionar una herramienta que pueda satisfacer las funcionalidades requeridas, luego se debe evaluar el factor de despliegue y mantenimiento del sistema. Finalmente, se logrará comprender como la herramienta puede llegar a afectar la organización y de esta forma se puede calcular el costo total de la implementación.

Por este motivo, se pretende crear un sistema de monitoreo usando software libre para la reducción de costos en la implementación del mismo, pero de acuerdo a lo indicado en el párrafo anterior se debe tener en cuenta que dispositivos y softwares serían los más adecuados, por lo que el dispositivo que más encajaría en el diseño de la solución sería el microcontrolador Raspberry Pi, pero existen varios modelos, por lo que de acuerdo con [9] la raspberry pi 3 no sería óptima para la obtención de datos del router a utilizarse, pues indica que:

Los valores del procesador y RAM comienzan a cambiar a medida que se aumentan el número de reglas, esto debido a que se implementa el sistema de detección de intrusos llamado SNORT, produciendo así un cambio negativo en el rendimiento del microcontrolador, a pesar de si provoca un aumento en las tasas de captura de paquetes.

Además, para el software que la raspberry pi debe tener para poder actuar como un snnifer y capturar el tráfico de la red, se deben revisar todas las opciones existentes de

uso libre, así como en el párrafo anterior se mencionó a SNORT, también existe uno llamado “Wifi Auditor”, que de acuerdo con [10] es uno de los más completos si se desea realizar un man in the middle:

WiFi-Ace permite que el atacante pueda interceptar y lanzar un ataque de ”hombre en el medio”, esto gracias a que puede inspeccionar los datos que fluyen entre la víctima y los recursos a los que accede en la web. WiFi Auditor se puede utilizar en un gran número de redes que utilizan WiFi, también es barato frente a otras alternativas como WiFi Pineapple.

En [11] se indica que se usó una raspberry pi 3 para la comunicación de datos y monitorización remota en tiempo real, pero para la adquisición de la información se usó una Smart metros de la serie Elite 440-443 de Secure Pvt. Ltd. para un sistema integrado solar-eólico-biogás basado en un sistema de micro red, además para mostrar dichos datos se usó VNC Connect, la cual es una plataforma Internet de las cosas (IoT - Internet of Things) de código abierto, en donde VNC Server se instaló en la raspberry y VNC Viewer se utilizó en los diversos dispositivos para realizar el respectivo monitoreo. A pesar de no ser un sistema de monitoreo de recursos TI se puede confirmar que el microcontrolador más adecuado es la raspberry pi para la recopilación de información relevante, i.e, que pueda funcionar como un middleware, así también en el trabajo anteriormente mencionado se usaron códigos de Python (scripts) que se escribieron en la raspberry, además los datos recopilados fueron almacenados en la nube, lo cual ha permitido analizar de mejor manera la base de datos que se utilizará en este proyecto.

Por otro lado, en [12] se indica que se realizó un análisis de cosecha inteligente con la Raspberry Pi basado en Internet de las cosas, el mismo que se presentó en un *dashboard*. Para la implementación de la arquitectura se usó un sensor *DHT22* para poder detectar la temperatura y humedad, para la recolección de los datos se usó un módulo wifi basado en *ESP8266*, una raspberry pi que funcionó como servidor pues se implementó el protocolo Message Queing Telemetry Transport (MQTT), así también se usó *Bootstrap* para el diseño del sistema de monitoreo, pues usa Synyactically Awesome Stylesheets (Sass) para una arquitectura que puede ser personalizada. Aunque el trabajo que se ha mencionado no recopila datos de recursos TI, permite analizar el microcontrolador más óptimo para recopilar datos en la presente investigación, así como el software para el diseño del aplicativo que se realizará para la presentación de la información y el respectivo

monitoreo de los laboratorios de sistemas telemáticos y redes de datos.

Así como en los trabajos previos mostrados en líneas anteriores, se tiene el de [13] en el cual se realizó un sistema de control de flujo basado en IoT usando Raspberry PI, pero a diferencia de las otras investigaciones en esta de aquí se utlizó un Arduino para la recopilación de los datos pues se usó un sensor para medir el nivel del agua, para luego enviar esta información a la raspberry pi en donde a través de scripts de Python se realizó la lectura de lo recopilado y de la creación del aplicativo web, para lo cual se usó Flask. Tomando en cuenta el trabajo mencionado, se puede contrastar el mismo con el presente trabajo final, pues en un capítulo posterior se mencionará la arquitectura a usar, pues para la presente propuesta se realizará un sistema de monitoreo de recursos TI, en donde se utilizará una raspberry pi como man in the middle para la recopilación de datos relevantes.

En [14] se indica que para el desarrollo de un sistema de monitoreo de intensidad de luz basado en la nube utilizar una raspberry pi es óptimo para que sea un servidor web, pues no tendría problemas de sobrecalentamiento al no recibir tanto tráfico, por lo que se usó Apache en el microcontrolador y se usó el lenguaje PHP, además para el almacenamiento de datos se utilizó MySQL y para el aplicativo se usó Acquire por el cual el cliente accedía a través de la dirección IP del servidor. Pero a diferencia del trabajo mencionado, para la presente investigación se utilizará una base de datos NoSQL pues se requerirá almacenar una gran cantidad de datos, así como también los datos se desean guardar como clave-valor.

También, como se indica en [15] para realizar la gestión y monitorización de redes para sistemas en la nube, se implementó una solución basada en Nagios, pues permite monitorizar las redes y servidores, además es de código abierto, así también da la opción de observar equipos y servicios, se usó Nconf para configurar el sistema de Nagios. De la misma forma, se usó una base de datos como servicio para proporcionar un enfoque más permisivo, pues no requiere de un esquema como las bases relacionales. Por consiguiente, el monitoreo de un sistema en la nube no es igual a uno de recursos TI, pues los recursos están virtualizados, pero se deben de monitorear los mismos parámetros, por lo que al igual que en el paper mencionado, se requiere de una base de datos no relacional para almacenar la información que se obtenga y mostrarla en el sistema que se propone en esta investigación.

De forma similar, para realizar la implementación de un sistema de monitoreo de redes, [16] indica que la herramienta *Nagios* para obtener información actual sobre el CPU, memoria, ping, ancho de banda, usuarios conectados, entre otros, además se usó directorio activo para tener una lista con los equipos conectados y obtener información de cada uno de estos a través de una búsqueda rápida. También, para visualizar la información se usó el portal web de Nagios en donde se pueden observar gráficas y estadísticas.

Por otro lado, para la implementación de monitoreo de red de un Hospital. [17] indica que usó el sistema de monitoreo *PandoraFMS*, el cual lo desplegó en *Centos* y se accedió a éste por *SSH/Http* usando la dirección IP, ya en este sistema utilizó herramientas como *NetScan* para detectar los host conectados y al dar clic sobre cada uno de ellos pudo observar información sobre el tiempo de conexión/ancho de banda consumido, así también en este sistema pudo observar tráfico de red con la herramienta *SNMP*, además de analizar puertos abiertos, paquetes perdidos y generar alertas que pudiesen ser enviadas a un email ante la detección de un problema en la red.

Así también, para realizar el rediseño e implementación del sistema de monitoreo de la red de telecomunicaciones de distribuidora nissan s.a, [18] indica que para la parte del frontend utilizó *PHP, BootStrap, KickStrap y Fundation*, pues estos ofrecen una gran variedad de componentes, además para el almacenamiento de los datos se usó *MySQL Workbench* pues la información que se guardó tenía un *id* único que identificaba a cada fila. Adicional, se usó *Apache Jmeter 2.13* para someter al sistema a pruebas de estrés y medir la capacidad de respuesta a determinados usuarios que consultarán el aplicativo. Del trabajo mencionado anteriormente, se destaca el software que implementaron para someter al sistema de monitoreo a pruebas de estrés, el cual será analizado para ser aplicado en la presente investigación.

Por otra parte, [19] indica que para el monitoreo y gestión de la seguridad de la red y plataformas windows y linux aplicado a empresas medianas realizó la implementación de la herramienta *OSSIM*, debido a que es una herramienta opensource que permite monitorear los logs de los eventos sucedidos en la red, pero además es un sistema de manejo de eventos y seguridad de la información; es decir, permitió observar las vulnerabilidades de la red, para de esta forma tomar acciones oportunas, además con esta herramienta se pudo observar si un equipo que iba a ser analizado estaba *UP* o

DOWN, así como detectó intrusos en la red monitoreada.

De esta forma, [20] describe la realización de un análisis profundo del tráfico de la red de computadoras y métricas en base a la monitorización en tiempo real, mediante datos graficados en un dashboard desarrollado en la plataforma de *Grafana*, por ello se incluye la identificación de vulnerabilidades expuestas en los ordenadores lo que permite mitigar estos riesgos e implementar los respectivos cambios físicos y lógicos incrementando la seguridad y mejorar la toma de decisiones. Dado que un sistema de monitoreo está enfocado en analizar parámetros como la trasmisión de paquetes, direcciones IP, direcciones MAC, así como los riesgos y vulnerabilidades que se pueden derivar de posibles fallos suscitados en la interacción de los dispositivos electrónicos que se encuentran en constante comunicación. Así, se destaca la funcionalidad de *Grafana* colaborativa y de código abierto lo cual presta grandes beneficios y utilidades para ser aprovechadas por los desarrolladores ofreciendo una ventaja de conectarse con cualquier lenguaje de programación a través de una API.

En cuanto al control y monitoreo de laboratorios, [21] implementó una plataforma basada en tecnología *RFID* sobre una arquitectura *Cloud computing open stack* que permitió controlar la información de entradas y salidas de los equipos de los laboratorios en tiempo real. La información del uso y manipulación de los dispositivos se recolecta mediante el uso de tecnología *RFID*, puesto que este tipo de tecnología es la más usada en cuanto se refiere a la detección y monitoreo de equipamiento centralizándose tanto en software y hardware. Consta de tres etapas de monitoreo siendo detección, lectura y manejo de datos respectivamente. Para el diseño de interfaz desplegó una aplicación web y de escritorio basado en código *C* para el servicio web y código en *Java* para la aplicación de escritorio usando un modelo de base de datos relacional *SQL* para la manipulación de datos. Por ello, se destaca de este trabajo la aplicación utilitaria para los laboratorios de una facultad, fundamentándose en arquitectura de *Open Stack* que brinda de escalabilidad, flexibilidad, compatibilidad siendo de código abierto brindando herramientas de plataformas cloud con creación de productos a bajo costo.

En consideración con [11] que desplegó una plataforma *CloudIOT* para el control y monitoreo de equipos y programas informáticos de aulas y laboratorios, mediante la creación de módulos que se encuentran instalados en los ordenadores, a su vez envian los datos recopilados al servidor en el que reside el proyecto o almacenando

las actividades en archivos fuera de línea. Este sistema además, está orientado para adicionar otros campos que son constantes por equipo como nombre, usuario, etc. La información que se va generando en un equipo se filtra, y se organiza en filas de tal manera que bajo todo concepto se garantiza la integridad y validez de los datos monitoreados. Lo destacable de este proyecto es que utliza herramientas *Open Source* en todas sus etapas. Para el almacenamiento de datos se inclina por un servidor de base de datos de *MariaDB* ofreciendo librerías, clases y controladores que conservan la denominación de *MySQL*. Para el almacenamiento en tiempo real de las aplicaciones ejecutadas, se respalda de forma local en un archivo plano *CSV* siendo la mejor alternativa consumiendo recursos de hardware en menor proporción comparados con *SQLite*. Esto brinda una alternativa eficiente para el manejo continuo de datos a nivel de consumo de recursos de red de forma análoga y trabajando con datos en una estructura relacional.

En relación con [22], que se focaliza en el control y supervisión del laboratorio de automatización y control totalmente integrado (LACTI) de la Universidad Politécnica Salesiana (UPS) a través de un servicio en la nube. El laboratorio cuenta con una planta de llenado y vaciado de un tanque cerrando con válvulas y sensores que es controlada por un PLC Siemens S7-1200 y un módulo IoT-2040 que contará con un sistema operativo Yocto Linux instalado en una microSD. Para la integración del servicio en la nube se crea desde IBM Cloud Watson una aplicación de Node-RED que establece la comunicación entre los dispositivos IoT y se comunican mediante claves API e incluyen un *dashboard* que representará todas las acciones de control y monitoreo para la planta. De forma similar es relevante la inclusión de herramientas open source en en desarrollo de la interfaz del servicio en la nube, lo cual favorece en el control remoto y en tiempo real de recolección de datos así como las alertas programadas y visualización del rendimiento, alarmas y seguridades para el monitoreo de datos.

De manera similar para el monitoreo de parámetros ambientales en interiores y exteriores [23], utilizó tecnología LoraWan para visualizar las variables ambientales a través de una interfaz web en tiempo real. Utlizando dispositivos como *Arduino MKR 1300*, nodo *Lora-RAK 811*, sensores y una *Raspberry Pi* 3 para la correcta adquisición y transferencia de datos integrando la plataforma IoT para la visualización correspondiente. La utilidad esencial de la *raspberry* en esta aplicación consiste en un nodo Gateway

que receptará la información de los nodos sensores y posteriormente enviará toda la información en intervalos de tiempo cortos hacia el nodo de monitoreo que dispone de la interfaz web desarrollada en Ubidots y gestionando los datos por TTN. De forma notable se implementa una solución open source para el dashboard que garantiza la compatibilidad entre todos los dispositivos y cuenta con una interfaz amigable para el usuario final.

Para la implementación, administración y monitoreo de una red corporativa simulada [24], se desplegó mediante un servidor de monitorización de código abierto para redes y aplicaciones *(Zabbix)* para un laboratorio de redes virtual en la Escuela superior politécnica del ejército (ESPE). Utlizando dispositivos de *hardware Mikrotik* y enrutamiento *Open shortest path first (OSPF)*, para conexión entre sedes se usó una red privada virtual (VPN) con IPsec para ser monitoreada en el servidor descrito que se encuentra instalado en una máquina virtual con *Ubuntu 20.04*, agregando un host por cada router estableciendo el protocolo SNMP para una conexión directa entre la red corporativa y el servidor web lo que permite monitorear los recursos del router como memoria, carga de CPU, interfaces y gráficas y para notificar fallos se envía un correo hacia el administrador con los detalles. Es relevante la utilidad del aplicativo del servidor web que incluye un *dashboard* resumiendo el monitoreo e información de estado de cada equipo en tiempo real respectivamente simuladas en el software de simulación de gráfica de redes *GNS3*.

En el presente trabajo final se propone crear un sistema de monitoreo de recursos TI utilizando una Raspberry Pi 3 para obtener los datos, este microcontrolador estará conectado a un router a través de Ethernet, pues actuará como un *man in the middle*, de esta manera los datos recopilados por la raspberry serán enviados a una base de datos *NoSQL*, pues no se requiere una estructura, solo una *clave-valor* para almacenar los datos, ya que éstos irán cambiando en el tiempo de manera impredecible, además una vez en la base de datos serán leídos por el software que se utilizará para la realización del aplicativo y de esta manera pueda ser visualizado por el usuario desde su dispositivo móvil.

Así también, el modelo de negocio que se propone es el de *'Bait Hook'*, pues se pretende ofrecer el producto final a un bajo costo, para que de esta manera los clientes interesados se vuelvan fieles al mismo y en un futuro poder ofrecerles herramientas

adicionales para un monitoreo más óptimo e incluso escalar el aplicativo a diferentes versiones. [3]

[3] https://www.sinnaps.com/blog-gestion-proyectos/modelo-de-negocios

CAPÍTULO 2

2. METODOLOGÍA

En este capítulo se describe el método que se utiliza para la creación del esquema del proyecto, de la misma manera se definen el tipo de investigación el cual es implementado para obtener la información más relevante para la realización del diseño planteado. Finalmente, se indica el porque de los datos requeridos y la manera en la estos son obtenidos.

Dada la finalidad de la metodología de describir la realidad del problema a tratar y realizar una recolección de datos, esta investigación es de tipo descriptiva y experimental. Finalmente, la información a utilizar para el progreso del proyecto es fruto de fuentes, como papers, libros, artículos científicos.

Para realizar la recopilación de los datos, se realizan pruebas en los laboratorios de sistemas telemáticos y redes de datos, mediante scripts de bash y python ejecutados en una raspberry pi.

Frente la inminente necesidad de realizar un monitoreo de red en tiempo real de industrias, instituciones educativas, empresas, hogares y la creciente demanda de la ciberseguridad en los diversos sitios mencionados anteriormente, nace la intención de implementar un sistema que permita conocer datos claves de la red, pero que a la vez sea de fácil manejo, entendimiento sencillo y costo reducido, es por esta razón que se implementará un sistema de monitoreo de recursos TI a través de un aplicativo móvil, el mismo que será opensource. Este sistema se describe a continuación en la arquitectura a usarse para el desarrollo del proyecto.

2.1 ARQUITECTURA

Para el presente trabajo se realizará el siguiente esquema de proyecto, el cual será implementado en los laboratorios de redes de datos y sistemas telemáticos. Con el fin de crear un sistema de monitoreo de recursos TI para el correcto funcionamiento de estos sitios.

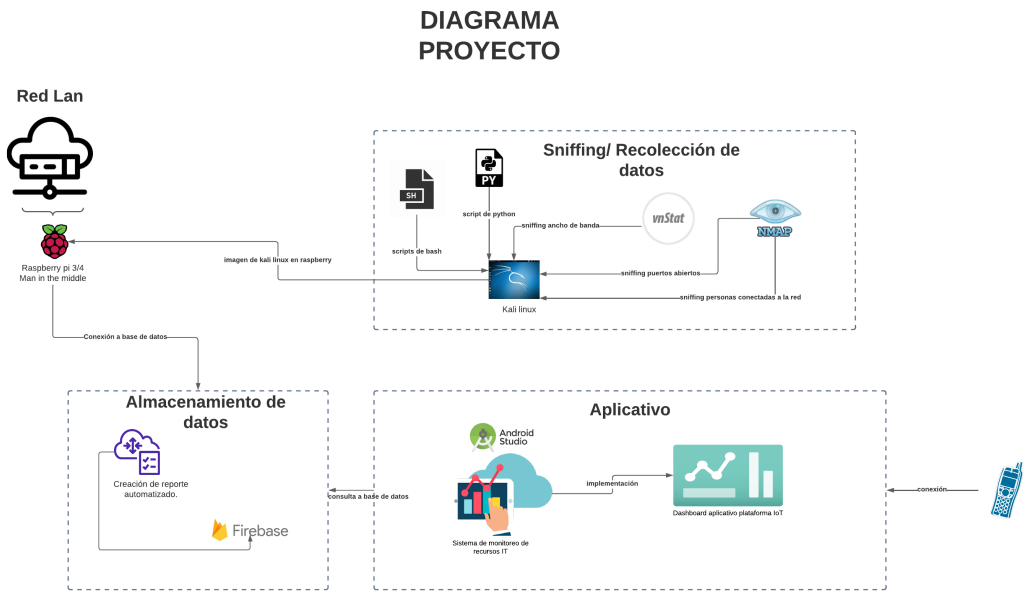

Figura 2.1: Esquema de proyecto seccionado en Recolección de información, Almacenamiento de datos y Muestra en Aplicativo.

Como se observa en el esquema de la figura 2.1, la red LAN es la que se encontrará en cada laboratorio junto con una raspberry pi 3 que actuará como *man in the middle* para poder interceptar información relevante del router.

2.1.1 RECOLECCIÓN DE DATOS

Como se observa en la figua 2.2, para la fase de recolección de datos se implementa kali linux en la raspberry pi, pues se realiza un ataque *man in the middle* para obtener la cantidad de hosts que se encuentran conectados a la red en tiempo real, así como los hosts que se desconectan de la misma, esto se realiza utilizando la herramienta *nmap* y un script de bash para recopilar solo la información necesaria, adicional se utiliza *nmap* también para obtener los puertos abiertos de un determinado host seleccionado por el usuario desde el aplicativo, así también se usa la herramienta *vnstat* para conseguir el

ancho de banda consumido por los host conectados a la red. Se implementa además un script de python en la raspberry para guardar y leer datos de la base de firebase, entre ellos el puerto seleccionado por el usuario desde la app para poder utilizarlo con la herramienta respectiva.

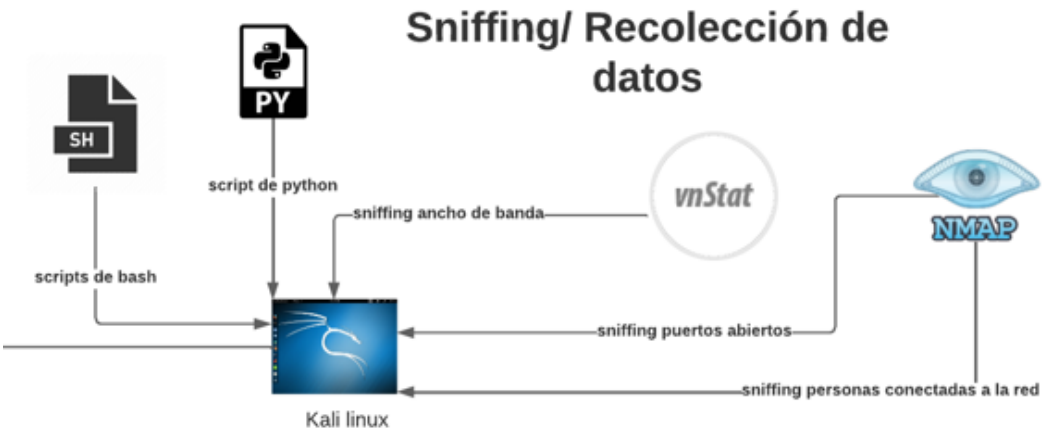

Figura 2.2: Recolección de datos/Sniffing implementando herramientas nmap, vnstat y scripts de bash/python para el tratado de información obtenida.

2.1.2 ALMACENAMIENTO DE DATOS

Para esta fase como se observa en la figura 2.3, una vez que se han ejecutado los scripts respectivos en la raspberry para la obtención de datos, se envía a la base de datos en tiempo real de *firebase* para posteriormente ser leídos y presentados al usuario a través del aplicativo.

Figura 2.3: Almacenamiento de datos

2.1.3 INTERACCIÓN CON EL USUARIO

Para esta fase como se observa en la figura 2.4, se utiliza el software *AndroidStudio* para la creación del aplicativo en donde se mostrarán los datos almacenados en la base de datos, además de que se envía la ip seleccionada por el usuario a la base para que pueda ser usada con la herramienta *nmap*. Además, este aplicativo se conectará con la plataforma IoT ya existente.

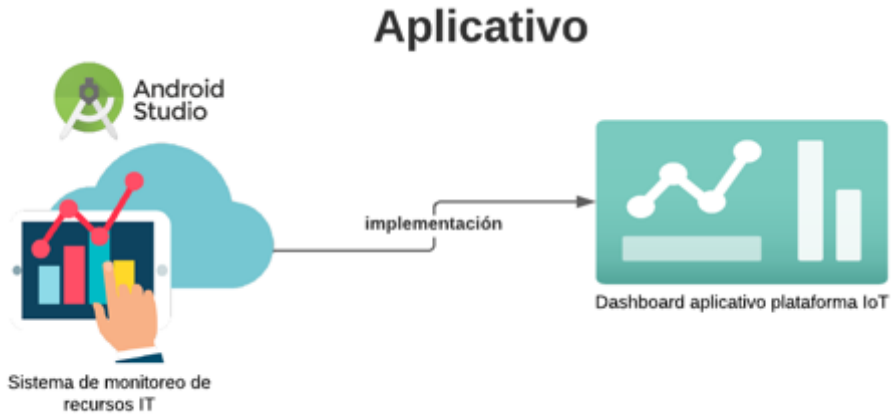

Figura 2.4: Sistema de monitoreo de recursos TI

2.2 MATERIALES Y MÉTODOS

2.2.1 MATERIALES

Los materiales que se utilizan en el proyecto se detallan a continuación:

- Raspberry pi 3
- Software Kali Linux
- Herramienta Nmap
- Herramienta vnstat
- Base de datos RealTime Firebase
- Software AndroidStudio

Se usa una raspberry pi 3 como *man in the middle*, pues en el microcontrolador se tienen scripts de bash y python que permiten obtener los datos relevantes de la red como host conectados en tiempo real, ancho de banda consumido, puertos abiertos.

Además, se utiliza kali linux en la raspberry pi, pues permite realizar un hacking ético a la red objetivo, adicional cuenta con herramientas para poder obtener los datos que se requieren presentar al usuario.

Así también, las herramientas *nmap e vnstat* se usan para realizar un sniffing de host conectados a la red, puertos del host especificado por comando y obtener la cantidad de ancho de banda consumido, se encuentran dentro de kali linux.

Adicional, se utiliza la base de realtime de firebase por ser NoSQL, maneja clave-valor, además de ser de fácil manejo y compatible con el software Android Studio.

Finalmente, se ópta por usar el software Android Studio para realizar el aplicativo, pues su uso es sencillo y la interfaz presentada al usuario no es compleja.

2.2.2 MÉTODOS

Los métodos aplicados a la red, con la finalidad de obtener y detallar las métricas de rendimiento son:

- Ethical Hacking

- Sniffing - Nmap
- Python - Scripting
- vnstat - Bandwidth, Latency
- Bash - Scripting

Se usa el método de ethical hacking para poder identificar las vulnerabilidades de los host conectados a la red, pues kali linux permite implementar este método, ya que posee herramientas especializadas para este tipo de prácticas. De la misma manera, *nmap e vnstat* se utilizan como métodos de recopilación de información de la red. Adicional, para el manejo de los datos obtenidos se usa el método de scripting, pues a través de scripts de bash se implementa los comandos respectivos, además de filtrar los datos más relevantes, así también con un *script* de python se realiza el envío y lectura de datos a la base de firebase.

2.3 MÉTRICAS

2.3.1 MÉTRICAS DE RENDIMIENTO

Para la obtención de las métricas de rendimiento se determinó fundamentalmente el uso de la memoria RAM en el dispositivo móvil durante la ejecución del aplicativo en *Megabytes (MB)*, el tamaño de la aplicación al instalarse en el dispositivo móvil del administrador en *Megabytes (MB)*, el uso de la batería determinado en el porcentaje de la carga del dipositivo móvil. Estos valores corresponden a nivel de la capa de aplicación donde se presenta el módulo del sistema de monitoreo de recursos TI para las consultas, observando el consumo de recursos en el dispositivo principal posterior a las pruebas.

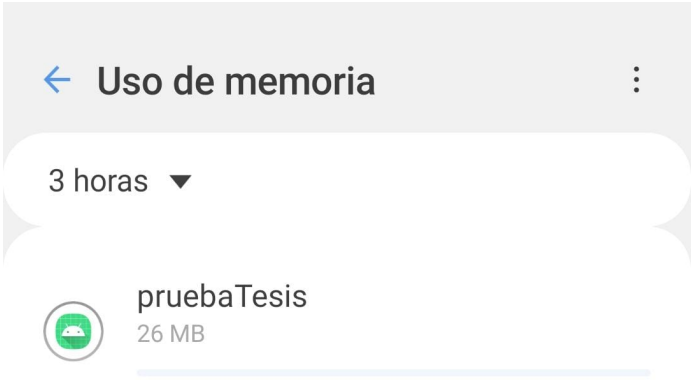

Figura 2.5: Muestra del almacenamiento de la aplicación en el dispositivo.

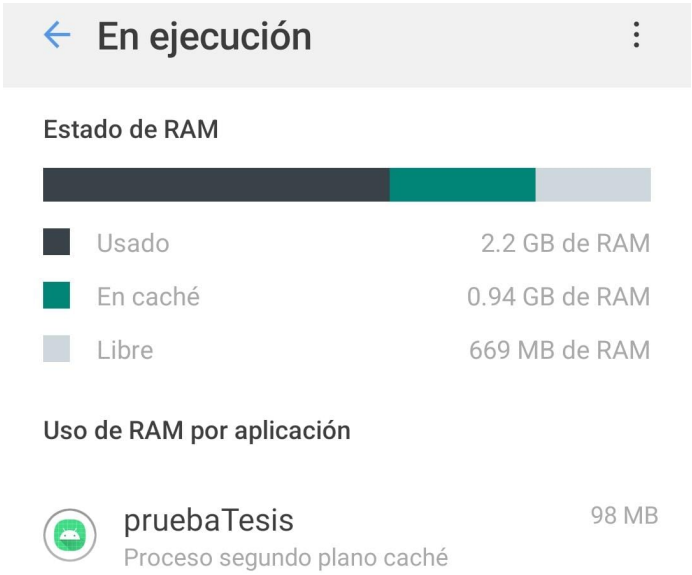

Figura 2.6: Muestra del uso de la memoria RAM en la ejecución de la aplicación.

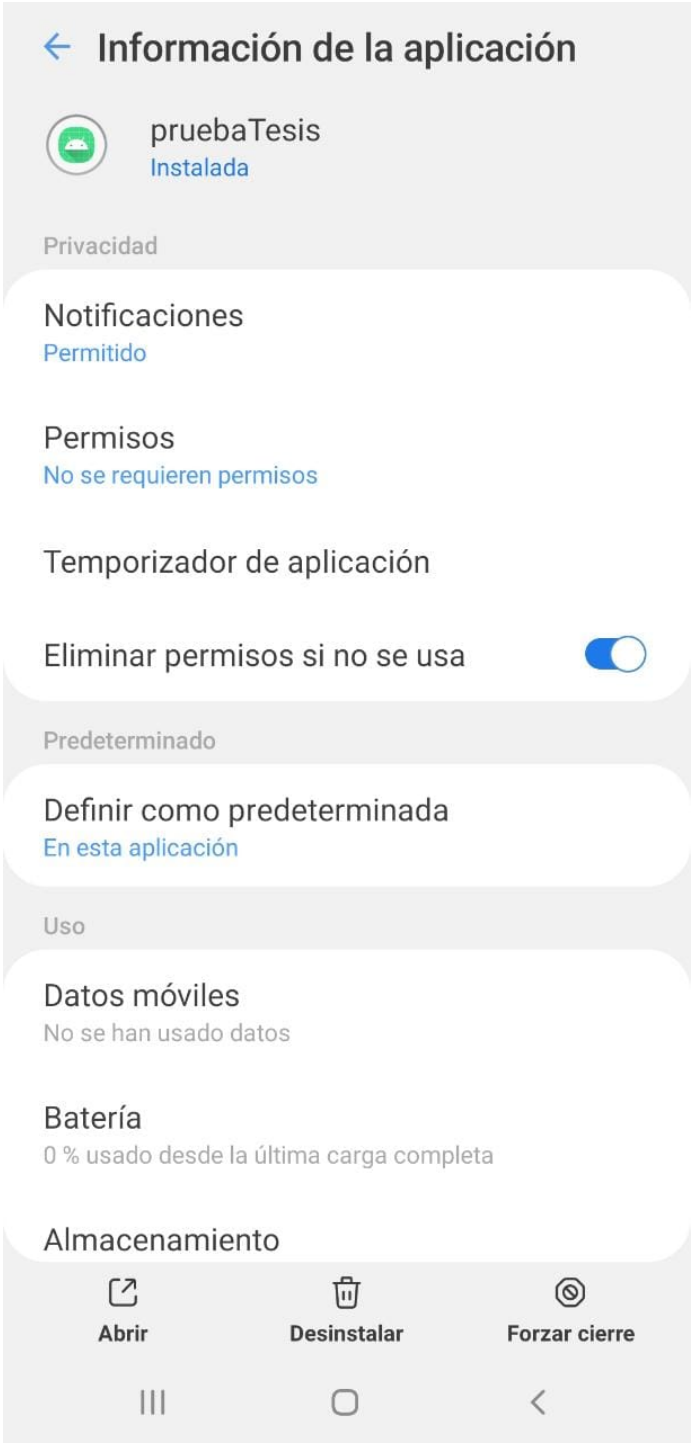

Figura 2.7: Porcentaje de batería utlizado en el dispositivo durante la ejecución.

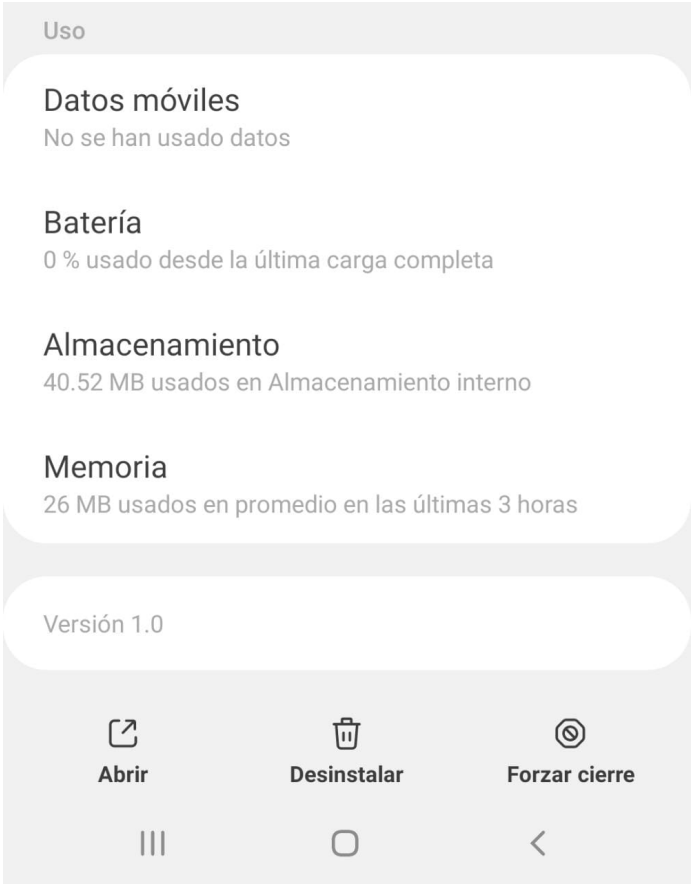

Figura 2.8: Información detallada de las métricas puntualizadas en el aplicativo.

2.3.2 MÉTRICAS DEL SISTEMA

- Cuello de botella (raspberry pi como proxy).
- Tiempo que tardará en ejecutarse el script para obtener los datos.
- Tiempo que tardará en obtenerse los datos al realizar el sniffing.
- Cantidad de datos que la base de datos puede almacenar por día.
- Tiempo que tardará en ejecutarse el aplicativo al usuario.

2.3.3 MÉTRICAS DE SATISFACCIÓN DEL USUARIO

Se realiza la obtención de estos datos para la etapa previa a la implementación final del proyecto, a través de un formulario de Google con preguntas dirigidas a los clientes y a los usuarios de los laboratorios de redes de datos y sistemas telemáticos, con las cuales se conocen los datos que contiene el aplicativo, el diseño del mismo, con que frecuencia se realiza una consulta, entre otros. La muestra de personas que realizan el formulario es de aproximadamente de 6 docentes y más de 60 estudiantes sumando por cada sesión práctica en el tiempo que se permite, pues es la cantidad de usuarios que se encuentran en una hora de clase de laboratorio, además se evalúa por semestre para relacionar los datos y obteniendo una mejor comparativa de datos durante el período vigente.

Entre las preguntas a realizarse están las siguientes:

- ¿Cómo podemos mejorar su experiencia en el monitoreo de recursos ti del laboratorio?
- ¿Con qué frecuencia utilizaría el sistema de monitoreo?

  Cada hora Cada 5 horas Cada día Cada 3 días Cada semana Cada mes
- ¿Cómo quisiera que fuese el método de busqueda de los reportes automatizados?

  Mensual Semanal Diario Anual
- ¿El producto le ayudaría a lograr sus objetivos?

  Sí No Tal vez

- ¿Qué mediciones le gustaría que se muestren en la plataforma de monitoreo?

  Latencia Páginas visitadas Tiempo de respuesta de las conexiones Ancho de banda Host conectados Tiempo de conexión de los host Puertos abiertos

- ¿Qué características del sistema de monitoreo consideraría más valiosas?

- ¿Qué características del sistema de monitoreo utilizaría con mayor frecuencia en su día a día?

CAPÍTULO 3

3. DISEÑOS E IMPLEMENTACIÓN

En el presente capítulo se muestran los componentes para la app tales como los diseños de bases de datos, de interacción de usuario y de comunicación. Para esto se parte de la identificación de los requerimientos de software y hardware, así como el proceso del flujo de la comunicación entre las diversas herramientas a usarse. Estos diseños proporcionan una idea completa del funcionamiento de la aplicación y del manejo de los datos. Adicional, se considera relevante que el diseño de la interfaz del aplicativo sea de fácil uso y de uso práctico para el usuario.

3.1 DISEÑO DE INTERACCIÓN DE USUARIO

Como se observa en la figura 3.1, el proceso comienza con el usuario ingresando al aplicativo, como primer paso debe autenticarse para seguir avanzando, luego de esto el usuario tendrá la opción de visualizar el reporte de recursos TI, en donde se le muestran los host conectados a la red en tiempo real, así como los puertos abiertos de cada host, el ancho de banda consumido y el tiempo de conexión, al mismo tiempo que estos datos son mostrados al usuario, se realizan las respectivas peticiones a la base de datos de firebase y a su vez también los scripts contenidos en la raspberry realizan consultas en ella.

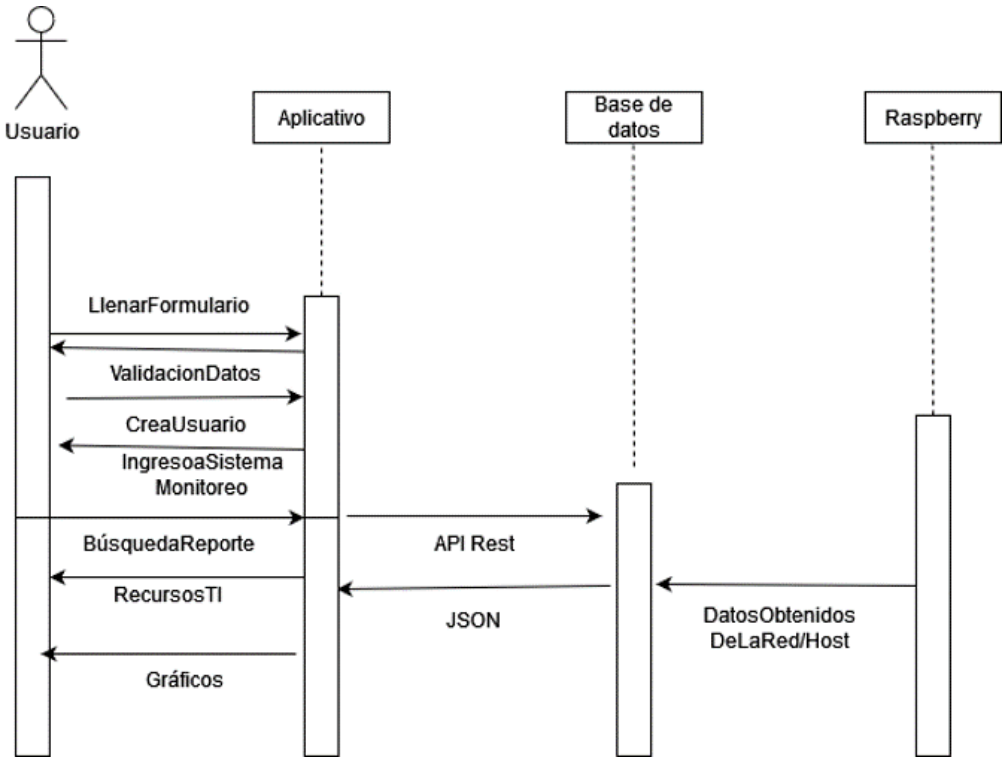

Figura 3.1: Diseño de interacción de usuario con el sistema

3.2 DISEÑO NO-RELACIONAL DE LA BASE DE DATOS

En la figura 3.2 se muestra el esquema de la base *realtime* de firebase, se utilizó una base de datos no relacional, debido a que permite tener escalabilidad en los datos que se almacenan, además de ser descentralizada, además guarda los datos como clave-valor, de la misma manera permite realizar consultas de manera más eficiente si se tiene una gran cantidad de datos, de esta manera cuando el usuario selecciona una ip para visualizar los puertos abiertos o cuando selecciona una hora determinada del día, un mes u día específico para visualizar el ancho de banda consumido no existirán inconvenientes al almacenar cada uno de estos datos, así como también la consulta de estos será mucho más sencilla. Adicional, se tiene la clave principal que es recursosTI, la misma que tiene asociadas las claves host, hostconectados, ip, ipSeleccionada, puertosactivos y totalpuertosactivos, que a su vez cada una de ellas tiene asociados respectivos valores. Para la clave host, se tiene el valor host, este dato se almacena desde el *script de python* que se encuentra en la *raspberry* para luego ser mostrado en el aplicativo. Para la clave hostconectados, se tiene el valor total, este dato se utiliza principalmente para obtener la cantidad de host conectados en tiempo real y aquellos que estuvieron conectados.

Para la clave ip, se tienen los valores fechaconexion, horaconexion e ip, estos datos son usados dentro del código de *AndroidStudio* para obtener el tiempo de conexión de cada host. Para la clave ipSeleccionada, se tiene el valor ip, este dato es almacenado desde el código de *AndroidStudio*, pues es la IPque el usuario da click para poder conocer su información, este número se lee desde el *script de python* de la *raspberry* para luego ser utilizado por la herramienta *nmap*. Para la clave puertosactivos, se tiene el valor puertosabiertos, este dato es obtenido desde el *script de bash* en la *raspberry* y leído desde el código de *AndroidStudio* para ser mostrado al usuario cuando se da *click* sobre una determinada ip. Finalmente, para la clave totalpuertosactivos, se tiene el valor total puertos, esta información es utilizada para obtener la cantidad total de puertos abiertos que tiene un determinado host.

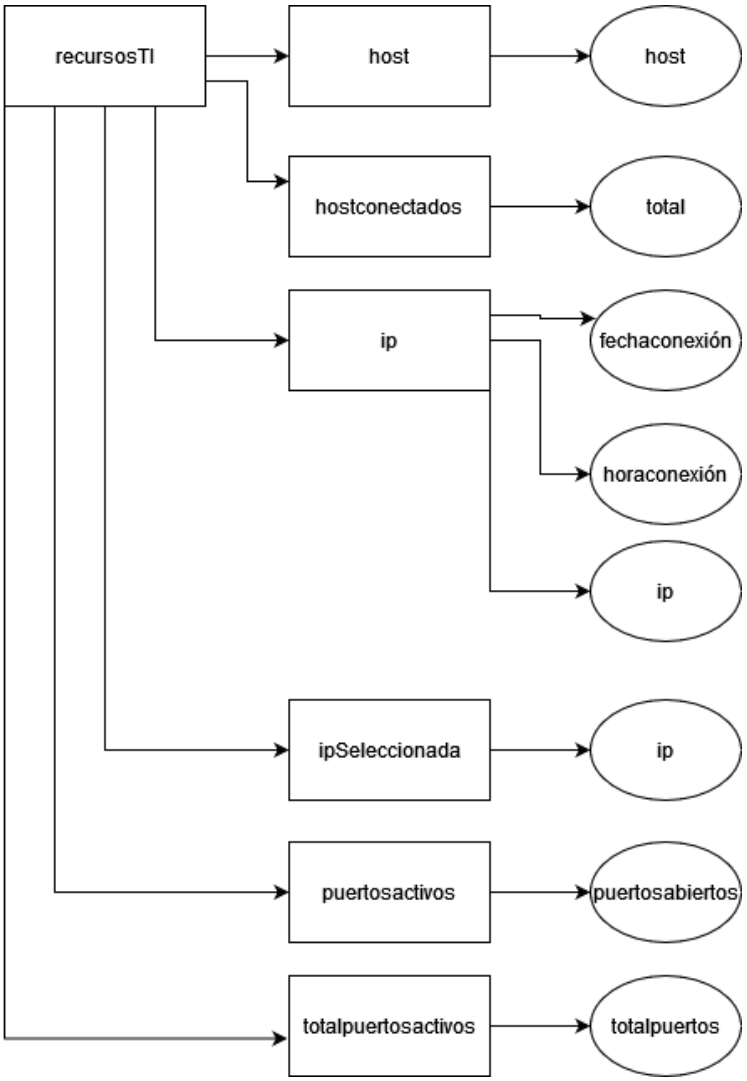

Figura 3.2: Diseño de la base de datos realtime firebase

3.3 DIAGRAMA DE COMUNICACIÓN - PUERTOS

Como se muestra en la figura 3.3, para obtener los puertos abiertos de un determinado host, el aplicativo realiza el almacenamiento de la IPseleccionada por el usuario, luego esta *IP* es leída por el script de python que se encuentra en la raspberry y a su vez lo almacena en un ".txt" para que el *script de bash* pueda utilizar este dato en la herramienta de *ethical hacking* correspondiente, despúes el resultado se vuelve a almacenar en txt para ser leído por el *script de bash* y almacenar este dato en la base de firebase, finalmente este dato es leído por el código de *AndroidStudio*, mostrado en una ventana del aplicativo.

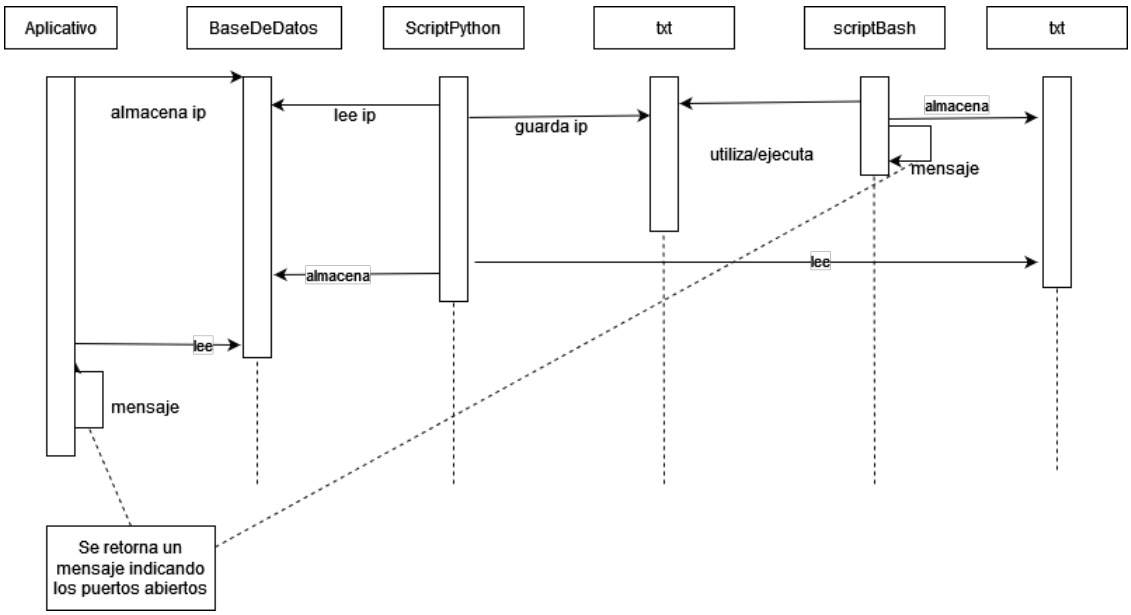

Figura 3.3: Diagrama de comunicación para obtener los puertos abiertos

3.4 DIAGRAMA DE COMUNICACIÓN - TIEMPO DE CONEXIÓN

En la figura 3.4, se muestra la manera en la que se obtiene el tiempo de conexión de los host en la red, ya que el script de python almacena en la base la fecha en la que el usuario se conecta y los hosts que estan conectados, para luego ser leidos por el aplicativo y mediante una función realizar la programación respectiva para obtener el tiempo que cada host se ha mantenido conectado a la red.

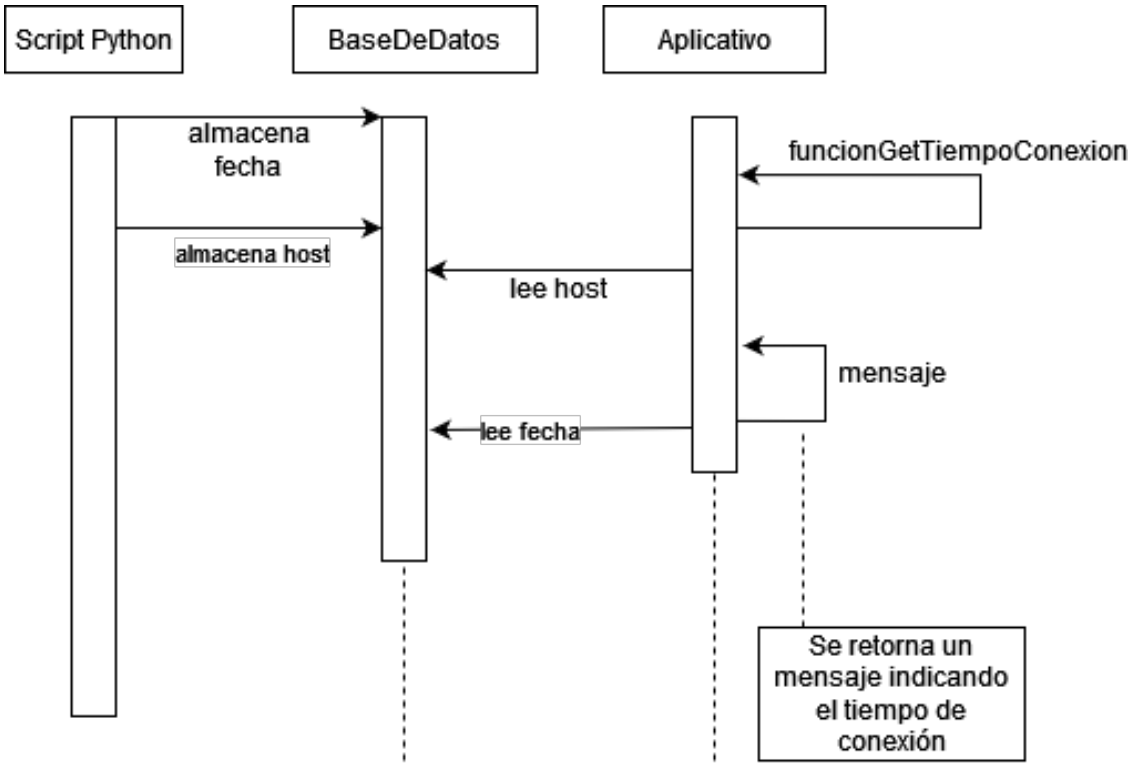

Figura 3.4: Diagrama de comunicación para obtener el tiempo de conexión de los host

3.5 DIAGRAMA DE COMUNICACIÓN - ANCHO DE BANDA

Como se muestra en la figura 3.5, para obtener el ancho de banda consumido, el aplicativo realiza el almacenamiento del mes, día u hora seleccionada por el usuario, luego estos datos son leídos por el script de python que se encuentra en la raspberry y a su vez lo almacena en ".txt" respectivos para que el *script de bash* pueda utilizar estos datos en la herramienta de *ethical hacking* correspondiente, despúes el resultado se vuelve a almacenar en txt para ser leído por el *script de bash* de separación de datos y almacenar esta información en la base de firebase, finalmente estos datos son leídos por el código de *AndroidStudio*, mostrados en una ventana del aplicativo a través de gráficos.

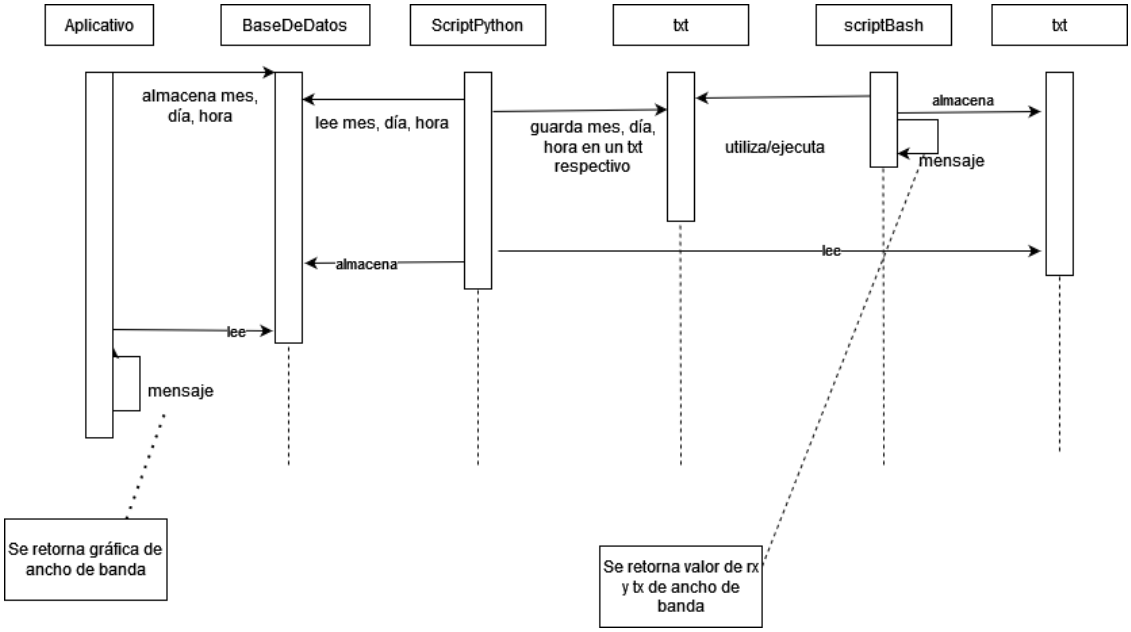

Figura 3.5: Diagrama de comunicación para obtener el ancho de banda consumido

3.6 DIAGRAMA DE COMUNICACIÓN PARA LA GENERACIÓN DE REPORTES

En la figura 3.6, se muestra la manera en la que se genera el reporte, el proceso comienza cuando el usuario selecciona si requiere generar el reporte de manera mensual o semanal, para luego de acuerdo a esta selección consultar en la base de datos y obtener la información requerida, luego de esto a través de una función en el código de *AndroidStudio* se genera el reporte de Excel con los datos respectivos.

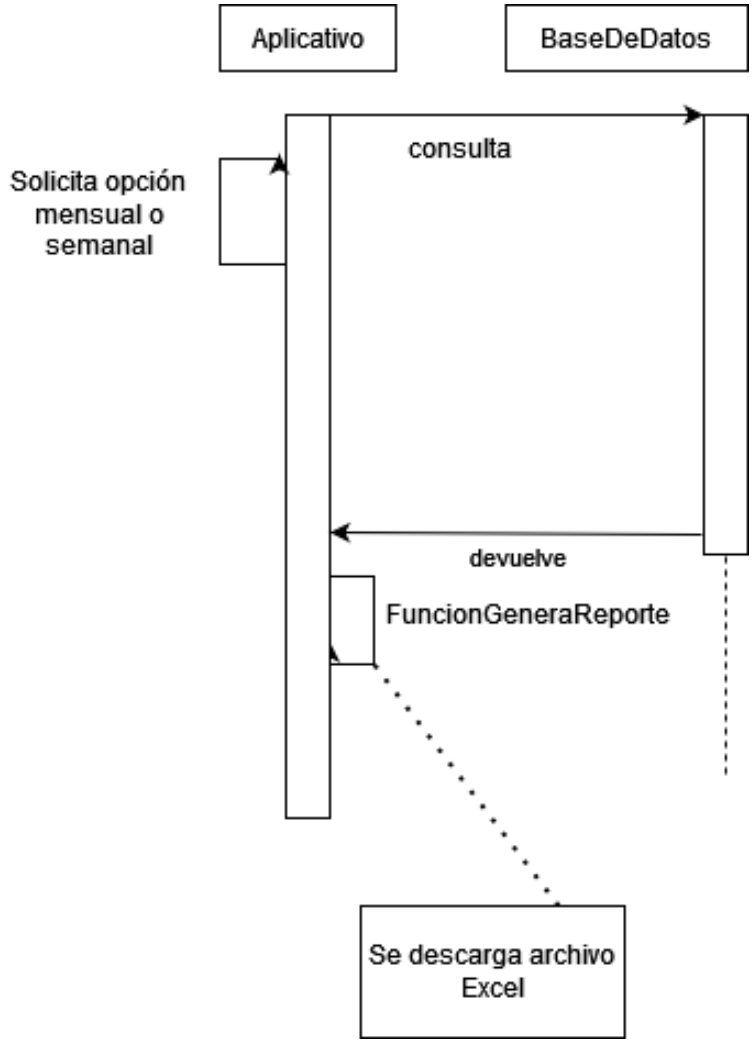

Figura 3.6: Diagrama de comunicación para generar reporte

3.7 IMPLEMENTACIÓN

3.7.1 HOST CONECTADOS

En el *código 3.1* se utiliza la herramienta de *nmap* de *Kali Linux* para realizar un scaneo a la red objetivo y guardar los datos en un *txt* y posteriormente obtener la información más relevante, adicional a esto el *script* se ejecuta cada segundo para un nuevo scaneo.

Código 3.1: Script de bash para obtener los host conectados a la red

```
// HostConectados.sh
#!/bin/bash
while true
do
    nmap 192.168.100.0/24
    sleep 1
done
```

En el *código 3.2* se comienza abriendo el *txt* donde se encuentran las *IP* que se han obtenido del *código 3.6* para luego crear el diccionario que se guardará en la base de datos de firebase. Este diccionario contiene la *IP*, hora de conexión y fecha de conexion que se guardará con la clave *IP*.

Código 3.2: Script de python que permite guardar a la base de datos de firebase la IPde cada host conectado a la red con la fecha en la que está conectado

```
// Conexion.py
def guardarIpFecha():
    with open('hosts3.txt') as archivo3:
        for linea3 in archivo3:
            datos3={
            "recursosTI":
            {
            "ip": linea3,
            "horaconexion": now.strftime("%H:%M:%S"),
            "fechaconexion": now.strftime("%d %m %Y")
            }
```

```
}
resultado3=db.reference("/recursosTI/ip")
for key,value in datos3.items():
    resultado3.push().set(value)
```

En el *código 3.3* se comienza abriendo el *txt* donde se encuentran las *IP* que se han obtenido del *código 3.6* para luego crear el diccionario que se guardará en la base de datos de firebase. Este diccionario contiene la *IP* que se guardará con la clave *host*.

Código 3.3: Script de python que permite guardar a la base de datos de firebase la IPde cada host conectado a la red

```
// Conexion.py
def guardarFirebasehost():
    with open('hosts3.txt') as archivo:
        for linea in archivo:
            datos={
            "recursosTI":
            {
            "host": linea
            }
            }
            dato={
            "hosts":linea
            }
            resultado = db.reference("/recursosTI/host")
            for key,value in datos.items():
                resultado.push().set(value)
```

En el *código 3.4* se comienza abriendo el *txt* donde se encuentra el total de host scaneados que se han obtenido del *código 3.6* para luego crear el diccionario que se guardará en la base de datos de firebase. Este diccionario contiene el total que se guardará con la clave *hostconectados*.

Código 3.4: Script de python que permite guardar a la base de datos de firebase el valor total de host conectados a la red

```
// Conexion.py
def guardarFirebasetotalhost():
    with open('totalhost.txt') as archivo2:
        for linea2 in archivo2:
            datos2={
            "recursosTI":
            {
            "total": linea2
            }
            }
            dato={
            "hosts":linea2
            }
            resultado2 = db.reference("/recursosTI/hostconectados")
            for key,value in datos2.items():
                resultado2.push().set(value)
```

En el *código 3.5* se comienza realizando la lectura del último valor correspondiente a la clave *ipSeleccionada*.

Código 3.5: Script de python que permite consultar la IPque el usuario seleccionó en el aplicativo

```
// Conexion.py
def consultaripSeleccionada():
    consultaTotal=consulta2.child("ipSeleccionada").order_by_key().limit_to_las(1).get()
    asx4=[]
    for b in consultaTotal:
        asx4.append(consultaTotal[b])
    lista = list (map(lambda v: v.strip(), asx4))
    lista .clear()
```

3.7.2 HOST CONECTADOS Y PUERTOS ABIERTOS

En el *código 3.6* se realiza un procesamiento de los datos obtenidos en los scripts de bash mencionados anteriormente, en donde se comienza realizando la lectura de los datos scaneados en el *código 3.1* que se encuentran almacenados en el *prueba.txt* para luego solo obtener las líneas que contengan la palabra *Host* y verificar que si la cantidad total de estas líneas es diferente de cero entonces se guarden en un archivo *totalhost.txt*, todo esto con la herramienta *grep* y después solo obtener la *IP* que se ha scaneado, esto con la herramienta *awk*. De la misma manera, se realiza la lectura de los datos scaneados en el *código 3.6*, en donde se realiza la misma comparación de que si las líneas que contienen la palabra *tcp* son diferentes de cero, entonces se almacenen en un archivo *puertosabiertos.txt*. Este script se ejecuta cada segundo para mantener actualizados los respectivos txt.

Código 3.6: Script de bash para filtrar la información recopilada

```
// FiltrarInformación.sh
#!/bin/bash

while true
do
archivo="/home/kali/Desktop/prueba.txt"
while IFS= read -r linea
do
 if [[ "$linea" == *"Host"* ]]
then
    echo "$linea"
 fi
done < "$archivo"
valor=`grep -w -c 'Host' "$archivo"`
 if [[ "$valor" != 0 ]]
then
    echo "El total de hosts conectados son"
    grep -w -c 'Host' "$archivo" > totalhost.txt
    grep -w 'Status:' "$archivo" > hosts2.txt
```

```
fi
archivo2="/home/kali/Desktop/puertos.txt"
while IFS= read -r linea2
do
if [[ "$valor2" != 0 ]]
then
    grep -w "tcp" "$archivo2" > puertosabiertos.txt
    grep -w -c "tcp" "$archivo2" > totalpuerto.txt
fi
done < "$archivo2"
sleep 1
done
```

En el *código 3.7* se realiza la conexión a la base de datos de firebase y se encuentran las librerías utilizadas en el script de python.

Código 3.7: Script de python librerias / credenciales

```
// Conexion.py
import firebase_admin
from firebase_admin import credentials
from firebase_admin import db
import os
import time
from datetime import datetime
```

3.7.3 PUERTOS ABIERTOS

En el *código 3.8* se comienza abriendo el *txt* donde se encuentran los puertos abiertos scaneados que se han obtenido del *código 3.3* para luego crear el diccionario que se guardará en la base de datos de firebase. Este diccionario contiene los puertos abiertos que se guardarán con la clave *puertosactivos*.

Código 3.8: Script de python que permite guardar a la base de datos de firebase los puertos abiertos del host seleccionado

```
// Conexion.py
def guardarpuertohost():
    with open('puertosabiertos.txt') as archivo5:
        for linea5 in archivo5:
            datos5={
            "recursosTI":
            {
            "puertosabiertos": linea5
            }
            }
            resultado5= db.reference("/recursosTI/puertosactivos")
            for key,value in datos5.items():
                resultado5.push().set(value)
```

En el *código 3.9* se comienza abriendo el *txt* donde se encuentra el total de puertos scaneados que se han obtenido del *código 3.3* para luego crear el diccionario que se guardará en la base de datos de firebase. Este diccionario contiene el total de puertos abiertos que se guardarán con la clave *totalpuertosactivos*.

Código 3.9: Script de python que permite guardar el total de puertos del host consultado

```
// Conexion.py
def guardartotalpuerto():
    with open('totalpuerto.txt') as archivo7:
        for linea6 in archivo7:
            datos8={
            "recursosTI":
            {
            "totalpuertos": linea6
            }
            }
            resultado7=db.reference("/recursosTI/totalpuertosactivos")
            for key,value in datos8.items():
                resultado7.push().set(value)
```

En el *código 3.10* se utiliza la herramienta de *nmap* de *Kali Linux* para realizar un

scaneo de los puertos que un determinado host tiene abierto, esto pasándole la *IP* objetivo que se encuentra en el *ipSeleccionada.txt* al comando para luego guardar el resultado en un txt y posteriormente obtener la información más relevante, adicional el script se ejecuta cada dos segundos para realizar un nuevo scaneo.

Código 3.10: Script de bash para obtener los puertos abiertos de un host

```
// PuertosAbiertos.sh
#!/bin/bash

while true
do
  archivo="/home/kali/Desktop/ipSeleccionada.txt"
  sudo nmap -sS -iL "$archivo"
  sleep 2
done
```

3.7.4 ANCHO DE BANDA

En el *código 3.11* se comienza realizando la lectura del último valor correspondiente a la clave *mesSeleccionado*.

Código 3.11: Script de python que permite consultar el mes seleccionado por el usuario

```
// Conexion.py
def consultarmes():
    consultaTotal=consulta2.child("mesSeleccionado").order_by_key().limit_to_last(1).get()
    asx4=[]
    for b in consultaTotal:
        asx4.append(consultaTotal[b])
    lista = list (map(lambda v: v.strip(), asx4))
    primervalor= lista [0]
    return primervalor
    lista .clear()
```

En el *código 3.12* se comienza realizando la lectura del último valor correspondiente

a la clave *diaSeleccionado*.

Código 3.12: Script de python que permite consultar el dia seleccionado por el usuario

```
// Conexion.py
def consultardia():
    consultaTotal=consulta2.child("diaSeleccionado").order_by_key().limit_to_last(1).get()
    asx4=[]
    for b in consultaTotal:
        asx4.append(consultaTotal[b])
    lista = list (map(lambda v: v.strip(), asx4))
    primervalor= lista [0]
    return primervalor
    lista .clear()
```

En el *código 3.13* se comienza realizando la lectura del último valor correspondiente a la clave *horaSeleccionada*.

Código 3.13: Script de python que permite consultar la hora seleccionada por el usuario

```
// Conexion.py
def consultarhora():
    consultaTotal=consulta2.child("horaSeleccionada").order_by_key().limit_to_last(1).get()
    asx4=[]
    for b in consultaTotal:
        asx4.append(consultaTotal[b])
    lista = list (map(lambda v: v.strip(), asx4))
    primervalor= lista [0]
    return primervalor
    lista .clear()
```

En el *código 3.14* se guarda los valores de ancho de banda de un mes determinado en la clave *anchobandames*.

Código 3.14: Script de python que permite guardar valores de ancho de banda de un mes determinado

```
// Conexion.py
#Permite guardar a la base de datos de firebase los valores de rx AB del mes
```

```
def guardarabmes():
    with open('rx.txt') as archivo6:
                                    for linea6 in archivo6:
                                        for linea7 in archivo7:
                                            for linea8 in archivo8:
                                                for linea9 in archivo9:
                                                    for linea10 in archivo10:
                                                        for linea11 in archivo11:
                                                            datos6={
                                                            "recursosTI":
                                                            {
                                                            "rx": linea6,
                                                            "rxmedida": linea7,
                                                            "tx": linea8,
                                                            "txmedida": linea9,
                                                            "avg": linea10,
                                                            "avgmedida": linea11
                                                            }
                                                            }
                                                            resultado5=

                                                            db.reference("/recursosTI/anchobandames")
                                                            for key,value in datos6.items():
                                                                resultado5.push().set(value)
```

En el *código 3.15* se guarda los valores de ancho de banda de un dia determinado en la clave *anchobandadia*.

Código 3.15: Script de python que permite guardar valores de ancho de banda de un dia determinado

```
// Conexion.py
#Permite guardar a la base de datos de firebase los valores de rx AB del día
def guardarabdia():
    with open('rxdia.txt') as archivo6:
                                    for linea6 in archivo6:
```

```
                for linea7 in archivo7:
                    for linea8 in archivo8:
                        for linea9 in archivo9:
                            for linea10 in archivo10:
                                for linea11 in archivo11:
                                    datos6={
                                    "recursosTI":
                                    {
                                    "rx": linea6,
                                    "rxmedida": linea7,
                                    "tx": linea8,
                                    "txmedida": linea9,
                                    "avg": linea10,
                                    "avgmedida": linea11
                                    }
                                    }
                                    resultado5=

                                    db.reference("/recursosTI/anchobandadia")
                                    for key,value in datos6.items():
                                        resultado5.push().set(value)
```

En el *código 3.16* se guarda los valores de ancho de banda de una hora determinada en la clave *anchobandahora*.

Código 3.16: Script de python que permite guardar valores de ancho de banda de una hora determinada

```
// Conexion.py
#Permite guardar a la base de datos de firebase los valores de rx AB por hora
def guardarabhora():
    with open('rxhora.txt') as archivo6:
                for linea6 in archivo6:
                    for linea7 in archivo7:
                        for linea8 in archivo8:
                            for linea9 in archivo9:
```

```
for linea10 in archivo10:
    for linea11 in archivo11:
        datos6={
        "recursosTI":
        {
        "rx": linea6,
        "rxmedida": linea7,
        "tx": linea8,
        "txmedida": linea9,
        "avg": linea10,
        "avgmedida": linea11
        }
        }
        resultado5=

        db.reference("/recursosTI/anchobandahora")
        for key,value in datos6.items():
            resultado5.push().set(value)
```

En el *código 3.17* se utiliza la herramienta de *vnstat* de *Kali Linux* para realizar la obtención de los valores de ancho de banda por mes, dia y hora de la interfaz wlan0.

Código 3.17: Script de bash para obtener el ancho de banda por mes, dia u hora

```
// anchodebandacomando.sh
#!/bin/bash
while true
do
    vnstat -m -i wlan0 > mesab.txt
    vnstat -h -i wlan0 > horaab.txt
    vnstat -d -i wlan0 > diaab.txt
    sleep 30
done
```

En el *código 3.18* se obtienen los datos de ancho de banda de la hora seleccionada por el usuario en la aplicación, además de realizar la separación de la información relevante

y guardarla en ”txt” respectivos.

Código 3.18: Script de bash para obtener el ancho de banda por hora de acuerdo a la seleccionada por el usuario

```
// obtenerabhora.sh
#!/bin/bash
while true
do
archivo2=”/home/kali/Desktop/archivo4.txt”
while IFS= read -r line
do
valor=‘grep -w -c ”$line” ”horaab.txt”‘
if [[ ”$valor” != 0 ]]
then
    grep -w ”$line” ”horaab.txt” > horaabtodo.txt
     tail -1 ”horaabtodo.txt” > horaabseparado.txt
    rx=‘awk ’{print $2}’ ”/home/kali/Desktop/horaabseparado.txt”‘
    medida=‘awk ’{print $3}’ ”/home/kali/Desktop/horaabseparado.txt”‘
    tx=‘awk ’{print $5}’ ”/home/kali/Desktop/horaabseparado.txt”‘
    medida2=‘awk ’{print $6}’ ”/home/kali/Desktop/horaabseparado.txt”‘
    avg=‘awk ’{print $11}’ ”/home/kali/Desktop/horaabseparado.txt”‘
    avgmedida=‘awk ’{print $12}’ ”/home/kali/Desktop/horaabseparado.txt”‘

fi
done < ”$archivo2”
sleep 2
done
```

En el *código 3.19* se obtienen los datos de ancho de banda del mes seleccionado por el usuario en la aplicación, además de realizar la separación de la información relevante y guardarla en ”txt” respectivos.

Código 3.19: Script de bash para obtener el ancho de banda por mes de acuerdo al seleccionado por el usuario

```
// obtenerabmes.sh
```

```
#!/bin/bash
while true
do
archivo2="/home/kali/Desktop/archivo2.txt"
while IFS= read -r line
do
valor=`grep -w -c "$line" "mesab.txt"`
 if [[ "$valor" != 0 ]]
then
    grep -w "$line" "mesab.txt" > mesabseparado.txt
    rx=`awk '{ print $2}' "/home/kali/Desktop/mesabseparado.txt"`
    medida=`awk '{print $3}' "/home/kali/Desktop/mesabseparado.txt"`
    tx=`awk '{ print $5}' "/home/kali/Desktop/mesabseparado.txt"`
    medida2=`awk '{print $6}' "/home/kali/Desktop/mesabseparado.txt"`
    avg=`awk '{ print $11}' "/home/kali/Desktop/mesabseparado.txt"`
    avgmedida=`awk '{print $12}' "/home/kali/Desktop/mesabseparado.txt"`
 fi
done < "$archivo2"
sleep 2
done
```

En el *código 3.20* se obtienen los datos de ancho de banda del dia seleccionado por el usuario en la aplicación, además de realizar la separación de la información relevante y guardarla en "txt" respectivos.

Código 3.20: Script de bash para obtener el ancho de banda por dia de acuerdo al seleccionado por el usuario

```
// obtenerabdia.sh
#!/bin/bash
while true
do
archivo2="/home/kali/Desktop/archivo3.txt"
while IFS= read -r line
do
valor=`grep -w -c "$line" "diaab.txt"`
```

```
if [[ "$valor" != 0 ]]
then
    grep -w "$line" "diaab.txt" > diaabseparado.txt
    rx=`awk '{print $2}' "/home/kali/Desktop/diaabseparado.txt"`
    medida=`awk '{print $3}' "/home/kali/Desktop/diaabseparado.txt"`
    tx=`awk '{print $5}' "/home/kali/Desktop/diaabseparado.txt"`
    medida2=`awk '{print $6}' "/home/kali/Desktop/diaabseparado.txt"`
    avg=`awk '{print $11}' "/home/kali/Desktop/diaabseparado.txt"`
    avgmedida=`awk '{print $12}' "/home/kali/Desktop/diaabseparado.txt"`
fi
done < "$archivo2"
sleep 2
done
```

CAPÍTULO 4

4. ANÁLISIS DE RESULTADOS Y EVALUACIÓN DE MÉTRICAS

4.1 ANÁLISIS DE RESULTADOS

Una vez que se ha realizado la obtención de los datos fundamentales para la implementación del sistema de monitoreo de recursos TI, se obtienen los resultados descritos a continuación de la aplicación móvil y de la base de datos de firebase.

4.1.1 APLICACIÓN MÓVIL

Se implementó el sistema de monitoreo de recursos TI a través de una aplicación móvil realizada con Android Studio.

En la figura 4.1 se muestra la ventana de inicio de la aplicación, en donde se tienen dos opciones, las cuales son el laboratorio de redes de datos y el laboratorio de sistemas telemáticos.

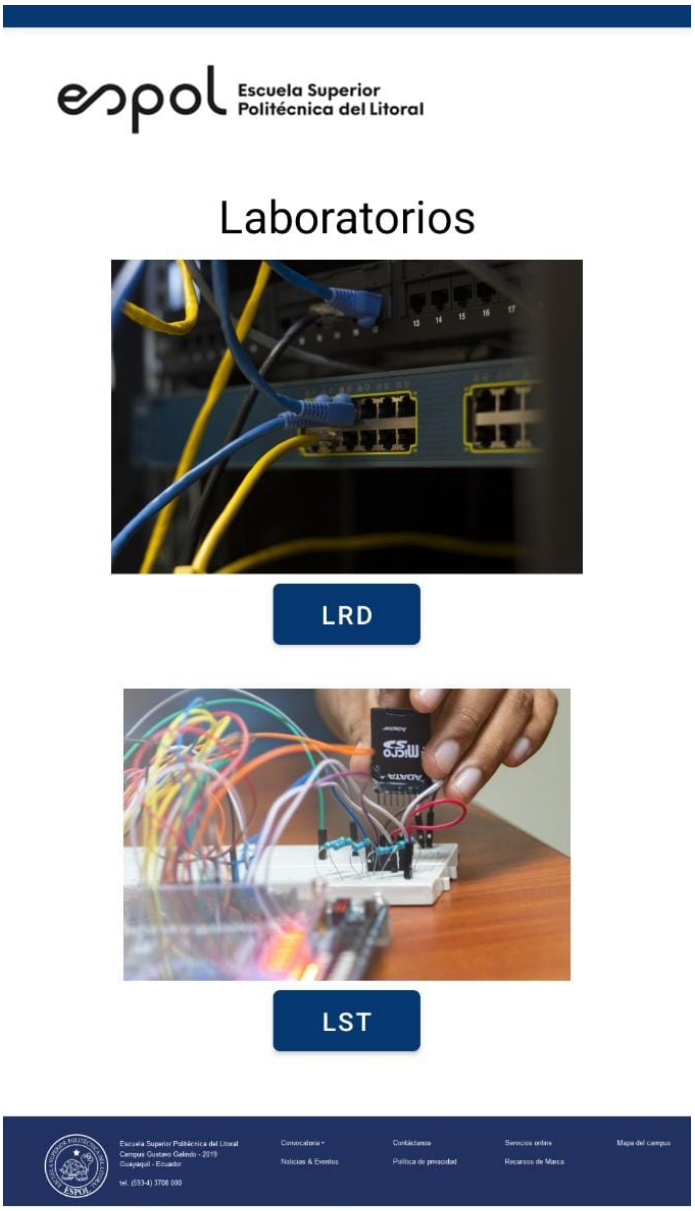

Figura 4.1: Ventana de inicio de aplicativo

Una vez que el usuario haya seleccionado un laboratorio para realizar el respectivo monitoreo, va a observar la siguiente ventana como se muestra en la figura 4.2, en donde se observan las ip de los host conectados, así como un gráfico con la cantidad total de host que estuvieron y están conectados actualmente, además una sección para visualizar el ancho de banda por día, mes u hora, así como también una opción para generar un reporte.

Figura 4.2: Ventana de monitoreo de host conectados

Adicional, en el sistema de monitoreo está la opción para visualizar los puertos abiertos de un determinado host, así como el tiempo de conexión del mismo, tal como se muestra en la figurar 4.3.

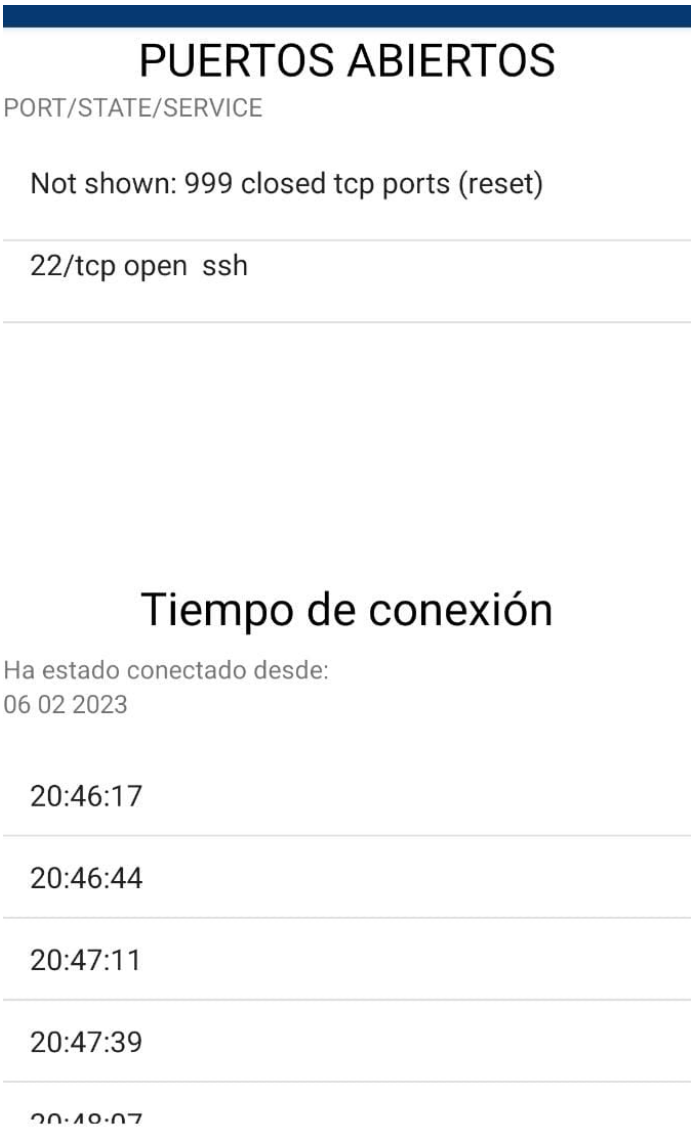

Figura 4.3: Ventana de puertos abiertos y tiempo de conexión de un determinado host

Por otro lado, también existe la opción de visualizar el ancho de banda consumido durante un día determinado, así como se muestra en la figura 4.4, en la cual se puede visualizar el ancho de banda rx, tx y avg a través de una gráfica, así como también se visualizan los valores respectivos como se muestra en la figura 4.5. Adicional, en las figuras 4.5, 4.7 y 4.9 se observarán los gráficos de ancho de banda, en donde se tienen los valores rx y tx (tasa de datos de transmisión y recepción), así como el valor de avg (valor promedio total de ancho de banda consumido de acuerdo con la selección del usuario). Además, en las tres gráficas se tienen valores iniciales de 0, 1 y a continuación el valor de rx, tx y avg respectivamente de acuerdo con el color que poseen los ejes. Los valores de 0 y 1 son para fines de muestra de una mejor graficación.

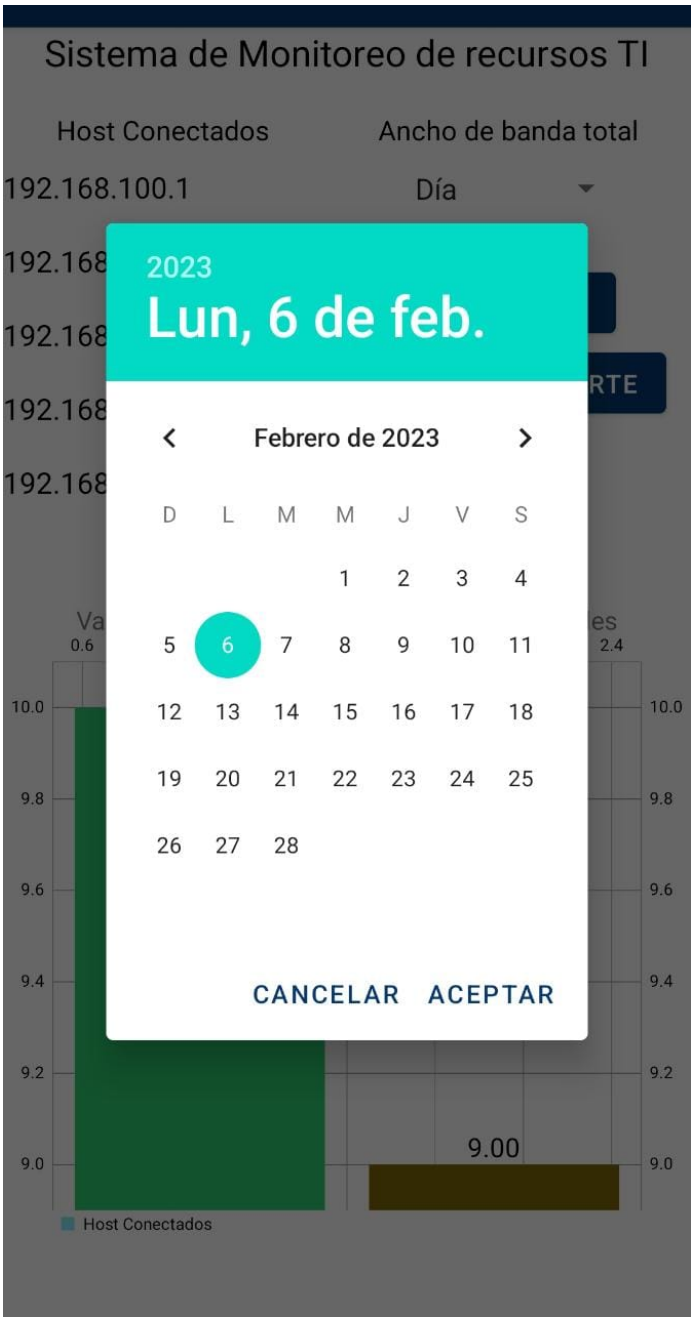

Figura 4.4: Opción para visualizar ancho de banda por día

Figura 4.5: Ancho de banda por día - rx, tx y avg

De la misma manera, también existe la opción de visualizar el ancho de banda consumido durante un mes determinado, así como se muestra en la figura 4.6, en la cual se puede visualizar el ancho de banda rx, tx y avg a través de una gráfica, así como también se visualizan los valores respectivos como se muestra en la figura 4.7.

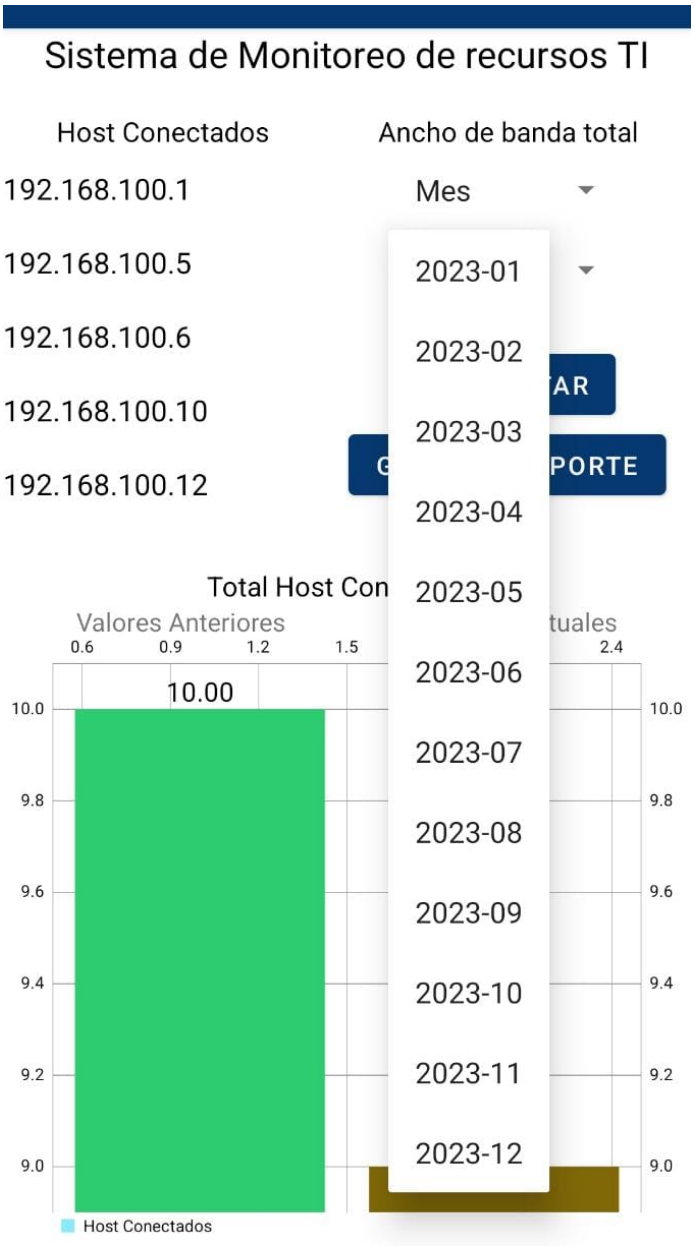

Figura 4.6: Opción para visualizar ancho de banda por mes

Figura 4.7: Ancho de banda por mes - rx, tx y avg

De la misma manera, también existe la opción de visualizar el ancho de banda consumido durante una hora determinada, así como se muestra en la figura 4.8, en la cual se puede visualizar el ancho de banda rx, tx y avg a través de una gráfica, así como también se visualizan los valores respectivos como se muestra en la figura 4.9.

Figura 4.8: Opción para visualizar ancho de banda por hora

Figura 4.9: Ancho de banda por hora - rx, tx y avg

4.1.2 BASE DE DATOS

Con respecto a la base de datos, se utilizó la base realtime de firebase, de esta manera a continuación se muestra en la figura 4.10 el consumo de la misma durante el periodo de pruebas de la presente investigación, especificamente en el mes de Febrero.

Figura 4.10: Consumo de base de datos en mes de Febrero

4.2 CASOS DE ESTUDIO

- **ADMINISTRADOR**

  En la facultad de ingeniería en electricidad y computación, para el desarrollo de las sesiones prácticas de materias como redes de datos, internetworking, redes inalámbricas y de sensores (RIS), sistemas en la nube, evaluación y simulación de redes (ESR), que corresponden a la carrera de Telemática, los jefes de los laboratorios vigentes de redes de datos (LRD) y sistemas telemáticos (LST) respectivamente, disponían de una plataforma IoT creada inicialmente en semestres previos para el control y manipulación inteligente de dispositivos conectados de forma inalámbrica en la red local asignada, pero al observar que para

las asignaturas anteriores mencionadas se presentaban inconvenientes durante la realización de prácticas con los ordenadores y dispositivos móviles conectados vía inalámbrica por un incremento de flujo en la información generada y a su vez porque desconocían si todos los estudiantes se encontraban desarrollando su trabajo. Por ello, los jefes determinaron nuevos requerimientos en el módulo que utlizaban para la administración, control y monitoreo de los recursos TI.

De esta forma, con el despliegue del módulo de administración en una aplicación móvil para cada titular del laboratorio, disponen de una mejora importante en el control de la cantidad de dispositivos conectados en el lapso de la sesión, así como el ancho de banda total que se consume en cada práctica y el estado de los puertos en los dispositivos u ordenadores. A tal efecto, se ejecutaron pruebas en el transcurso de una clase en cada laboratorio obteniendo el total de host conectados desarrollando la práctica, el ancho de banda total consumido y sus puertos respectivamente, donde el administrador al finalizar la semana de prácticas puede consultar el reporte generado de forma automatizada en el período que disponga visualizar, así como el monitoreo en tiempo real del consumo de los recursos en cada sesión de los laboratorios proyectando un mejor control y disponibilidad de los presentes.

- **USUARIO**

En el marco del incremento de los estudiantes registrados para cada materia pertinente a la carrera de Telemática, quienes son los que acceden a los dispositivos para la realización de sus prácticas semanales. Contando con una asistencia promedio de diez a doce alumnos por cada sesión diferente, reportaban inconsistencias por fallas de conectividad, retrasos en respuestas hacia los ordenadores durante el desarrollo y problemas de recursos de red. Sin embargo, se desconocían la causa de los obstáculos presentados en cada área de trabajo, por lo que no contaban con una herramienta o una forma de poder notificar a los jefes de los laboratorios y a su vez de forma bidireccional, no contaban con un módulo que administre el laboratorio. Enfatizando la responsabilidad de que cada estudiante debe mantenerse en la realización de su tarea, mas no en otras distracciones como páginas de redes sociales o contenido multimedia lo cual conlleva a un mayor consumo de recursos disponibles y provocando una disminución en la productividad

de los compañeros presentes.

Así, para cada grupo de estudiantes que disponen del uso de los laboratorios se notificó e informó la implementación de un módulo para el respectivo monitoreo de recursos TI que se generaba durante cada sesión práctica. Lo que mejoró significativamente la cuantificación de los dispositivos que se encontraban activos y trabajando, el ancho de banda consumido en el desarrollo y el estado de los puertos que representan un aviso a nivel de seguridad informática para los protocolos y servicos que se encuentren utilizando y permitiendo a los alumnos reportar mediante formularios los problemas que se presenten hacia su administrador, llevando un mejor control del ordenador utilizado en el lapso de la clase y logrando identificar los requerimientos en infraestructura digital y conectividad en cada materia. A su vez, permitiendo incrementar los recursos TI para los trabajos de laboratorios que lo precisen.

4.3 EVALUACIÓN DE LAS MÉTRICAS DEL SISTEMA

- **CUELLO DE BOTELLA**

  Para la implementación en la recopilación de datos se utilizó dos modelos del microcontrolador raspberry pi 3 y raspberry pi 4 respectivamente, que determinaron la sobrecarga de trabajo al utilizarse como un proxy en la comunicación entre los ordenadores y el router principal de cada laboratorio. Cabe resaltar que este inconveniente se produjo en ambos modelos, al conectarse más de cinco dispositivos de forma inalámbrica lo que imposibilita realizar las tareas de comunicación con la base de datos y posterior con la conexión en la aplicación móvil.

  Esto ocasionó que se sustituya la funcionalidad de la raspberry como proxy, de tal forma que se aplicó mediante el sistema operativo kali linux la utilidad de hacking ético que dispone y reduciendo en su totalidad el cuello de botella provocado inicialmente. Ambos modelos se configuraron con esta herramienta notando una mejora considerable en el sistema de monitoreo.

- **TIEMPO QUE TARDARÁ EN EJECUTARSE EL SCRIPT PARA OBTENER LOS DATOS**

En relación con el tiempo de ejecución de los scripts de bash, el tiempo promedio que tardaron fue de 5 segundos en la raspberry pi 3 y de 3 segundos en la raspberry pi 4 esto debido al mejor procesador y mayor cantidad de memoria disponible para el modelo posterior.

- **TIEMPO QUE TARDARÁ EN OBTENERSE LOS DATOS AL REALIZAR EL SNIFFING**

  En cuanto al tiempo en el que se obtuvo la recopilación de datos, tardó un promedio de 6 segundos para la raspberry pi 3 y de 4 segundos para el modelo vigente. Posterior a ello se procede con la conexión hacia la base de datos de firebase para el monitoreo correspondiente desde la aplicación móvil.

- **CANTIDAD DE DATOS QUE LA BASE DE DATOS PUEDE ALMACENAR POR DÍA**

  Para el almacenamiento de datos disponible de forma gratuita su capacidad es de 1GiB, permitiendo operaciones de lectura de documentos de 50000 por día y un límite de 200 solicitudes, por lo que el administrador será el único que puede consultar cuando disponga del monitoreo en tiempo real considerando los parámetros, conexiones y las veces que puede realizar la correspondiente consulta de reportes de recursos TI.

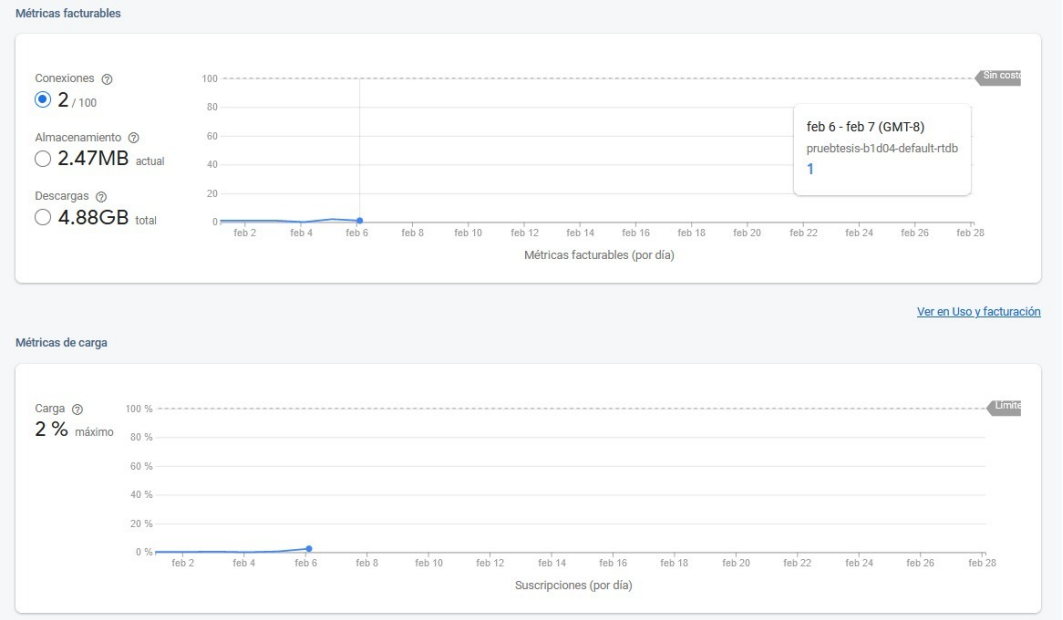

Figura 4.11: Métricas de almacenamiento y carga de la base de datos en tiempo real Firebase

- **TIEMPO QUE TARDARÁ EN EJECUTARSE EL APLICATIVO AL USUARIO**

  En la ejecución de la aplicación móvil desde un dispositivo Android se toma un tiempo promedio de 3 segundos para iniciar la pantalla principal que muestra los laboratorios disponibles para el respectivo monitoreo de recursos TI en tiempo real.

4.4 EVALUACIÓN DE LAS MÉTRICAS DE SATISFACCIÓN

- **ADMINISTRADOR**

  Una vez aplicados los instrumentos de recolección de la información, se procedió a hacer el método correspondiente para la indagación de estos, por cuanto, la información que se presentará a continuación ayudará a conocer los requerimientos de los jefes de laboratorio con respecto al sistema de monitoreo de recursos TI, por cuanto mostrará la percepción que tiene cada jefe de laboratorio de la carrera de ingeniería en telemática con respecto a las características del sistema a implementar.

  Con relación a si existe un sistema de monitoreo de recursos TI en los laboratorios de Telemática, se aprecia que el 100% de los encuestados indicó que no, de la misma manera el 100% aseveró que les gustaría que se implementara un sistema de monitoreo de recursos TI.

  Como segundo punto, se sitúa con un 33.3% la frecuencia con la que usarían el sistema de monitoreo de manera semanal y por último con un 66.7% la frecuencia de uso de manera diaria.

  Como tercer punto, se sitúa con un 100% la latencia, como recurso TI que a los jefes de laboratorio les gustaría observar en el sistema de monitoreo, por otro lado con un 100% el ancho de banda, así también con un 66.7% el tiempo de conexión de los hosts, de la misma manera con un 100% los puertos abiertos de un determinado host, adicional con un 100% la cantidad de hosts conectados y con un 33.3% como recuersos TI adicionales que quisieran que se muestren en el sistema de monitoreo se tienen los puertos más usados, puertos cerrados y estadística de tráfico por servicios.

Como cuarto punto, se sitúa con un 66.7% el método de generación de reportes automatizados semanales y con un 33.3% el método de generación de reportes mensuales.

Como quinto punto, se sitúa con un 66.7% la facilidad para manejar la app como característica principal del sistema de monitoreo, así también le sigue con un 66.7% la rápidez, de la misma manera con un 33.3% la liviandad de la aplicación y con un 66.7% el diseño del sistema.

Como sexto y último punto, se sitúa con un 100% la afirmación de que la implementación del sistema de monitoreo de recursos TI ayudaría a llevar un mejor control de los laboratorios de la carrera de ingeniería en telemática.

- **USUARIO**

Una vez aplicados los instrumentos de recolección de la información, se procedió a hacer el método correspondiente para la indagación de estos, por cuanto, la información que se presentará a continuación ayudará a conocer la opinión de los estudiantes que utilizan los laboratorios de ingeniería en telemática con respecto al sistema de monitoreo de recursos TI, por cuanto mostrará la percepción que tiene cada alumno de la carrera de ingeniería en telemática con respecto a las características del sistema a implementar.

Con relación a si conocen si existe un sistema de monitoreo de recursos TI en los laboratorios de Telemática, se aprecia que el 92.3% de los encuestados indicó que no, mientras que el 7.7% indicó que si.

De acuerdo con el 7.7%, el 50% indica que le es indiferente el sistema de monitoreo, mientras que el otro 50% indica que se siente muy cómodo con esta herramienta. Adicional, el 100% indica que no sienten que el sistema invade su privacidad.

De acuerdo con el 92.3% que desconoce si existe un sistema de monitoreo de recursos TI en los laboratorios de Telemática, el 87.5% indica que estaría de acuerdo con que se implemente este sistema, adicional el 12.5% indica que tal vez estarían de acuerdo; es decir, no se encuentran convencidos de su implementación.

Como segundo punto, el 12.5% indica que les es indiferente la implementación del sistema, por otro lado el 8.3% no se preocupa por la implementación, así también el

16.7% indica que se sentirían medianamente cómodos con el sistema de monitoreo, finalmente el 62.5% se sentiría muy cómodo con esta herramienta.

Como tercer punto, el 25% indica que no sentirían que la herramienta de monitoreo invade su privacidad, así también el 8.3% indica que le es indiferente la implementación del sistema, adicional el 37.5% indica que no le preocupa si el sistema invade su privacidad, ya que consideran que es para fines de seguridad, además el 25% indica que sentirían que se invade un poco su privacidad y finalmente, el 4.2% indica que sentirían que invaden demasiado su privacidad, pero es una mínima parte de la muestra tomada para la presente encuesta.

CAPÍTULO 5

5. CONCLUSIONES, RECOMENDACIONES Y LINEAS FUTURAS

5.1 CONCLUSIONES

- En la administración de los laboratorios se desplegó un sistema de monitoreo de recursos TI mediante una aplicación móvil considerando los requerimientos y la elección de herramientas de desarrollo en software libre incrementando la escalabilidad en el diseño y la funcionalidad para los dispositivos.

- Se determinó la cantidad de dispositivos conectados de forma inalámbrica, mostrando en el aplicativo la IP correspondiente a la red de cada laboratorio y el tráfico consumido durante la sesión práctica con un promedio de 12 estudiantes y un valor de 8.56 MB/s requeridos de forma general para un completo desarrollo de las actividades.

- Se diseñó una aplicación móvil Android con los parámetros establecidos por el jefe de laboratorio que permitieron el monitoreo, control de métricas solicitadas y mejora de seguridad informática a nivel de detección de puertos abiertos en tiempo real de los laboratorios de Telemática.

- Se realizaron pruebas durante sesiones prácticas ordinarias y sin usuarios a la vez, validando la cantidad de recursos TI consumidos en distintos contextos que se dispone cada laboratorio para la generación automatizada de los reportes semanales y mensuales en la aplicación móvil.

5.2 RECOMENDACIONES

Una vez que se ha concluido la tesis, se propone indagar sobre otras funcionalidades relacionadas con el monitoreo de recursos TI:

- Trabajar en mejorar la presentación del tiempo de conexión, para que se pueda mostrar la hora en la que se conextó y la hora en la que se desconectó, además de que se pueda sacar un promedio de cuanto tiempo ese host ha estado conectado al día, semana o mes.
- Extender la investigación expuesta en esta tesis al campo de ciberseguridad, para implementar más opciones de monitoreo para protección de la red.
- Analizar con mayor detenimiento los resultados expuestos en la presente tesis para lineas futuras de investigación.
- Extender el trabajo propuesto a diferentes áreas, tales como hogar, negocio, empresas, instituciones educativas como colegios u escuelas.

5.3 LINEAS FUTURAS

Como futuras líneas de investigación se encuentran funcionalidades que pueden contribuir a un manejo más eficaz de los recursos TI de los laboratorios de Telemática, las mismas que pueden analizarse en investigaciones posteriores, debido a que no se encuentran contempladas dentro de los objetivos planteados inicialmente.

Por otro lado, como se ha mencionado anteriormente, los nuevos análisis de las posibles funcionalidades a implementarse en el sistema de monitoreo, podrán ser soluciones a nuevos problemas que puedan ocurrir con el monitoreo de recursos TI, que hasta el presente proyecto no se han presentado, por lo que se presenta a continuación la lista de futuras investigaciones.

- Es necesario realizar la investigación respectiva para poder implementar al sistema de monitoreo actual la funcionalidad de detectar los puertos que alguna aplicación esté intentando utilizar, pero que esté bloqueado por la red general de ESPOL, ya que esto permitiría al Jefe de Laboratorio solicitar fácilmente la liberación de ese puerto específico al Departamento de Soporte Técnico (DST) de FIEC.

- Por otro lado, también es necesario investigar la posibilidad de implementar la funcionalidad de poder bloquear puertos y ejecutar acciones desde el sistema de monitoreo, como desconectar host, limitar ancho de banda, entre otros.
- Así también, se requiere investigar la viabilidad del sistema de monitoreo para otros laboratorios de la facultad de FIEC, así como del resto de facultades de la ESPOL.
- Adicional, se debe analizar la posibilidad de implementar la opción de visualizar puertos más usados, puertos cerrados y estadística de tráfico por servicios.

BIBLIOGRAFÍA

[1] N. CEPAL, “La nueva revolución digital: de la Internet del consumo a la Internet de la producción,” 08 2016.

[2] V. H. G. Tomala, “Control y monitoreo en tiempo real de los equipos informáticos para los laboratorios 1-2-3 de informática de la Universidad Estatal Península de Santa Elena,” 11 2020.

[3] G. Junco Romero and S. Rabelo Padua, “Los recursos de red y su monitoreo,” *Revista Cubana de InformÃtica MÃ*, vol. 10, pp. 76 – 83, 06 2018.

[4] M. d. P. Sánchez, A. Estrada, and W. Rojas, “Modelo de costos de servicios de tecnologías de información en instituciones de educación superior,” *TICAL*, no. 6, 2016.

[5] C. de Pablos Heredero, J. J. L. H. Agius, S. M.-R. Romero, and S. M. Salgado, *Organización y transformación de los sistemas de información en la empresa*. esic, 2019.

[6] E. G. G. N. Bryan Orlando Pilco Montero, “Integración de módulos IoT con Sistemas Hardware/Software para Edificios Inteligentes mediante la Estandarización de Datos de BrickSchema.,” 2022.

[7] A. B. Gonzáles, “Gestión de proyectos de software,” 2017.

[8] J. Hernantes, G. Gallardo, and N. Serrano, “It infrastructure-monitoring tools,” *IEEE Software*, vol. 32, no. 4, pp. 88–93, 2015.

[9] M. Coşar and H. E. Kiran, “Measurement of raspberry pi performance in network traffic analysis,” in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2018.

[10] P. Shinde, A. Karve, P. Mandaliya, and S. Patil, “Wireless security audit penetration test using raspberry pi,” in *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, pp. 1–4, 2018.

[11] T. Q. P. Andres, “SISTEMA OPEN SOURCE CON CONEXIÓN A LA PLATAFORMA CLOUDIOT PARA EL MONITOREO DE PROGRAMAS INFORMÁTICOS EN LOS LABORATORIOS DE LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL.,” 06 2019.

[12] D. Negi, A. Kumar, P. Kadam, and B. N. Savant, “Smart harvest analysis using raspberry pi based on internet of things,” in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pp. 1–5, 2018.

[13] S. Gunde, A. K. Chikaraddi, and V. P. Baligar, “Iot based flow control system using raspberry pi,” in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pp. 1386–1390, 2017.

[14] N. P. Kumar and R. K. Jatoth, “Development of cloud based light intensity monitoring system using raspberry pi,” in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 1356–1361, 2015.

[15] G. Suciu, S. Halunga, A. Ochian, and V. Suciu, “Network management and monitoring for cloud systems,” in *Proceedings of the 2014 6th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1–4, 2014.

[16] A. Velasco and G. Cagua, “ImplementaciÓn de un sistema de monitoreo de redes utilizando herramientas open source y proveer servicios de directorio a travÉs de active directory en la facultad de filosofÍa, letras y ciencias de la educaciÓn de la universidad de guayaquil,” 2017.

[17] J. Guerrero, “DiseÑo e implementaciÓn de un sistema de monitoreo a la red de datos de entidad prestadora del servicio de salud.,” 2020.

[18] M. Gallego, “RediseÑo e implementaciÓn del sistema de monitoreo de la red de telecomunicaciones de distribuidora nissan s.a,” 2015.

[19] A. Bravo, A. Villafuerte, and J. Patiño, “Implantación de una herramienta ossim para el monitoreo y gestión de la seguridad de la red y plataformas windows y linux aplicado a empresas medianas,” 2015.

[20] W. F. Pilaló, “Monitorización mediante GRAFANA de la seguridad de una red de computadoras para mitigar las vulnerabilidades en el tráfico de datos.,” 2021.

[21] V. D. Ríos Villacorta, Alberto Vasco Cabrera, “Plataforma de control y monitoreo del equipamiento de laboratorios basado en tecnología RFID sobre una arquitectura Cloud Computing,” 02 2018.

[22] F. E. Aguilar Gavilanes, José Andrés Villavicencio Ramos, “Implementación de un sistema de monitoreo para el control de la planta B del laboratorio LACTI de la Universidad Politécnica Salesiana – UPS sede Cuenca a través de servicios en la nube,” 11 2021.

[23] C. P. V. A. Narváez Narváez Karen Stefany, “DISEÑO Y DESARROLLO DE UN PROTOTIPO DE RED DE SENSORES IOT UTILIZANDO TECNOLOGÍA LORAWAN PARA EL MONITOREO DE PARÁMETROS AMBIENTALES EN INTERIORES Y EXTERIORES,” 2020.

[24] W. T. A. J. Salcedo Sambachi Daniela Aracely, “Implementación, administración y monitoreo de una red corporativa simulada en el Laboratorio de Redes Virtual de la Universidad de las Fuerzas Armadas ESPE sede Latacunga mediante un servidor Zabbix,” 09 2021.

APÉNDICES

A Encuesta a Jefes de laboratorio y estudiantes la carrera de ingeniería en Telemática.

A.1 SISTEMA DE MONITOREO DE RECURSOS TI - JEFES DE LABORATORIO

Formulario de carácter investigativo para recopilar información real con respecto a las métricas de usuario del proyecto integrador de la carrera de ingeniería en Telemática de la ESPOL.

Encuestados - Jefes de laboratorio de la carrera de ingeniería en Telemática:

- Ing. Christopher Vaccaro (cvaccaro@espol.edu.ec)
- Ing. Steven Santillán (steisant@espol.edu.ec)
- Ing. Sandraa Coello (saiscoel@espol.edu.ec)

Figura 1: Apéndice Primera pregunta - Jefes de laboratorio

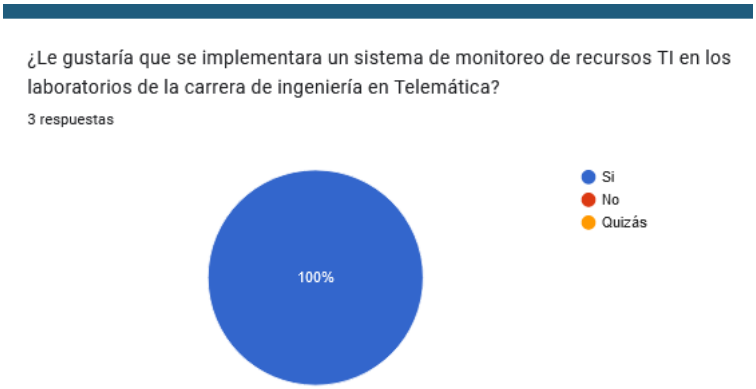

Figura 2: Apéndice Segunda pregunta - Jefes de laboratorio

Figura 3: Apéndice Tercera pregunta - Jefes de laboratorio

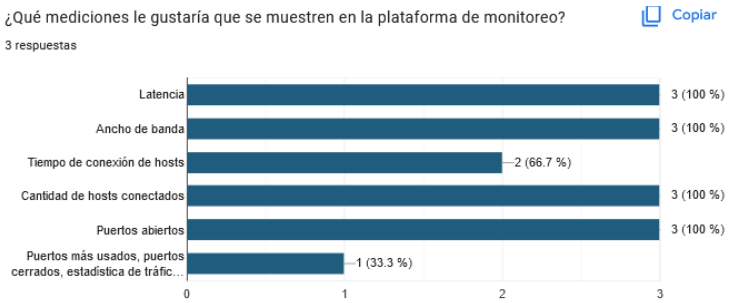

Figura 4: Apéndice Cuarta pregunta - Jefes de laboratorio

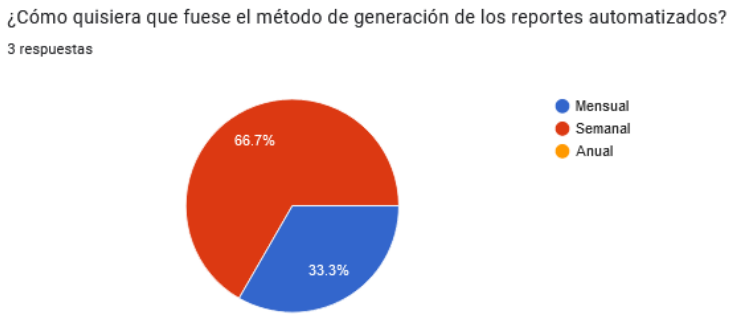

Figura 5: Apéndice Quinta pregunta - Jefes de laboratorio

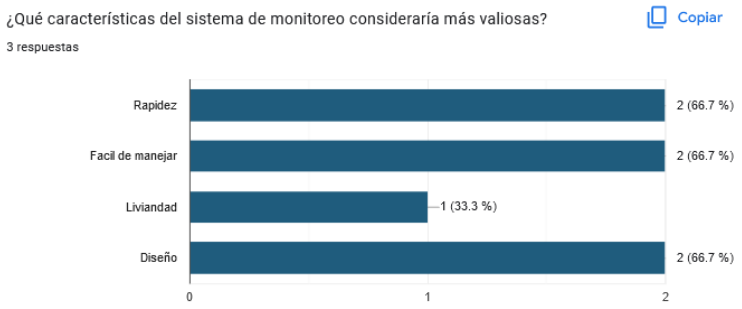

Figura 6: Apéndice Sexta pregunta - Jefes de laboratorio

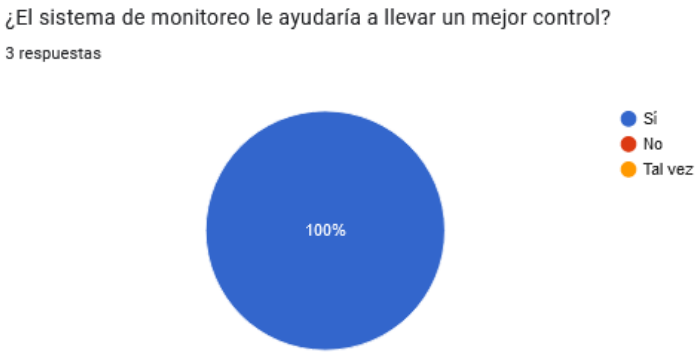

Figura 7: Apéndice Séptima pregunta - Jefes de laboratorio

A.2 SISTEMA DE MONITOREO DE RECURSOS TI - ESTUDIANTES

Formulario de carácter investigativo para recopilar información real con respecto a las métricas de usuario del proyecto integrador de la carrera de ingeniería en Telemática de la ESPOL.

Encuestados - Estudiantes de la carrera de ingeniería en Telemática:

Total: 26 alumnos

- ljblacio@espol.edu.ec

- maytehuratdo@gmail.com
- cinthiabustamantesarango@gmail.com
- juacmend@espol.edu.ec
- fradaini@espol.edu.ec
- angdesan@espol.edu.ec
- davargas@espol.edu.ec
- sofglova@espol.edu.ec
- gusgcast@fiec.espol.edu.ec
- andrescevallosc1998@gmail.com
- lljvargas@espol.edu.ec
- vquispe@idata.ec
- bsponce@espol.edu.ec
- m.beatriz_2016@outlook.es
- marciv2015@hotmail.com
- aguirrepinedajoselynkatherine@gmail.com
- darwinelizalde1982@hotmail.com
- chicadiana13@gmail.com
- gabriela_sanchezmc@hotmail.com
- jhon15quince@gmail.com
- may-arnijos@hotmail.com
- jjzapata@espol.edu.ec
- ceciliahonores51@gmail.com
- darwing1500@hotmail.com

- ferortega58@hotmail.com
- anyele_89@hotmail.com

A.2.1 CONOCIMIENTO DE ALGUNA HERRAMIENTA DE MONITOREO IMPLEMENTADA EN LOS LABORATORIOS DE TELEMÁTICA

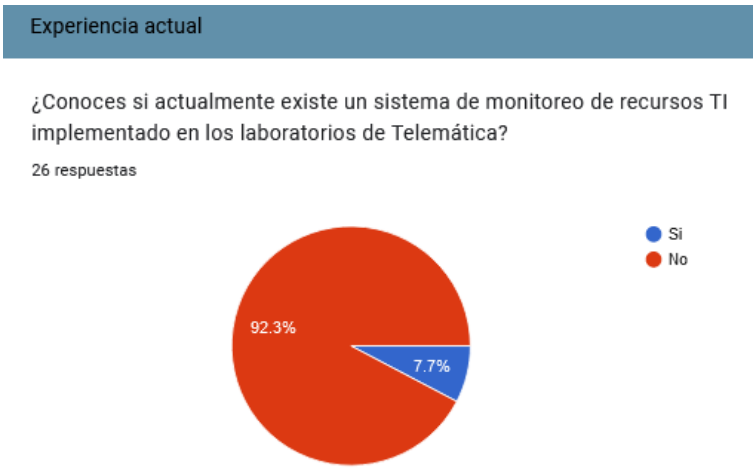

Figura 8: Apéndice Primera pregunta - Estudiantes

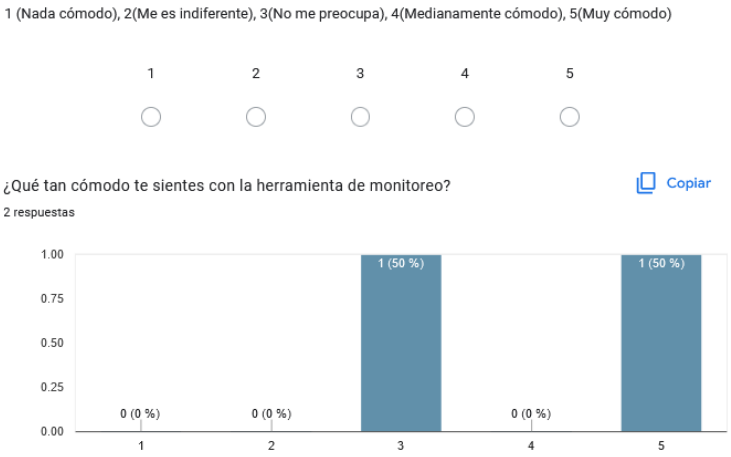

Figura 9: Apéndice Segunda pregunta - Estudiantes

Figura 10: Apéndice Tercera pregunta - Estudiantes

A.2.2 DESCONOCIMIENTO DE ALGUNA HERRAMIENTA DE MONITOREO IMPLEMENTADA EN LOS LABORATORIOS DE TELEMÁTICA

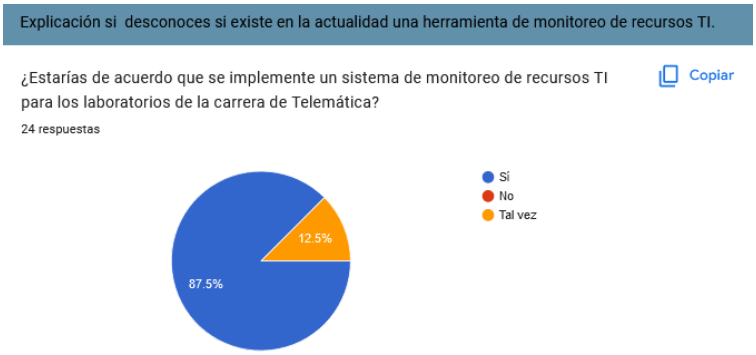

Figura 11: Apéndice Cuarta pregunta - Estudiantes

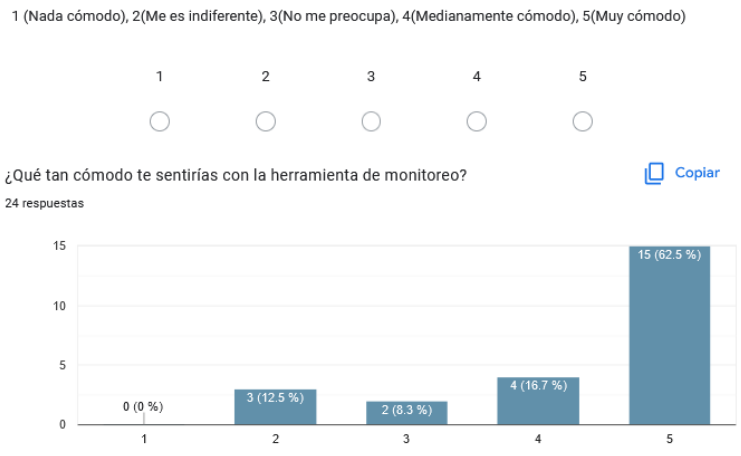

Figura 12: Apéndice Quinta pregunta - Estudiantes

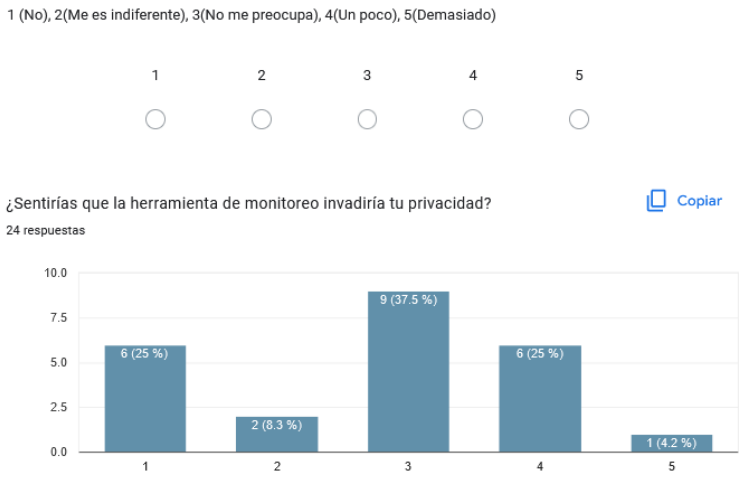

Figura 13: Apéndice Sexta pregunta - Estudiantes

B CÓDIGO EN LA RASPBERRY PARA EJECUTAR LAS RESPECTIVAS FUNCIONES DETALLADAS EN EL CAP3

Código 1: Script de bash para ejecutar las funciones detalladas en el Cap 3 para subir y consultar datos a la base de firebase

```
// Conexion.py
while True:
    if os.stat('/home/kali/Desktop/hosts3.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/totalhost.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/puertosabiertos.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/tx.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/txmedida.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rx.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rxmedida.txt').st_size == 0:
```

```
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avg.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avgmedida.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/txdia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/txmedidadia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rxdia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rxmedidadia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avgdia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avgmedidadia.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/txhora.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/txmedidahora.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rxhora.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/rxmedidahora.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avghora.txt').st_size == 0:
        print('El archivo esta vacio')
    if os.stat('/home/kali/Desktop/avgmedidahora.txt').st_size == 0:
        print('El archivo esta vacio')
    else:
        print('El archivo no esta vacio')
        guardarIpFecha()
        guardarFirebasehost()
        guardarFirebasetotalhost()
```

```
        guardarabmes()
        guardarabdia()
        guardarabhora()
        ipseleccionada=consultaripSeleccionada()
         file =open('ipSeleccionada.txt',  'w')
         file .write(ipseleccionada)
         file .close()

        mesSeleccionado=consultarmes()

        diaSeleccionado=consultardia()

        horaSeleccionada=consultarhora()

        with open('archivo2.txt', mode='w') as file_object:
            print(mesSeleccionado, file=file_object)



        with open('archivo3.txt', mode='w') as file_object:
            print(diaSeleccionado, file=file_object)

        with open('archivo4.txt', mode='w') as file_object:
            print(horaSeleccionada, file=file_object)
        guardartotalpuerto()
        guardarpuertohost()

    time.sleep(8)
```

C CÓDIGO EN ANDROID STUDIO

C.1 CONFIGURACIONES

C.1.1 ANDROID MANIFEST

Código 2: Configuraciones para el correcto funcionamiento de la app

```
// Android Manifest
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http :// schemas.android.com/tools"
    package="com.amst.pruebatesis">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.PruebaTesis"
        tools:targetApi="31">
        <activity
            android:name=".anchoDeBanda"
            android:exported="false" />
        <activity
            android:name=".puertos"
            android:exported="false" />
        <activity
            android:name=".hostConectados"
            android:exported="false" />
```

```
        < activity
            android:name=".tiempodeconexion"
            android:exported="false" />
        < activity
            android:name=".MainActivity"
            android:exported="true">
            <intent- filter >
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent- filter >
        </ activity >
    </application>

</manifest>
```

C.1.2 BUILDGRADLE PROJECT

Código 3: Configuraciones para el correcto funcionamiento de la app

```
// BuildGradle Project
buildscript {

    repositories {
        google()  // Google's Maven repository
        mavenCentral() // Maven Central repository
        maven {
            url 'https :// jitpack .io'
        }

    }

    dependencies {
```

```
        classpath 'com.google.gms:google-services:4.3.13'

    }

}

plugins {
    id 'com.android.application' version '7.2.1' apply false
    id 'com.android.library' version '7.2.1' apply false
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

C.1.3 BUILDGRADLE APP

Código 4: Configuraciones para el correcto funcionamiento de la app

```
// BuildGradle App
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
}

android {
    compileSdk 32

    defaultConfig {
        applicationId "com.amst.pruebatesis"
        minSdk 21
        targetSdk 32
        versionCode 1
        versionName "1.0"
```

```
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation platform('com.google.firebase:firebase-bom:31.1.1')
    implementation 'com.google.firebase:firebase-analytics'
    implementation "com.androidplot:androidplot-core:1.5.10"
    implementation 'androidx.appcompat:appcompat:1.5.1'
    implementation 'com.google.android.material:material:1.7.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    implementation 'com.firebaseui:firebase-ui-firestore:8.0.2'
    implementation 'com.google.firebase:firebase-firestore:24.4.1'
    implementation 'com.google.firebase:firebase-database:20.1.0'
    implementation "androidx.recyclerview:recyclerview:1.2.1"
    // For control over item selection of both touch and mouse driven selection
    implementation "androidx.recyclerview:recyclerview-selection:1.1.0"
    implementation 'com.github.PhilJay:MPAndroidChart:v3.1.0-alpha'
```

```
}
```

C.1.4 SETTINGS GRADLE

Código 5: Configuraciones para el correcto funcionamiento de la app

```
// Settings Gradle
pluginManagement {
    repositories {
        gradlePluginPortal()
        google()
        mavenCentral()

    }
}
dependencyResolutionManagement {
    // repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
        maven { url "https :// jitpack . io" }
    }
}
rootProject.name = "pruebaTesis"
include ':app'
```

C.2 JAVA CLASS

C.2.1 VENTANA PRINCIPAL

Código 6: Clase de java que permite pasar a la siguiente ventana de acuerdo a la selección del laboratorio que el usuario realice

```
// MainActivity
public class MainActivity extends AppCompatActivity {
```

```
    private Button button2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button2=(Button) findViewById(R.id.button2);

        button2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent3 = new Intent(MainActivity.this, hostConectados.class);
                startActivity(intent3);
            }
        });

    }
}
```

C.2.2 HOST CONECTADOS

Código 7: Clase de java que permite consultar ip de host, se utiliza en ListHostAdapter para llenar el listView

```
// host
public class host {

    private String host;

    public host(String host) {
        this.host = host;
    }
```

```
    public   host(){}

    public  String  getHost() {
        return  host;
    }

    public  void  setHost(String  host)  {
        this .host = host;
    }
}
```

Código 8: Clase de java que permite llenar el listView

```
//  ListHostAdapter
public  class  ListaHostAdapter extends
RecyclerView.Adapter<ListaHostAdapter.ViewHolder> implements View.OnClickListener {

    private   int   resource;
    private   int   resource2;
    private  ArrayList<host>  hostlista ;
    private  ArrayList<puerto>  puertolista ;
    // private   static  Context context;
    private  View.OnClickListener listener ;


    public  ListaHostAdapter(ArrayList<host> hostlista ,  int  resource) {
        this . hostlista  =  hostlista ;
        this .resource=resource;
    }

    @NonNull
    @Override
    public  ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
        View view = LayoutInflater .from(viewGroup.getContext()).inflate(resource,
        viewGroup, false);
```

```
        view.setOnClickListener(this);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder viewHolder, int i) {
        host host = hostlista.get(i);
        viewHolder.viewHost.setText(host.getHost());
        // host item = (host) hostlista.get(i);
        // String nombre= item.getHost();
        // Intent intent = new Intent(context, hostConectados.class);
        // intent.putExtra("nombre", nombre);

    }

    @Override
    public int getItemCount() {
        return hostlista.size();
    }

    public void setOnClickListener(View.OnClickListener listener){
        this.listener=listener;
    }

    @Override
    public void onClick(View view) {
        if (listener!=null){
            listener.onClick(view);

        }
    }

    public class ViewHolder extends RecyclerView.ViewHolder{
        private TextView viewHost;
```

```
        public View view;

        public ViewHolder(View view){
            super(view);
            this.view=view;
            this.viewHost= (TextView) view.findViewById(R.id.viewHost);

        }

    }


}
```

C.3 XML

C.3.1 VENTANA PRINCIPAL

Código 9: XML que muestra dos opciones, las cuales son laboratorio de redes de datos y laboratorio de sistemas telemáticos

```
// activity_main.xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
```

```
android: orientation =" vertical ">


<ImageView
    android:id="@+id/imageView"
    android:layout_width="358dp"
    android:layout_height="wrap_content"
    android: layout_gravity="center"
    android:foregroundGravity="center"
    android: gravity =" left "
    app:srcCompat="@drawable/espol" />

<TextView
    android:layout_width="match_parent"
    android:layout_height="53dp"
    android: layout_gravity="center"
    android:foregroundGravity="center"
    android: gravity ="center"
    android: text="Laboratorios"
    android:textColor="#000000"
    android:textSize="25sp" />

<ImageView
    android: id="@+id/imageView2"
    android:layout_width="match_parent"
    android:layout_height="188dp"
    android: layout_gravity="center"
    android:foregroundGravity="center"
    android: gravity =" left "
    app:srcCompat="@drawable/redesdedatos" />

<Button
    android: id="@+id/button2"
    android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:foregroundGravity="center"
            android:text="LRD" />

        <TextView
            android:id="@+id/textView16"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <ImageView
            android:id="@+id/imageView3"
            android:layout_width="match_parent"
            android:layout_height="175dp"
            app:srcCompat="@drawable/abet" />

        <Button
            android:id="@+id/button3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:foregroundGravity="center"
            android:text="LST" />

        <ImageView
            android:id="@+id/imageView4"
            android:layout_width="451dp"
            android:layout_height="120dp"
            app:srcCompat="@drawable/espol2" />
    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

C.3.2 HOST CONECTADOS

Código 10: XML para mostrarse en la listView

```
// lista_item_host.xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation=" vertical ">

        <TextView
            android:id="@+id/viewHost"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textColor="#000000"
            android:textSize="16sp" />
    </LinearLayout>
</RelativeLayout>
```