UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN TELECOMUNICACIONES

Tema: Deep Packet Inspection en el router Mikrotik de borde de la red de un ISP con el fin de identificar y prevenir ataques externos.

Trabajo de Titulación, previo a la obtención del Grado Académico de

Magíster en Telecomunicaciones

Modalidad de Titulación "Proyecto de Desarrollo"

Autor: Ingeniero Santiago Martín López Narváez

Director: Ingeniero David Omar Guevara Aulestia, Magíster

Ambato – Ecuador

2021

APROBACIÓN DEL TRABAJO DE TITULACIÓN

A la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas Electrónica e Industrial.

El Tribunal receptor de la defensa del Trabajo de Titulación presidido por la Ingeniera Elsa Pilar Urrutia Urrutia Magíster, e integrado por los señores Ingeniero Félix Óscar Fernández Peña Doctor e Ingeniero Rubén Eduardo Nogales Portero Magíster, designados por la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, para receptar el Trabajo de Titulación con el tema: "Deep Packet Inspection en el router Mikrotik de borde de la red de un ISP con el fin de identificar y prevenir ataques externos", elaborado y presentado por el señor Ingeniero Santiago Martín López Narváez, para optar por el Grado Académico de Magíster en Telecomunicaciones; una vez escuchada la defensa oral del Trabajo de Titulación el Tribunal aprueba y remite el trabajo para uso y custodia en las bibliotecas de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Elsa Pilar Urrutia Urrutia Mg.

Presidente y Miembro del Tribunal de Defensa

Firmado electrónicamente por:
FELIX OSCAR FERNANDEZ PENA

Ing. Félix Óscar Fernández Peña, Dr.

Miembro del Tribunal de Defensa

Firmado electrónicamente por:
RUBEN EDUARDO NOGALES PORTERO

Ing. Rubén Eduardo Nogales Portero, Mg.

Miembro del Tribunal de Defensa

AUTORÍA DEL TRABAJO DE TITULACIÓN

La responsabilidad de las opiniones, comentarios y críticas emitidas en el Trabajo de Titulación presentado con el tema: “Deep Packet Inspection en el router Mikrotik de borde de la red de un ISP con el fin de identificar y prevenir ataques externos”, le corresponde exclusivamente a: Ingeniero Santiago Martín López Narváez, Autor bajo la Dirección del Ingeniero David Omar Guevara Aulestia, Magíster, Director del Trabajo de Titulación; y el patrimonio intelectual a la Universidad Técnica de Ambato.

Firmado electrónicamente por:
SANTIAGO MARTIN LOPEZ NARVAEZ

Ing. Santiago Martín López Narváez

AUTOR

Firmado electrónicamente por:
DAVID OMAR GUEVARA AULESTIA

Ing. David Omar Guevara Aulestia, Mg.

DIRECTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que el Trabajo de Titulación, sirva como un documento disponible para su lectura, consulta y procesos de investigación, según las normas de la Institución.

Cedo los Derechos de mi Trabajo de Titulación, con fines de difusión pública, además apruebo la reproducción de este, dentro de las regulaciones de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
SANTIAGO MARTIN LOPEZ NARVAEZ

Ing. Santiago Martín López Narváez

c.c. 1804037982

INDICE GENERAL

Contenido

INDICE DE TABLAS

INDICE DE FIGURAS

GLOSARIO DE TÉRMINOS

API	Application Programming Interface
BGP	Border Gateway Protocol
DoS	Ataque de Denegación de Servicio (Denial-of-Service attack)
DDoS	Ataque de Denegación de Servicio Distribuido (Distributed Denial-of-Service attack)
DPI	Inspección Profunda de Paquetes (Deep Packet Inspection)
ELK	Elasticsearch Logstach Kibana
ISP	Proveedor de Servicios de Internet (Internet Service Provider)
KVM	Kernel-based Virtual Machine
NAS	Almacenamiento Conectado a la Red (Network Attached Storage)
NOC	Centro de Operaciones de Red (Network Operations Center)
VM	Virtual Machine

AGRADECIMIENTO

Me gustaría expresar un especial agradecimiento a Laura Medina y Segundo Guerrero por el respaldo que me extendieron cuando les comuniqué mi deseo de seguir esta maestría.

DEDICATORIA

A Dios, mi esposa Rebeca y mis hijos Elí y Leo

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN TELECOMUNICACIONES

TEMA:

DEEP PACKET INSPECTION EN EL ROUTER MIKROTIK DE BORDE DE LA RED DE UN ISP CON EL FIN DE IDENTIFICAR Y PREVENIR ATAQUES EXTERNOS.

AUTOR: Ing. Santiago Martín López Narváez

DIRECTOR: Ing. David Omar Guevara Aulestia, Mg

LÍNEA DE INVESTIGACIÓN: Tecnologías, Seguridad y Gestión de Redes de Comunicaciones

FECHA: 30 de junio del 2021

RESUMEN EJECUTIVO

La protección de equipos de infraestructura de red y su disponibilidad se ven afectados por el aumento de ataques cibernéticos. Por esto, existe la necesidad de que se desarrollen métodos que permitan reducir el impacto negativo que provocan los ataques. Es indispensable que cualquier entidad empresarial, sea esta, pequeña, mediana o grande, cuente con su debida seguridad de red para protegerse. De igual manera, un proveedor de servicios de internet (ISP por sus siglas en inglés) debe contar con todas las seguridades, de modo que el usuario final reciba ininterrumpidamente el servicio y que la información que intercambia en internet este resguardada. En este proyecto de titulación, se presenta una herramienta que permite detectar ataques DDoS con el uso de DPI y NetFlow, para que pueda ser empleado en la red de borde de un proveedor de servicios de internet. El algoritmo propuesto, para la detección de ataques, está elaborado en Python y se lo probó en la red del ISP durante ocho días, obteniendo una cantidad de aciertos en la detección de ataques del 92%. Para el colector NetFlow se usa Elasticsearch y Kibana que permiten presentar la información recolectada de forma ordenada y elaborar gráficas de interés para el ISP. Cabe recalcar que el script elaborado está desarrollado para que una vez detectado el ataque, se comunique con el router de borde de marca Mikrotik que usa el ISP y se bloquee la IP de origen del ataque. Gracias a esta herramienta, el ISP puede contar con una capa de seguridad y ofrecer mayor disponibilidad a sus clientes.

Descriptores: Seguridad, Redes, DDoS, DPI, NetFlow

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN TELECOMUNICACIONES

THEME:

DEEP PACKET INSPECTION IN AN ISP'S MIKROTIK NETWORK BORDER ROUTER FOR THE PURPOSE OF IDENTIFYING AND PREVENTING EXTERNAL ATTACKS.

AUTHOR: Ing. Santiago Martín López Narváez

DIRECTED BY: Ing. David Omar Guevara Aulestia, Mg

LINE OF RESEARCH: Technologies, Security and Communications Networks Management

DATE: June 30th, 2021

EXECUTIVE SUMMARY

The network infrastructure equipment's protection and its availability are affected by the increase of cyberattacks. For this reason, there is a need for developing methods to reduce the negative impact caused by attacks. It is essential that any small, medium, or large business, has its proper network security to protect itself. Similarly, an Internet Service Provider (ISP) must have all the security. Therefore, their customers have the service without interruption, and the information they exchanged on the internet is also properly protected. This research project suggests a tool to detect DDoS attacks with the use of DPI and NetFlow. Thus, the network's border of the internet service provider uses the tool mentioned before. The algorithm uses Python code for detecting attacks, and it tested on the ISP's network for eight days, obtaining true positives hits in the detection of cyberattacks of 92%. The NetFlow collector uses Elasticsearch and Kibana, allowing the information collected to be presented in an orderly manner and to create graphs of interest to the ISP. It is crucial to highlight that the developed script communicates with the ISP's Mikrotik border router to block the attack source IP. Thanks to this tool, the ISP can have a layer of security and offer greater availability to its customers.

Keywords: Network security, DDoS, DPI, NetFlow

1. CAPÍTULO I

1.1 Introducción

Con el incremento de ciber ataques, investigadores han puesto su mayor esfuerzo para proveer seguridad en Internet. Es un reto importante proteger tanto la información, así como, asegurar la disponibilidad de una red, ya sea esta empresarial, gubernamental o un ISP. En estos últimos años se han dado diferentes tipos de ataques vulnerando seguridades y exponiendo valiosa información. A pesar del aumento de herramientas de protección, estos ataques se incrementan y cada vez son más elaborados.

Las herramientas que se enfocan en seguridad muchas veces se ven limitadas por el alto consumo de recursos de hardware. Esta limitante se ve reflejada en la capacidad de paquetes que pueden ser analizados en tiempo real. Es por esto que existe la necesidad que se desarrollen métodos de detección de ataques que permitan optimizar recursos y sean precisos.

El uso de Deep Packet Inspection (DPI) en detección de ataques, permite que se pueda utilizar diferentes métodos de detección e ir mejorando el tiempo de respuesta y la precisión en la inspección de un paquete.

1.2 Justificación

La seguridad en redes de datos es algo fundamental y que no se debe pasar por alto. Poder prevenir y detectar ataques a tiempo en una red, es una tarea que constantemente se encuentra en desarrollo. Denegación de Servicio Distribuido (DDoS por sus siglas en inglés) es una de los ataques cibernéticos que se presentan constantemente por lo que se ha puesto mucho esfuerzo en poder contrarrestarlo. En el 2020 según el reporte presentado por NETSCOUT, se detectaron más de 10 millones de ataques con un promedio de 150 mil ataques por mes [1]. También, en el Reporte Anual de Internet (2018-2023) de Cisco, presentado en Marzo del 2020, se pronosticó que para ese año los ataques DDoS estarían cerca de 10,8 millones y para el 2023 estarán sobre los 15 millones, según se muestra en la Figura 1.

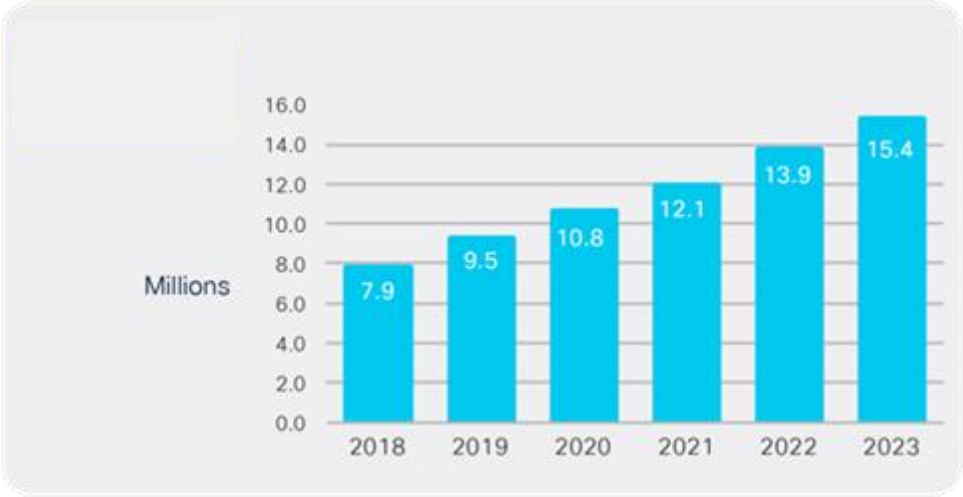


Figura 1. Pronóstico del número de ataques DDoS hasta el 2023
Fuente: Cisco Annual Internet Report, 2018–2023 [2]

Es ahí la importancia de tener herramientas que estén a la par de contrarrestar estos ataques. En el mercado se pueden encontrar herramientas que ofrecen mitigar ataques DDoS. El costo para adquirir estas herramientas puede ser alto dependiendo de las características y la cantidad de tráfico soporte. Con el uso de NetFlow para recolectar los datos y Deep Packet Inspection se puede tener una herramienta que permite inspeccionar profundamente un paquete de datos y determinar si es parte de un ataque [3].

1.3 Objetivos

1.3.1. Objetivo General

Implementar una aplicación que bloquee paquetes maliciosos que entran al Router Mikrotik.

1.3.2. Objetivos Específicos

- Analizar el protocolo NetFlow que actualmente soporta Mikrotik RouterOS
- Determinar los ataques externos DoS y cómo detenerlos.
- Implementar una aplicación que bloquee los paquetes maliciosos.
- Validar los resultados obtenidos en la detección de ataques.

2. CAPÍTULO II

2.1 Estado del Arte

Según El-Maghraby R.T. et al [4], DPI es parte esencial para reconocer aplicaciones de redes. También señala que es ampliamente utilizado para seguridad en redes. Un ejemplo de aplicación de DPI se encuentra en [5], donde se analiza la importancia de la seguridad en dispositivos IoT (Internet of Things) debido al incremento de ataques DDoS a que se ven expuestos. Por la utilidad que se le reconoce, en el presente trabajo de titulación, se utiliza DPI para el análisis de puertos e IPs que intervienen en el intercambio de datos.

Uno de los métodos más utilizados para encontrar ciertas características en los paquetes analizados es estudiado en [6], donde se describe cómo se realizan comparaciones automáticas con expresiones regulares. Además, hacen un análisis del hardware para acelerar el proceso de comparación. Concluyen exponiendo que la complejidad y la extensión de una expresión regular, incrementa exponencialmente el requerimiento de memoria, el cual puede llegar a sobrepasar la capacidad y velocidad actualmente disponible; todo esto por la demanda de enlaces de alta velocidad y capacidad, que sobrepasan a la velocidad de las memorias. Es por eso, que una de las partes para la implementación de la aplicación a tomar en cuenta para la detección de ataques DDoS, es el hardware utilizado.

Ataques DDoS se determinaron con DPI en [7] utilizando una librería para Linux llamada libcap. Esta librería permite enviar y recibir paquetes a través de una red fácilmente. Luego de capturar los paquetes, se almacenaban en una base datos para posteriormente analizarlos. En el artículo que también se ha desarrollado una herramienta de detección de ataques DDoS fue en [8]. En este caso, se utilizó datos enviados a través de NetFlow y analizados empleando machine learning. Concluyen que la herramienta desarrollada promedia es del 99% y los falsos positivos son menor al 0,5% del total de los datos analizados. Los dos artículos mencionados anteriormente, utilizan métodos diferentes de recolección de datos. Para fines de este proyecto se utiliza NetFlow, que entre sus ventajas es la posibilidad de utilizar una topología de red en la que el colector de información no tiene que estar en medio de la red; de hecho, el colector puede estar en la nube.

Para la detección de anomalías en una red y adquisición de datos en [9] se ha utilizado procesos de Big Data en conjunto con NetFlow. En éste artículo se encontró que uno de los problemas para la detección de ataques es la gran cantidad de información que se transporta a través de las redes, dificultando el proceso de ordenarla e identificarla. En esta investigación, se toma en cuenta la recomendación que presenta con respecto a la cantidad de información, limitando la recolección de información para las horas pico de una semana aleatoria.

Según Tian [10], después de realizar tres experimentos, concluye que el rendimiento de Elasticsearch para almacenar datos tomados con NetFlow es mucho mejor al que hay utilizando una base de datos SQL; esto debido a la gran cantidad de información que se puede almacenar y la flexibilidad para ordenarla. Es importante este estudio que muestra una experiencia positiva con Elasticsearch y, por eso se adopta para su aplicación en este proyecto de titulación.

Para Wang et al. [11], ELK (Elasticsearch, Logstach y Kibana) no solo permite guardar la información de una forma adecuada, sino que permite mostrar de una forma gráfica los datos para su análisis. Algunas gráficas de interés para el ISP se pudieron obtener mediante Kibana.

2.2 Marco Teórico

2.2.1. Denegación de Servicio DoS

Un ataque de Denegación de Servicio (DoS por sus siglas en inglés) busca interrumpir el funcionamiento normal de un servicio o una red, impidiendo el acceso al usuario. El ataque que proviene de una sola fuente inunda de tráfico la red o el servidor haciéndole inaccesible.

Un ataque de Denegación de Servicio Distribuida (DDoS por sus siglas en inglés) es un tipo de ataque DoS que igualmente busca interrumpir el funcionamiento normal de un servicio, servidor o red, inundando con una gran cantidad de tráfico el objetivo del ataque o su red circundante. Para efectuar un ataque DDoS se utilizan múltiples dispositivos informáticos infectados, los cuales se convierten en fuente

de tráfico de ataque. Se puede considerar a un ataque DDoS como un atasco en el tránsito que evita que la información llegue al destino deseado.

Para que se ejecute un ataque DDoS, el atacante tiene que tomar control de una red de máquinas en línea. Cada computador o maquina (tales como dispositivos IoT) son infectados con malware, convirtiéndose en bots (zombis). El atacante tiene control remoto sobre todo el grupo de bots el cual es llamado botnet.

Existe una gran variedad de ataques DDoS. Se los puede dividir principalmente en dos grupos: inundación de ancho de banda y agotamiento de recursos [12].

Inundación de ancho de banda

Es un tipo de ataque DDoS que busca agotar todo el ancho de banda que dispone una organización, principalmente el objetivo son los enlaces externos o equipos de borde. Los ataques de este tipo más conocidos son inundación ICMP y UDP

Agotamiento de recursos

Este tipo de ataque explota ciertas características o algún protocolo en la máquina objetivo. El ejemplo más conocido es la inundación de conexión (a.k.a SYN flood), que utiliza parte del handshake de tres vías en el protocolo TCP. Busca agotar los recursos de la máquina atacada para que sea inalcanzable. En el ataque “SYN flood”, el atacante envía peticiones de conexión TCP tan rápidas que la máquina objetivo no puede procesarlas, causando sobrecarga en la red.

2.2.2. Deep Packet Inspection

Deep Packet Inspection (por sus siglas en inglés DPI) se refiere a la inspección profunda de los paquetes que fluyen a través de una red y, en base a los resultados de la inspección, se toman decisiones. Básicamente para DPI se aplican dos procesos que son Identificación y Acción. La identificación consiste en encontrar una característica especial en el paquete; esta puede ser, protocolos, virus, gusanos, formato, etc. Una vez identificado el paquete malicioso entra el proceso de acción. La Acción consiste en decidir como el paquete será tratado de acuerdo al resultado de la identificación [6].

El primer paso para el funcionamiento de DPI es recibir el paquete, para ello existen algunos métodos tales como: port mirroring, cables divisores, middlebox [13], etc.

2.2.3. Aplicaciones DPI

Según el artículo [6], las posibles aplicaciones de DPI son:

- Seguridad en redes
- Administración de Ancho de Banda
- Perfiles de Usuario/Inyección de Publicidad
- Derechos de Autor
- Vigilancia Gubernamental y Censura

2.2.4. Métodos para la implementación de DPI

Como previamente se ha dicho, DPI puede ser utilizado para inspeccionar la cabecera de un paquete y así identificar aplicaciones. También puede ser utilizado para inspeccionar los datos que el paquete lleva e identificar el contenido. A continuación se enumera los métodos presentados en [14]:

- Basado en los puertos
- Combinación de Patrones
- Análisis Estocástico
- Decodificación del protocolo

2.2.5. Protocolo NetFlow

NetFlow es un protocolo de red propuesto por Cisco Systems para monitorear el tráfico de red. NetFlow está compuesto por un conjunto de atributos y puede ser capturado o enviado por equipos de red tales como routers y switches [8]. La Figura 2 muestra la arquitectura de red en la que NetFlow puede ser utilizada.

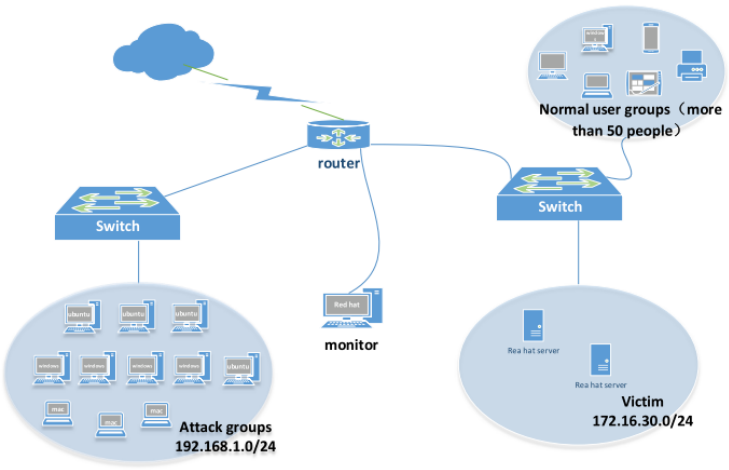


Figura 2. Arquitectura de red para NetFlow
Fuente: Tomado de [8]

Los sistemas que utilizan NetFlow para inspección de paquetes, requieren más recursos de hardware como CPU, memoria y ancho de banda por la cantidad de conexiones simultaneas.

Al analizar la información obtenida con NetFlow, se puede identificar problemas que existen como congestión de una red, conocer cuál es la fuente y destino, determinar la clase de servicio (CoS) de las aplicaciones y usuarios, etc. También, NetFlow permite realizar mediciones precisas de tráfico.

Un sistema NetFlow depende principalmente de dos componentes: NetFlow caché que es el encargado de tomar y guardar la información para posteriormente enviarla y, el que recibe los paquetes llamado colector NetFlow. La forma en que opera el NetFlow caché, es básicamente almacenando “flows records” quienes son los que contienen varios campos de información y se envían al colector. Durante todo el proceso el paquete IP no es modificado, además que para la infraestructura de red existente es transparente. NetFlow es capaz de monitorear flujos IPv4, así como flujos IPv6 y puede capturar información tanto de los paquetes que ingresan, así como de los que salen.

Versiones de Netflow[15]

Las versiones de NetFlow son: 1, 5, 7, 8 y 9. Las versiones no mencionadas nunca se lanzaron oficialmente por lo tanto no se utilizan. Para los equipos Mikrotik que

utilizan el sistema operativo RouterOS, existe la herramienta Traffic Flow que es compatible con las versiones 1, 5 y 9 de NetFlow.

La versión 1, conocida como el formato original de exportación, actualmente no se recomienda su uso a menos que sea realmente necesario o en caso de que algún equipo antiguo por compatibilidad lo requiera.

El formato de la versión 5 del protocolo NetFlow agrega, entre otras mejoras, información de los flujos de los Sistemas Autónomos del protocolo BGP. En la Figura 3 se muestran algunos de los elementos que se exporta en esta versión.

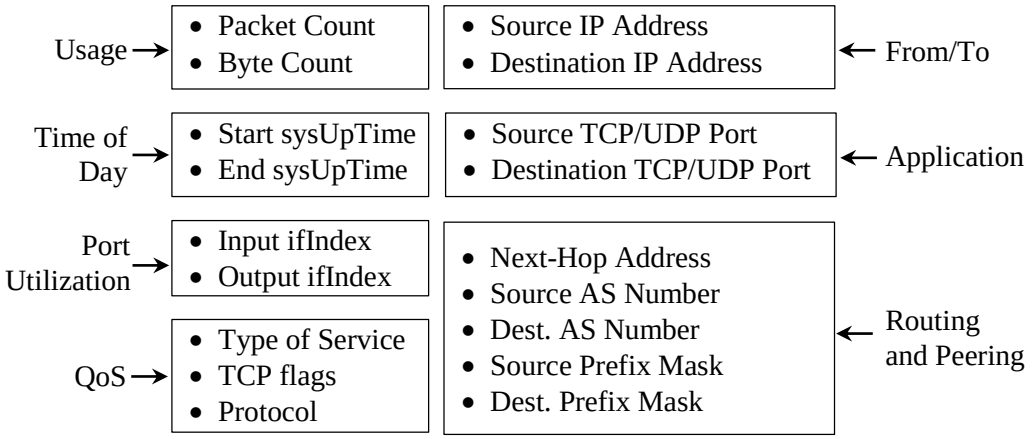


Figura 3. Elementos que se exportan en NetFlow versión 5
Fuente: Tomado de [15]

La versión 9 del protocolo es la más actual. La versatilidad de esta versión radica en que se basa en plantillas, lo que permite que se pueda ir mejorando sin necesidad de cambios en el formato de los flujos. Las plantillas permiten describir los campos que estarán siendo enviados en los próximos conjuntos de flujos. En un paquete de exportación, se puede mezclar varias plantillas y conjunto de flujos, como se muestra en la Figura 4.

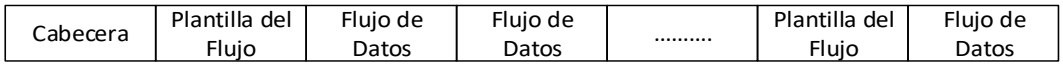


Figura 4. Paquete de exportación de la versión 9 del protocolo NetFlow
Fuente: Tomado de [15]

Esta versión del protocolo introduce nuevas características tales como: envío de flujos a diferentes colectores según los datos; filtros según la data que se requiera; soporte para MPLS y Multicast; entre otras mejoras.

La versión 9 ha sido tomada por el IETF como base para estandarizar IPFIX (IP Flow Information Export). IPFIX es un protocolo generalizado para que diferentes fabricantes puedan desarrollar equipos y software que explote las ventajas de NetFlow.

La versión 9 de NetFlow es la que se utiliza para este proyecto. El router Mikrotik puede trabajar con la versión 5 y 9 pero se prefiere la versión más actual porque se puede tener más características de los paquetes que fluyen a través del router. Según lo señalado en los ataques DDoS, los puertos de los servicios que mayormente son explotados son los de interés para protección de la red del ISP. Estos puertos serán identificados mediante DPI.

3. CAPÍTULO III

3.1 Metodología

3.1.1. Ubicación

Este proyecto de titulación se desarrolló en el NOC principal del ISP Speedycom Cia. Ltda., ubicado en la ciudad de Ambato. El NOC cuenta con todas las facilidades y redundancias para mantener un servicio continuo a sus usuarios. Es el sitio donde llegan los enlaces principales hacia las salidas internacionales.

3.1.2. Equipos y Materiales

Para el desarrollo de la aplicación y análisis de la información se utilizaron los siguientes equipos y software mostrados en la Tabla **1**:

Equipos	Router Mikrotik CCR1072-1G-8S+ Servidor Dell R740 1xCPU Intel Xeon Silver 4116 NAS Synology DS3614xs+
Software	Proxmox Sistema Operativo Debian 9 RouterOS Mikrotik Elasticsearch, Kibana Python

Tabla 1. Equipos y software empleados
Fuente: Elaborado por el investigador

En este proyecto se elaboró una aplicación con Deep Packet Inspection y NetFlow colector que detecta ataques DDoS. Todos los paquetes NetFlow que el router Mikrotik de borde del ISP generaba, han sido recolectados y almacenados con Elasticsearch. Posteriormente, haciendo uso de Kibana se pueden clasificar y presentar de una forma gráfica toda la información obtenida. Una vez filtrada la información, a través del API de Mikrotik y un script de Python, se envía la dirección IP atacante para ser bloqueada. Esta aplicación se puso en el ISP ocho días para analizar la información y observar si está siendo víctima de algún tipo de ataque principalmente de DDoS.

3.2 Tipo de investigación

En este proyecto se aplica el estudio descriptivo con el fin de determinar la capacidad de la aplicación diseñada para detectar ataques a la red del ISP, con un enfoque cuantitativo.

3.3 Población y Muestra

Los datos que se recolectaron para analizarlos son de un ISP que tiene alrededor de 5000 usuarios, alcanzando un tráfico máximo de alrededor de 6 Gbps en horas pico.

3.4 Hipótesis

Con la aplicación basada en DPI y NetFlow, se puede bloquear los ataques DDoS hacia la red del ISP y mantener la continuidad del servicio de internet hacia los usuarios finales.

3.5 Recolección de información

La información que atraviesa el equipo de borde del ISP, es la que se recolecta para posteriormente analizarla, como muestra la siguiente Figura 5.

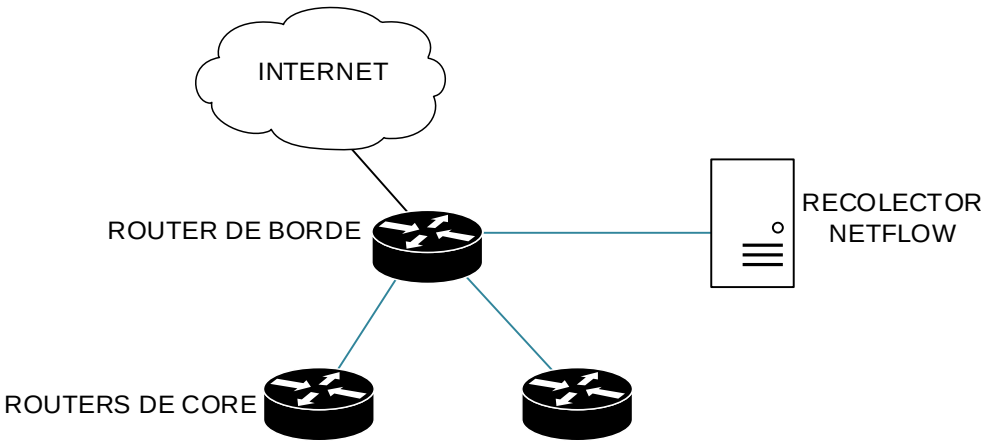


Figura 5. Esquema de red aplicado para la recolección de datos
Fuente: Elaborado por el investigador

La información fue recolectada durante las horas pico, en las que mayor tráfico está cursando por el equipo de borde del ISP. Según el trafico histórico tomado del monitoreo del ISP mostrado en la Figura 6, la hora pico aproximadamente es desde las 18H00 hasta las 23H00. Los días en los que se alcanza un mayor tráfico de red son domingo y martes.

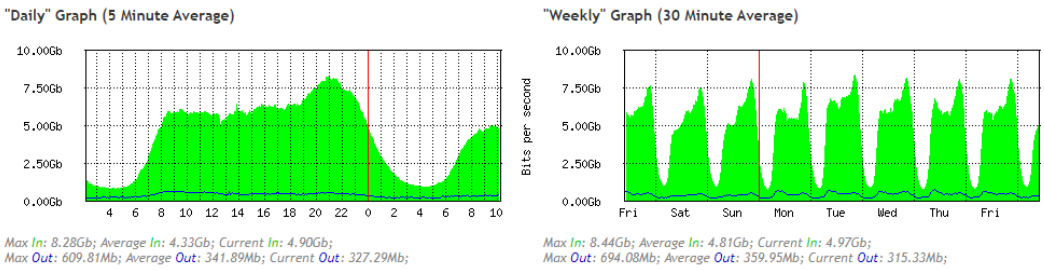


Figura 6. Grafica diaria y semanal del tráfico del equipo de borde del ISP
Fuente: Tomado de los datos históricos del router de borde del ISP

La información que se analiza en este proyecto de tesis y por límites en el almacenamiento, se colectó desde las 18H00 hasta las 23H00, durante 8 días seguidos de una semana aleatoria. Hay que tomar en cuenta que el tráfico de cada día es variable. Incluso, puede haber una mayor variabilidad en el tráfico si se trata de un día festivo. No obstante, los datos en las condiciones en que se recopilaron son característicos de un tráfico típico. Esto da validez a los resultados experimentales obtenidos.

Una vez recolectada la información se analizó aplicando el algoritmo que se muestra en el diagrama de flujo de la Figura 8. Como se ha indicado en el marco teórico, los diferentes ataques DDoS se enfocan en atacar a un mismo objetivo desde diferentes bots, por lo que el algoritmo se basa en detectar las IPs de origen que están atacando las IPs del ISP. De los flows obtenidos se filtra los puertos de destino que mayormente son atacados en UDP y TCP respectivamente (21, 22, 23, 53, 80, 161, 443, 5060, 8080), o a su vez los puertos que se quiere proteger según el servicio. De todos esos flows, se filtra las IPs de origen que mayor cantidad de paquetes tienen y se las compara con la “Whitelist”. Las IPs que no se encontraron en la “Whitelist” se las continúa analizando. Se busca que estas IPs coincidan con las de otros puertos de destino atacados, para proceder con el envío al firewall del router de borde y que sean bloqueadas. El código ejecutado para encontrar las IPs atacantes se muestra en la Figura 7.

En la lista blanca o “Whitelist” encontramos las IPs de los servicios comunes como son: los caches del ISP, caches del proveedor internacional, servicios de Google, servicios de Facebook, servicios de Zoom y servicios de Microsoft.

En el Anexo 2 se muestra todo el código utilizado para bloquear las IPs atacantes.

```
filter ={

  "aggs": {
    "4": {
      "terms": {
        "field": "source.port",
        "order": {
          "2": "desc"
        },
        "size": 500
      },
      "aggs": {
        "2": {
          "sum": {
            "field": "network.bytes"
          }
        },
        "3": {
          "sum": {
            "field": "network.packets"
          }
        }
      }
    }
  },
  "size": 0,
  "stored_fields": [
    "*"
  ],
  "script_fields": {},
  "docvalue_fields": [
    {
      "field": "@timestamp",
      "format": "date_time"
    },
    {
      "field": "netflow.collection_time_milliseconds",
      "format": "date_time"

    },
    {
      "field": "netflow.collection_time_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.exporter.timestamp",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_microseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_nanoseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_start_microseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_start_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_start_nanoseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_start_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.max_export_seconds",
      "format": "date_time"
    },
```

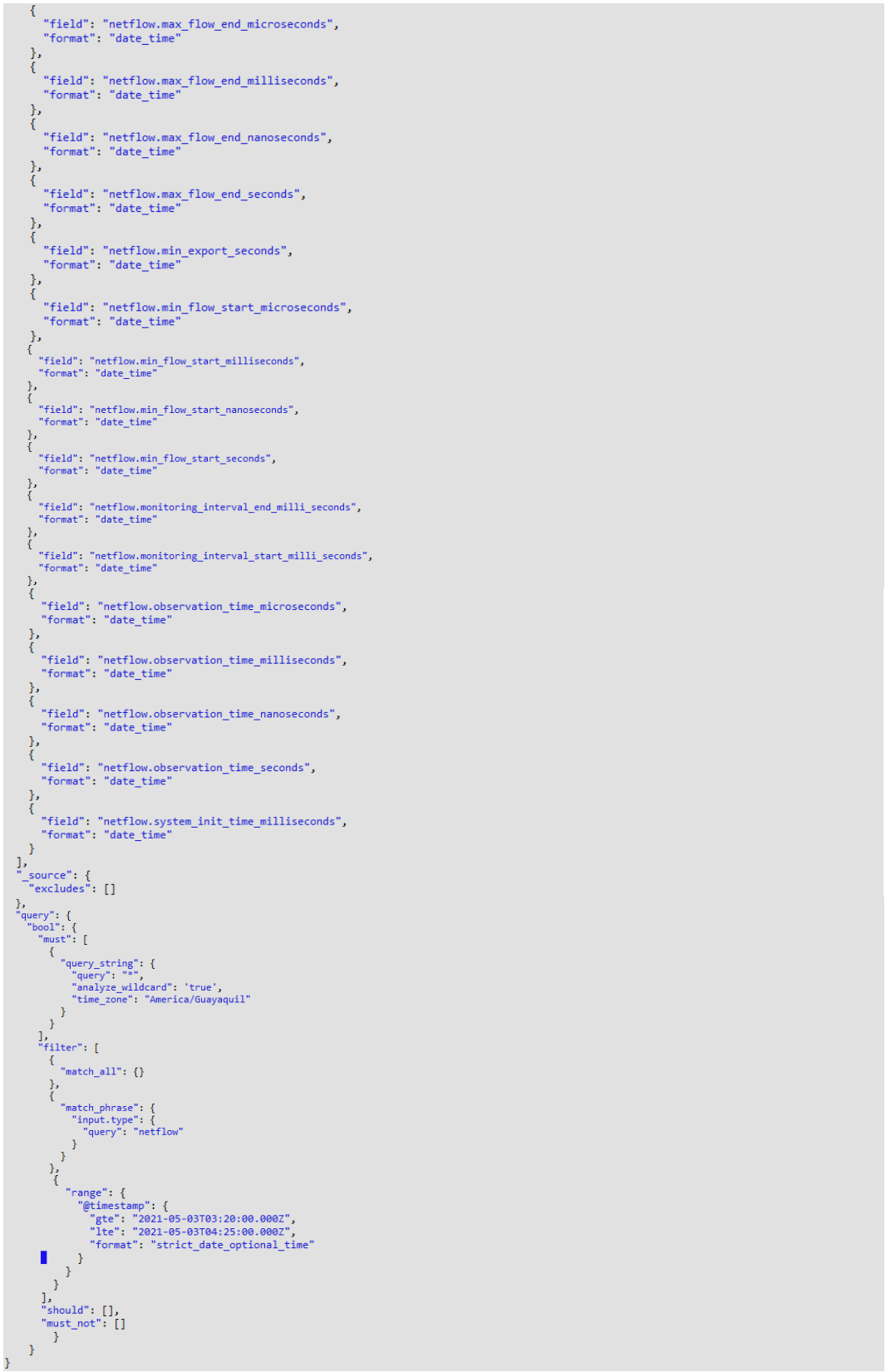


Figura 7. Filtro en Python para obtener las IPs atacantes
Fuente: Elaborado por el investigador

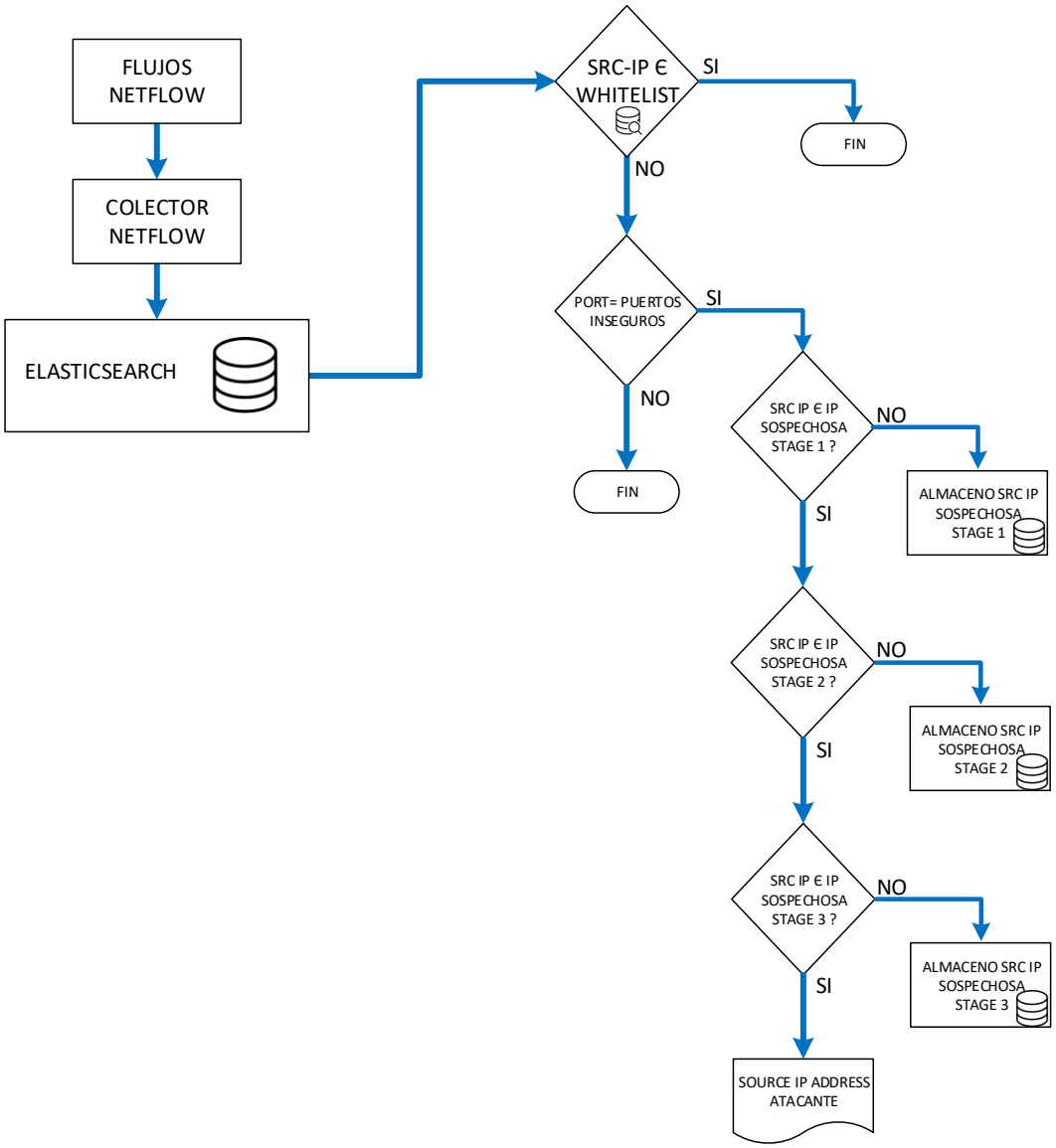


Figura 8. Diagrama de flujo del algoritmo utilizado en el análisis de la información
Fuente: Elaborado por el investigador

Si cada IP de origen tiene tres o más intentos de conexión hacia los puertos mencionados anteriormente, con destino las IPs del ISP, se las toma como atacantes.

En el servidor proporcionado por el ISP se instaló Proxmox; esta plataforma de administración de virtualización es de software libre e integra el hypervisor de KVM y contenedores LXC.

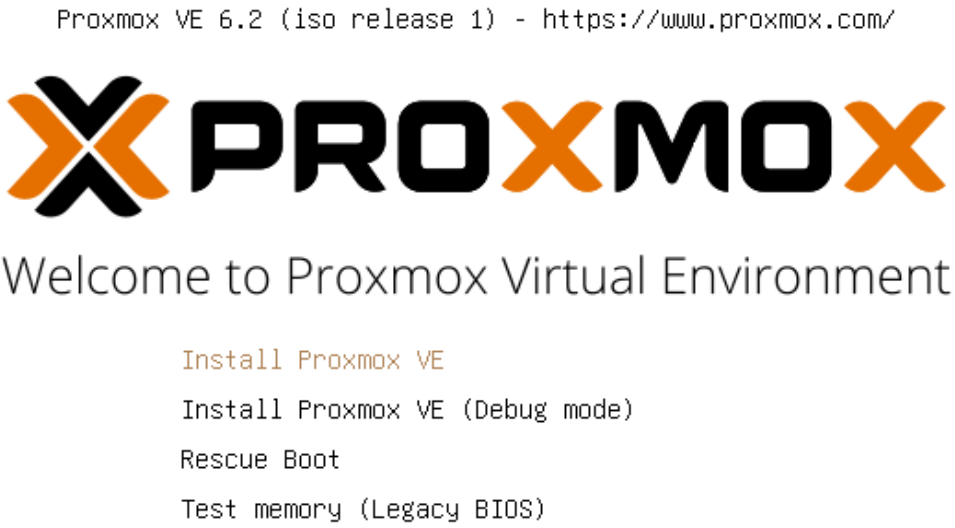


Figura 9. Captura de pantalla del menú para la instalación de Proxmox
Fuente: Tomado del proceso de instalación de Proxmox

La Figura 9 muestra la pantalla principal de instalación de Proxmox. La instalación se hizo con los parámetros por defecto y se integró a un NAS Synology en donde se almacenan los discos duros y los respaldos de las máquinas virtuales.

En el panel de administración de Proxmox se creó una máquina virtual con las características que muestra la Figura 10.

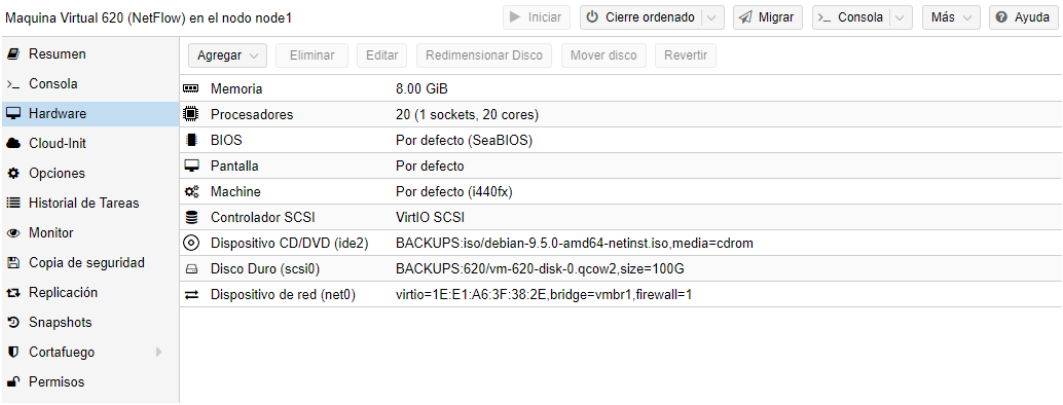


Figura 10. Captura de las características de la máquina virtual utilizada
Fuente: Elaborado por el investigador

El sistema operativo instalado en la máquina virtual es Debian 9 que también es software libre y además ofrece entre otras funciones soporte, seguridad y

estabilidad. Debian es uno de los sistemas operativos que más se usa a nivel mundial y sirve de base para algunas otras distribuciones de Linux.

Ya en la máquina virtual creada e instalado el sistema operativo, se procedió a instalar todas las herramientas para NetFlow, DPI, Elasticsearch y Kibana.

En la Figura 11 se muestra el Dashboard de Kibana que es una herramienta en donde se puede presentar la información que ha sido recolectada con el uso de NetFlow.

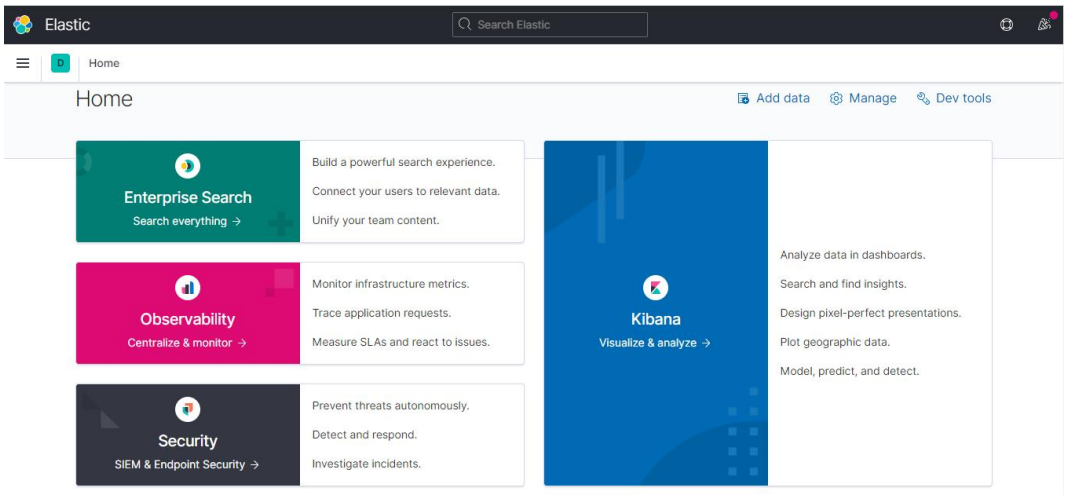


Figura 11. Dashboard principal de Kibana
Fuente: Tomado del dashboard web de Kibana

Para empezar a enviar los flow del Router Mikrotik se hizo la configuración que muestra la Figura 12 la cual indica como destino la dirección IP de la máquina virtual que tiene el colector NetFlow y en donde se va a realizar el análisis de la información.

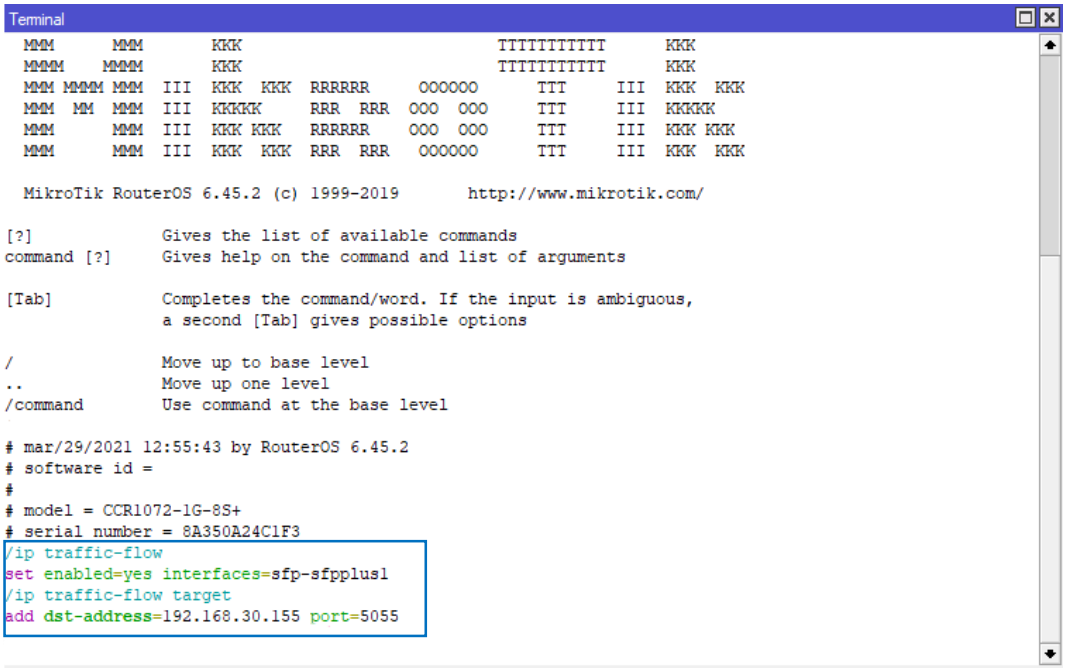


Figura 12. Terminal del equipo de borde Mikrotik del ISP con la configuración de traffic-flow
Fuente: Tomado de la configuración hecha en el equipo de borde del ISP

4. CAPÍTULO IV

4.6 Resultados

En la **Tabla 2** se muestra la cantidad total de flows y bytes recolectados cada día, durante ocho días seguidos a las horas propuestas.

Fecha y hora	Bytes	Cantidad de eventos
2021-05-02 18H00 a 2021-05-02 23H00	3,65 TB	24 239 642
2021-05-03 18H00 a 2021-05-03 23H00	4,13 TB	35 162 562
2021-05-04 18H00 a 2021-05-04 23H00	3,83 TB	26 362 072
2021-05-05 18H00 a 2021-05-05 23H00	4,102 TB	35 083 690
2021-05-06 18H00 a 2021-05-06 23H00	4,107 TB	35 264 762
2021-05-07 18H00 a 2021-05-07 23H00	4,20 TB	35 329 398
2021-05-08 18H00 a 2021-05-08 23H00	4,50 TB	35 209 916
2021-05-09 18H00 a 2021-05-09 23H00	4,44 TB	36 099 948

Tabla 2. Cantidad de eventos y bytes de información recolectada por día.
Fuente: Elaborado por el investigador

De igual forma, en la **Figura 13** se observa el ejemplo de la cantidad promedio de flows y bytes recolectados en un día.

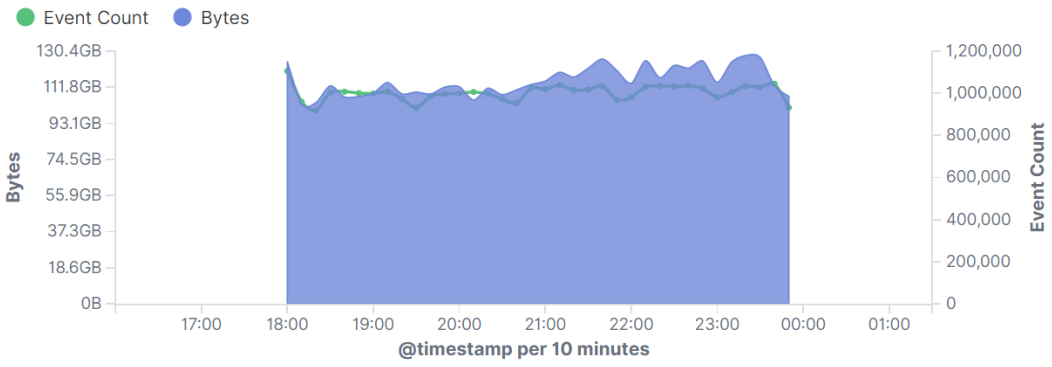


Figura 13. Ejemplo de un día de datos recolectados de 18H00 a 23H00
Fuente: Elaborado por el investigador

Se puede observar que la información que pasó a través del router BGP, supera los 4 Tera Bytes en algunos días, durante las horas que el colector Netflow trabajó. Es por esto que se necesita más recursos de hardware para que la gran cantidad de información pueda ser analizada y responder rápidamente ante un ataque. Los

primeros dos días de recolección de información, coparon toda la capacidad del disco duro que se le asignó a la máquina virtual al principio; para los siguientes días se aumentó la capacidad de 100GB a 800GB lo que permitió completar la recolección de los demás días propuestos.

El uso del CPU se evidencia en la Figura 14. Se puede apreciar que el momento de recolectar la información llega a ocupar aproximadamente el 25% de la capacidad del CPU. Cabe mencionar que se asignó la totalidad de recursos de CPU a la máquina virtual que recolecta los flow's.

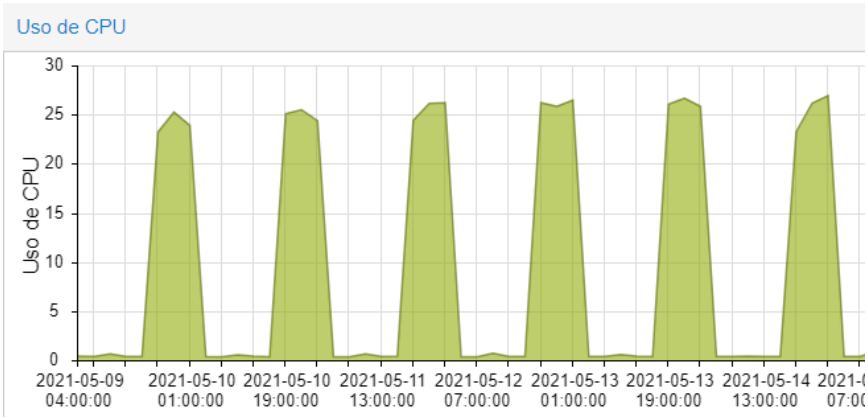


Figura 14. Uso de CPU de la máquina virtual durante la recolección de información
Fuente: Tomado de las gráficas históricas de la VM de Proxmox

Del mismo modo, la memoria asignada a la maquina virtual fue de 8GB lo que se evidencia en la Figura 15. El momento que el recolector NetFlow adquiria la información subia el uso de la memoria RAM.

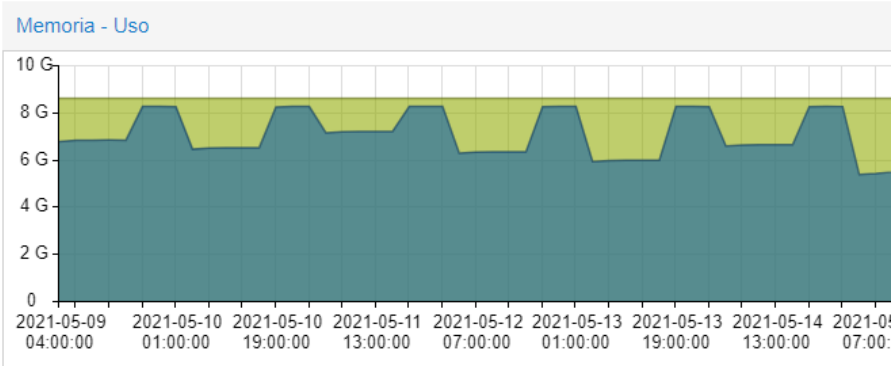


Figura 15. Uso de memoria RAM de la máquina virtual durante la recolección de datos
Fuente: Tomado de las gráficas históricas de la VM de Proxmox

El momento en que mas se hace indispensable contar con la mayor cantidad de memoria RAM, es cuando se analiza la información. El tiempo que demora el análisis de la información, es inversamente proporcional a la cantidad de memoria RAM; es decir, mientras mas recursos de memoria se tenga, menos tiempo se demora en analizar los datos.

El tráfico de red que se genera entre el colector y generador de flow's en promedio es de 1,5Mbps. Relativamente es bajo para la cantidad de tráfico que cursa a través del router Mikrotik. Se muestra el tráfico de red de la maquina virtual en la siguiente Figura 16.

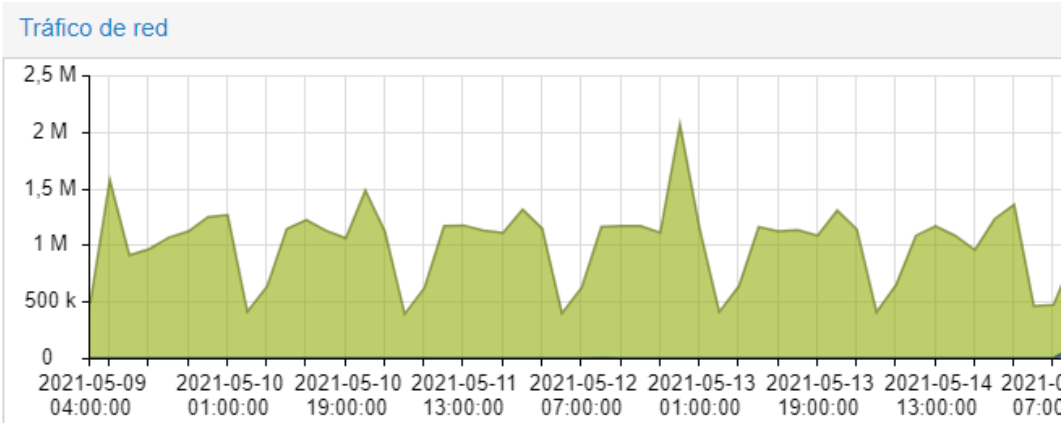


Figura 16. Uso de la interfaz de red de la máquina virtual durante la recolección de datos
Fuente: Tomado de las gráficas históricas de la VM de Proxmox

Kibana ofrece algunas ventajas al momento de presentar la información recolectada a traves de NetFlow, pudiendo personalizar los reportes y realizar gráficos según la necesidad.

En la Figura 17 se muestra una tabla de cómo se puede presentar los datos a través del dashboard de Kibana. Además, se observa la información de IP de fuente y destino en el tiempo.

Time	source.ip	source.port	destination.ip	destination.port	network.transport	network.bytes	network.packets
May 2, 2021 @ 23:58:51.000	8.8.8.8	53	190.99.76.12	40824	udp	1,009B	4
May 2, 2021 @ 23:58:51.000	23.201.103.181	443	190.99.76.18	54974	tcp	232B	4
May 2, 2021 @ 23:58:51.000	190.99.74.195	53	190.99.76.8	54480	udp	127B	1
May 2, 2021 @ 23:58:51.000	148.153.169.131	10012	190.99.76.10	32924	udp	36B	1
May 2, 2021 @ 23:58:51.000	23.50.114.142	443	190.99.78.16	41884	tcp	95.7KB	102
May 2, 2021 @ 23:58:51.000	142.250.78.174	80	190.99.76.6	42064	tcp	247B	3
May 2, 2021 @ 23:58:51.000	172.30.131.245	40600	190.99.72.155	53	udp	64B	1
May 2, 2021 @ 23:58:51.000	157.240.14.53	443	190.99.76.6	45308	tcp	197B	3
May 2, 2021 @ 23:58:51.000	190.99.72.155	53	190.99.78.6	40600	udp	80B	1

Figura 17. Datos presentados en Kibana en forma de tabla
Fuente: Elaborado por el investigador, Kibana

Estos datos son los que se procesaron aplicando el algoritmo propuesto examinando las IPs y los puertos que intervienen en el intercambio de información.

En el dashboard de Kibana también se puede visualizar graficas de los AS (Autonomous Systems) según la cantidad de bytes que están traficando, así como el top de los países y ciudades de las cuales proviene la información

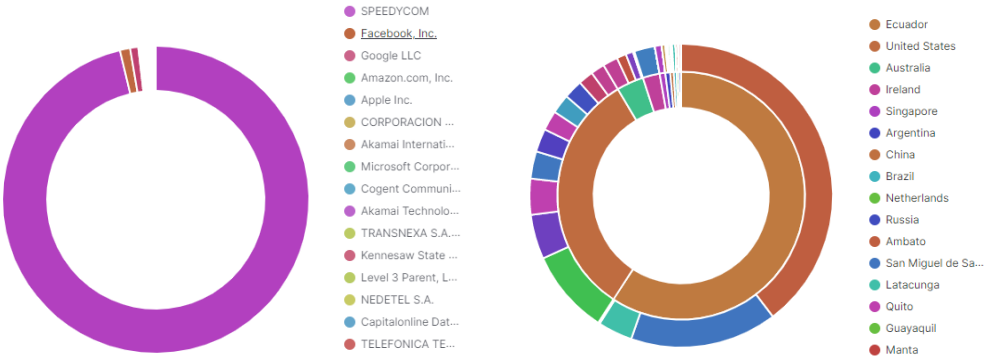


Figura 18. Grafica Top de los Autonomous System, Países y Ciudades
Fuente: Elaborado por el investigador, Kibana

4.1 Análisis de Resultados

Una vez recolectada la información, se analizó y se pudo evidenciar en el router Mikrotik las reglas de firewall agregadas por el script de Python. Esto se muestra en la Figura 19. Para determinar si las IPs que están en la lista de firewall efectivamente se debían bloquear, se buscó si la IP está en blacklist de páginas web conocidas en internet.

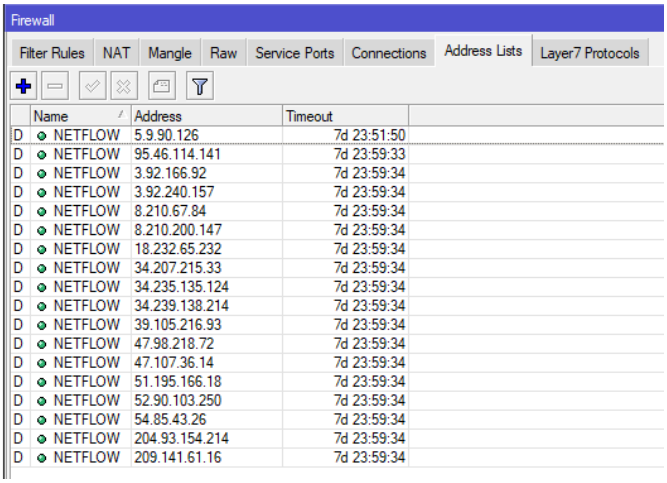


Figura 19. Address List de las IPs agregadas por el script de Python
Fuente: Tomado de la configuración hecha en el equipo de borde del ISP

Se tomó y se buscó una IP en una blacklist que se muestra en la Figura.

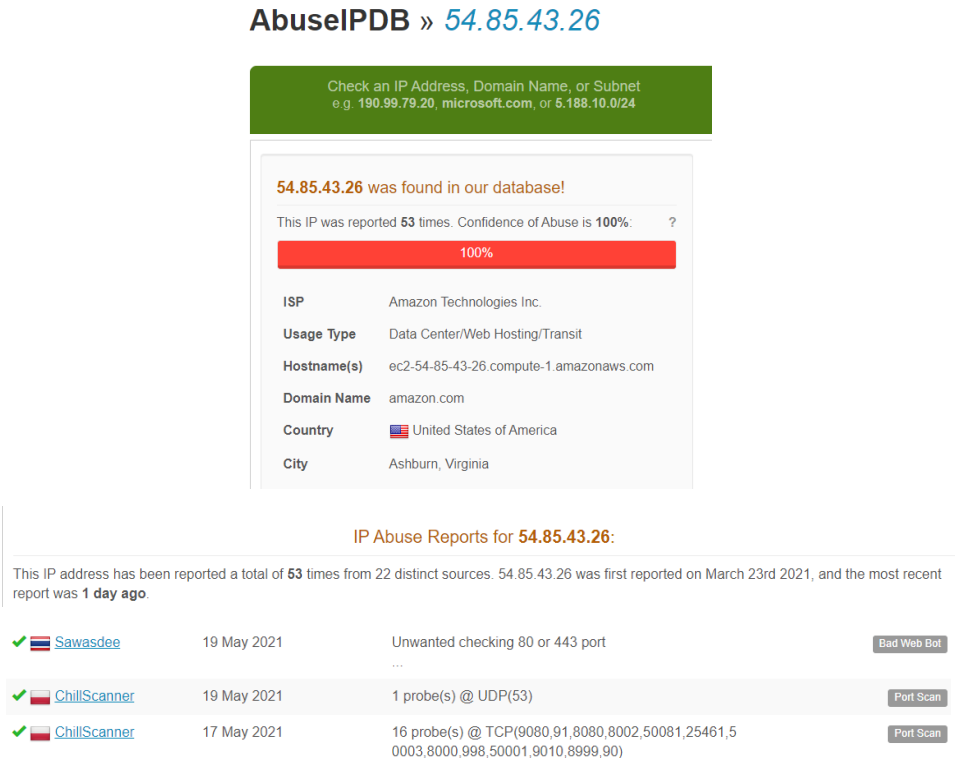


Figura 20. Captura de pantalla del reporte de abuseipdb.com de la IP bloqueada en el firewall del equipo Mikrotik de borde
Fuente: Consultado en abuseipdb.com

Se consultó si la IP esta enlistada en otra base de datos blacklist, como lo muestra la Figura 21, y se puede ver que también se encuentra listada.

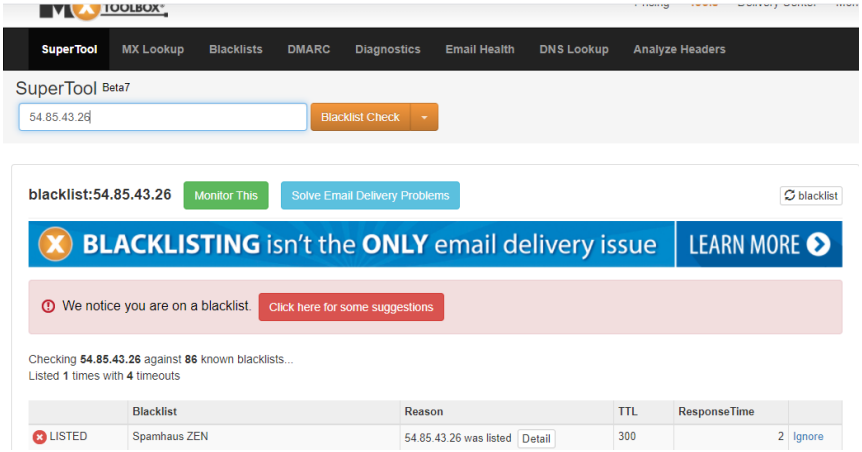


Figura 21. Captura de pantalla del reporte tomado de mxtoolbox de la IP bloqueada
Fuente: Consultado en abuseipdb.com

En la Figura 21 se observa que la IP tomada del firewall si está listada en la base de datos blacklist, lo que confirma que esta IP ha estado atacando a diferentes servidores.

El total de IPs encontradas en el firewall después de analizar los datos recolectados durante los ocho días, fueron de 511 como lo muestra la Figura 22.

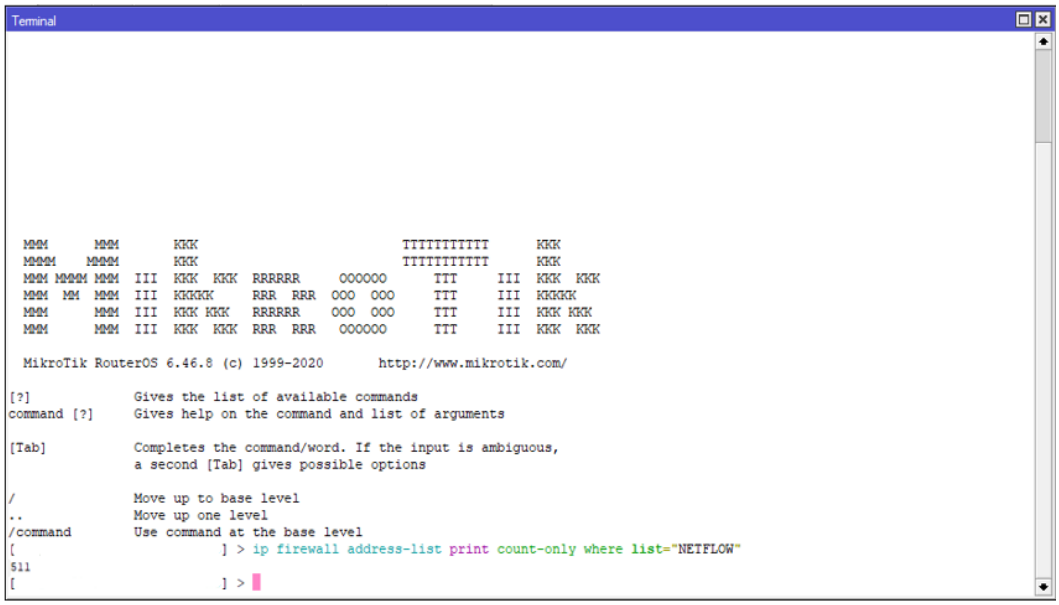


Figura 22. Número de IPs agregadas por el script de Python en el Address List del Mikrotik de borde del ISP
Fuente: Tomado de la configuración hecha en el equipo de borde del ISP

Del total de IPs que fueron agregadas en el firewall de Mikrotik, 38 no se encontró enlistada en ninguna base de datos “blacklist” en el internet.

	N°	Porcentaje
Aciertos IPs bloqueadas	473	92,56 %
Errores IPs bloqueadas	38	7,44 %
Total	511	

Tabla 3. Número de aciertos y errores y sus porcentajes
Fuente: Elaborado por el investigador

En la Tabla 3 se evidencia, al momento de bloquear IPs que están atacando la red del ISP, se encontró que el porcentaje de aciertos es 92,56%. Así mismo, el porcentaje de errores es de 7,44%. El porcentaje de errores se puede tomar como ataques potenciales no comprobados, debido a que no se pudo verificar que las IPs estuvieran enlistadas en “blacklist” y de igual forma, no se puede aseverar que hayan sido de trafico de algún usuario en general.

Como ya se ha indicado anteriormente, los datos que el router Mikrotik del ISP envió durante los ochos días a través de NetFlow, se analizaron cada día. Después del último día de recolección se sumó el total de IPs detectadas para obtener el porcentaje de aciertos y errores. Este experimento se puede repetir para más días con un mayor almacenamiento.

El modelo de red que muestra la **Figura 5**, muestra que se puede obtener la información de forma eficiente al ser una implementación basado en un hardware que lo permite. Esa información se puede enviar a la nube donde se puede contar con mayores recursos para almacenamiento. La forma en que se recuperan los datos en el presente trabajo constituye en sí una contribución válida en cuanto a detección de ataques DoS se refiere. Este mecanismo podría integrarse satisfactoriamente con otros algoritmos de detección, utilizando los datos recopilados de la forma propuesta en este proyecto.

5. CAPÍTULO V

5.2 Conclusiones

- El Protocolo NetFlow es una herramienta que puede ser usada con diferentes fines, entre ellos el poder detectar ataques a una red de interés. En este proyecto se ha logrado analizar los flows que envía el router Mikrotik de borde del ISP. La cantidad de flow's que se puede analizar depende de la cantidad de recursos de hardware que se tenga en el colector NetFlow. La cantidad de información que atraviesa el router Mikrotik de borde del ISP, hizo que el colector NetFlow se llenara rápidamente por lo que se tuvo que limitar el tiempo de recolección.

- Los ataques DoS y DDoS se enfocan en agotar los recursos que existen en una red, ya sea del hardware de los equipos o del ancho de banda disponible. Para poder detectar los ataques se implementó un colector NetFlow en el cual también se analizó los datos en busca de posibles ataques. De esta manera se hizo posible quitar del router Mikrotik de borde del ISP, la carga de buscar ataques en la información que lo atravesaba.

- La aplicación que se llegó a obtener nos da un porcentaje de aciertos en la detección de ataques sobre el 92%, lo que nos permite asegurar la red y los equipos del ISP. En esta aplicación, con ayuda de Elasticsearch y Kibana, también se pudo tener algunos datos interesantes para el ISP como: servicios más usados por parte de los usuarios, países y ciudades con mayor tráfico hacia la red del ISP, AS con la mayor cantidad de tráfico recibido, entro otras.

5.3 Recomendaciones

- Si se dispone de mayor cantidad de recursos de hardware, se puede realizar un análisis más profundo de la información incluso con un algoritmo más complejo.
- Para mejorar la precisión en la detección de ataques, se puede hacer uso de Machine Learning. Esto permite conocer el normal comportamiento de una red y si hay una variación, con el uso de Machine Learning, se podría acelerar la detección y evitar daños severos en la red de interés.
- Se puede tener mayor flexibilidad utilizando Redes Definidas por Software (SDN por sus siglas en inglés) y, en conjunto con DPI es posible identificar patrones de tráfico como ataques y a su vez ir agregando reglas de filtrado en los equipos controlados con SDN.
- NetFlow es una herramienta muy útil por su poco consumo de red y versatilidad al momento de recolectar la información. Para analizar cada paquete que atraviesa una red, se puede utilizar un middlebox. Midlebox se trata de un equipo tipo “bridge” por donde pasa toda la información que trafica una red y no es necesario enviarlo hacia un colector.

6. BIBLIOGRAFÍA

[1] "NETSCOUT netscout.com https://www.netscout.com/threatreport." .

[2] "Cisco Annual Report https://www.cisco.com https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html." .

[3] Y. Guo, Y. Gao, Y. Wang, M. Qin, Y. Pu, Z. Wang, D. Liu, X. Chen, T. Gao, T. Lv, and others, "DPI & DFI: a malicious behavior detection method combining deep packet inspection and deep flow inspection," *Procedia engineering*, vol. 174, pp. 1309–1314, 2017.

[4] R. T. El-Maghraby, N. M. Abd Elazim, and A. M. Bahaa-Eldin, "A survey on deep packet inspection," in *2017 12th International Conference on Computer Engineering and Systems (ICCES)*, 2017, pp. 188–197.

[5] C. D. McDermott, W. Haynes, and A. V. Petrovksi, "Threat Detection and Analysis in the Internet of Things using Deep Packet Inspection.," *IJCSA*, vol. 3, no. 1, pp. 61–83, 2018.

[6] C. Xu, S. Chen, J. Su, S.-M. Yiu, and L. C. Hui, "A survey on regular expression matching for deep packet inspection: Applications, algorithms, and hardware platforms," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2991–3029, 2016.

[7] E. Özer and M. Iskefiyeli, "Detection of DDoS attack via deep packet analysis in real time systems," in *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 1137–1140.

[8] J. Hou, P. Fu, Z. Cao, and A. Xu, "Machine learning based ddos detection through netflow analysis," in *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*, 2018, pp. 1–6.

[9] D. S. Terzi, R. Terzi, and S. Sagiroglu, "Big data analytics for network anomaly detection from netflow data," in *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 592–597.

[10] Z. Tian, "Management of large scale NetFlow data by distributed systems," NTNU, 2016.

[11] Y.-T. Wang, C.-T. Yang, E. Kristiani, M.-L. Liu, C.-H. Lai, W.-J. Jiang, and Y.-W. Chan, "The Implementation of NetFlow Log System Using Ceph and ELK Stack," in *International Conference on Frontier Computing*, 2018, pp. 256–265.

[12] K. Singh, K. S. Dhindsa, and B. Bhushan, "Distributed Defense: An Edge over Centralized Defense against DDos Attacks.," *International Journal of Computer Network & Information Security*, vol. 9, no. 3, 2017.

[13] J. Sherry, C. Lan, R. A. Popa, and S. Ratnasamy, "Blindbox: Deep packet inspection over encrypted traffic," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 213–226.

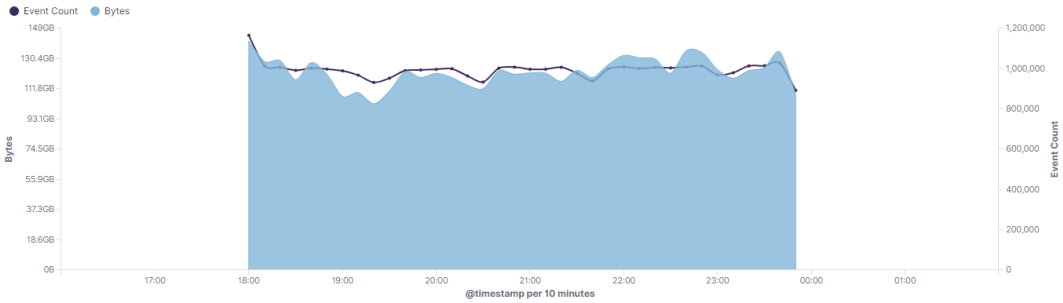
[14] G. D. L. T. Parra, P. Rad, and K.-K. R. Choo, "Implementation of deep packet inspection in smart grids and industrial Internet of Things: Challenges and opportunities," *Journal of Network and Computer Applications*, vol. 135, pp. 32–46, 2019.

[15] “NetFlow eTutoriales.org http://etutorials.org/Networking/network+management/Part+II+Implementations+on+the+Cisco+Devices/Chapter+7.+NetFlow/Fundamentals+of+NetFlow/.” .

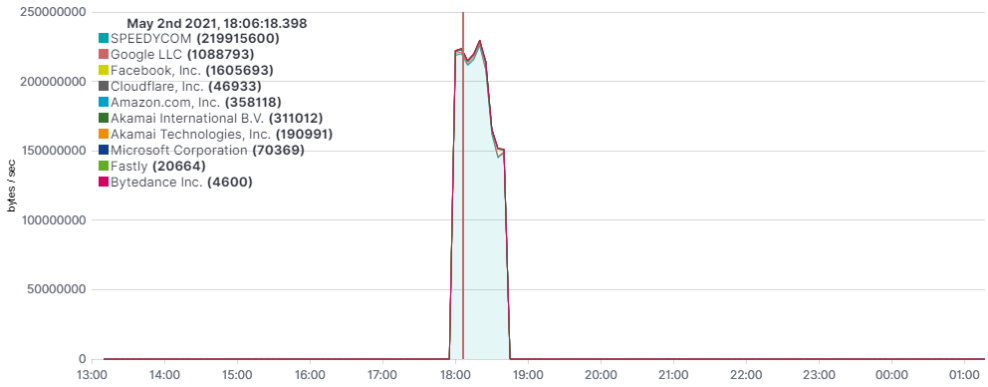
7. ANEXOS

7.1 ANEXO 1

GRÁFICAS DE LOS DATOS RECOLECTADOS DE CADA DÍA



Autonomous System

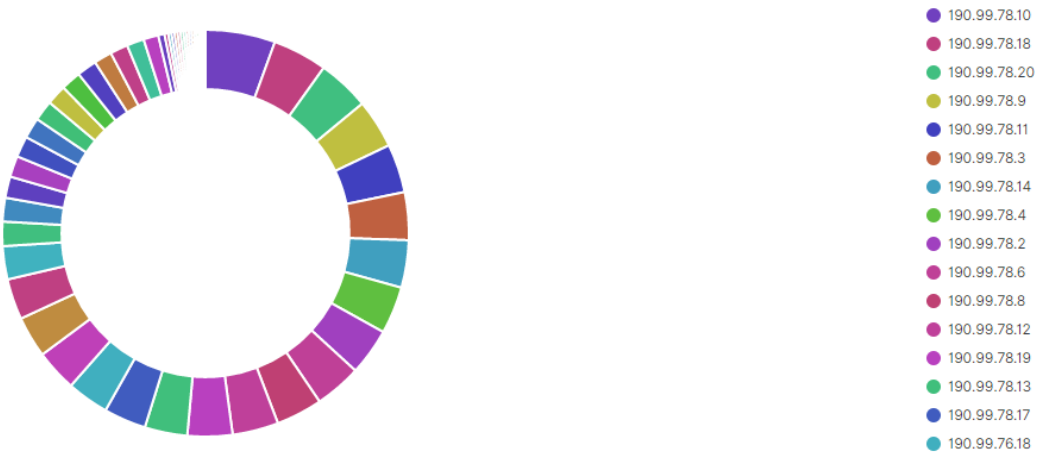


Source Ports

Destination Ports

IP Source

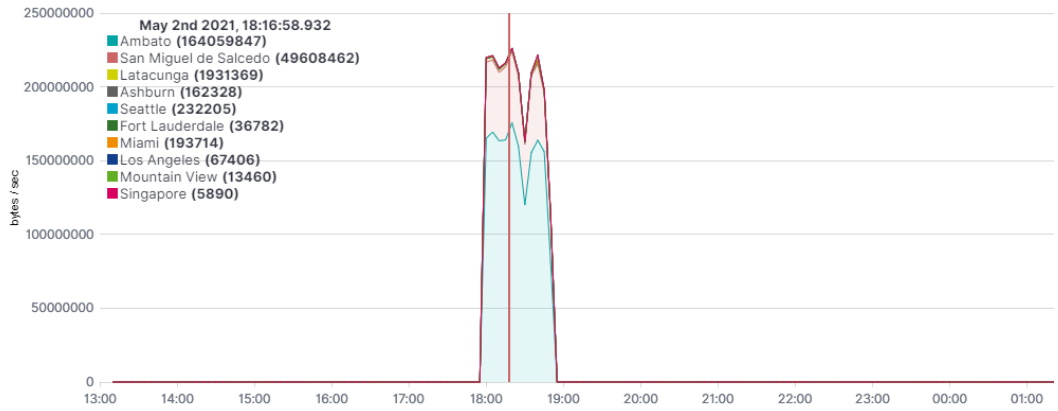
IP Destination

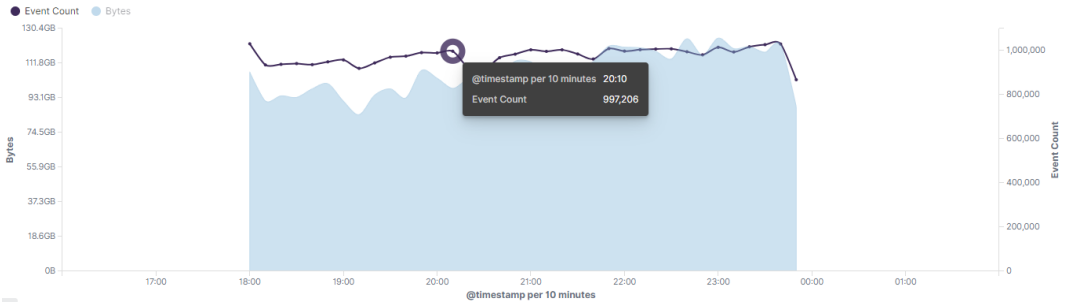


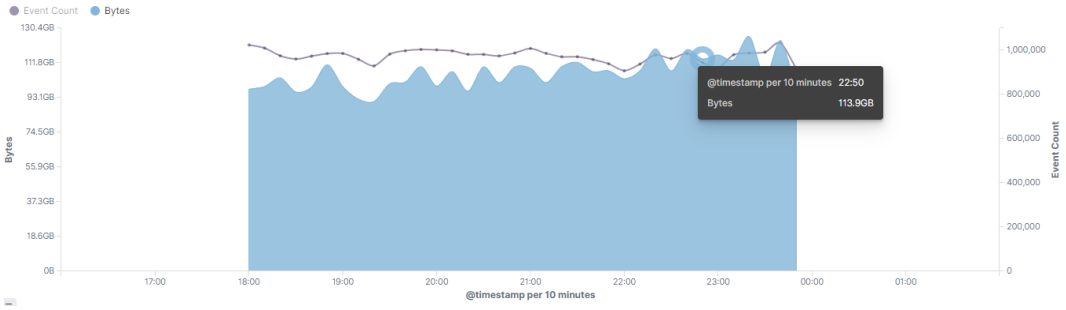
Países

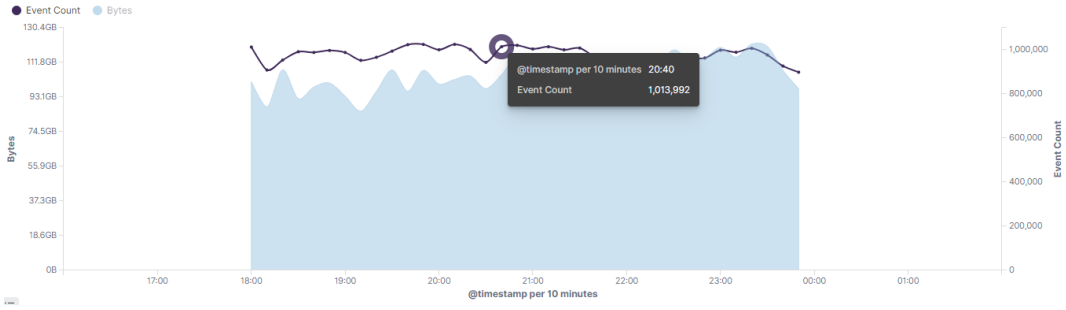

May 2nd 2021, 18:15:20.388
Ecuador (217234742)
United States (5288931)
Australia (6429)
Ireland (543914)
Singapore (36049)
Argentina (189852)
China (7737)
Brazil (13812)
Netherlands (34967)
Russia (2222)
bytes / sec
250000000
200000000
150000000
100000000
50000000
0
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00

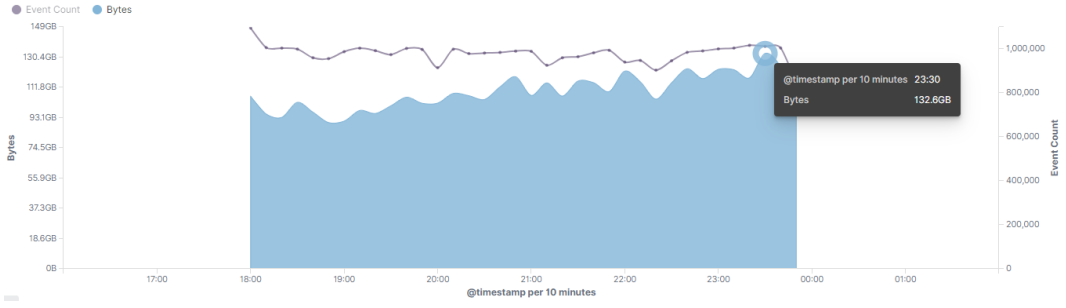
Ciudades

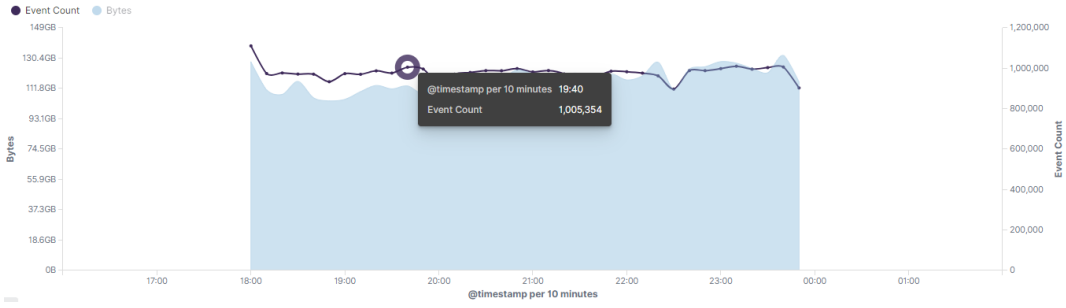

May 2nd 2021, 18:16:58.932
Ambato (164059847)
San Miguel de Salcedo (49608462)
Latacunga (1931369)
Ashburn (162328)
Seattle (232205)
Fort Lauderdale (36782)
Miami (193714)
Los Angeles (67406)
Mountain View (13460)
Singapore (5890)
bytes / sec
250000000
200000000
150000000
100000000
50000000
0
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00

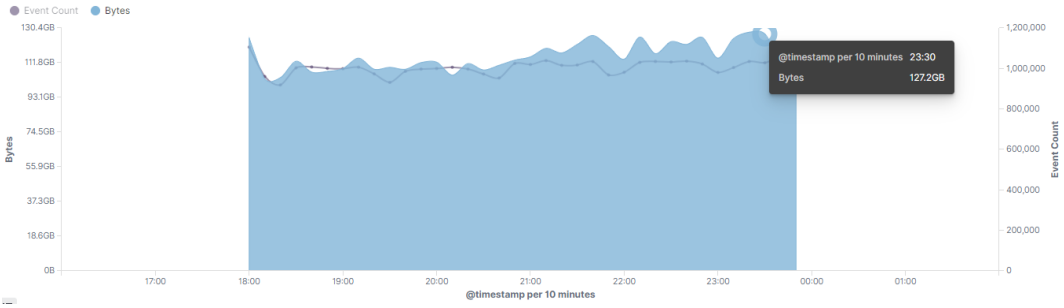

Event Count
Bytes
@timestamp per 10 minutes 20:10
Event Count 997,206
Bytes
Event Count
@timestamp per 10 minutes

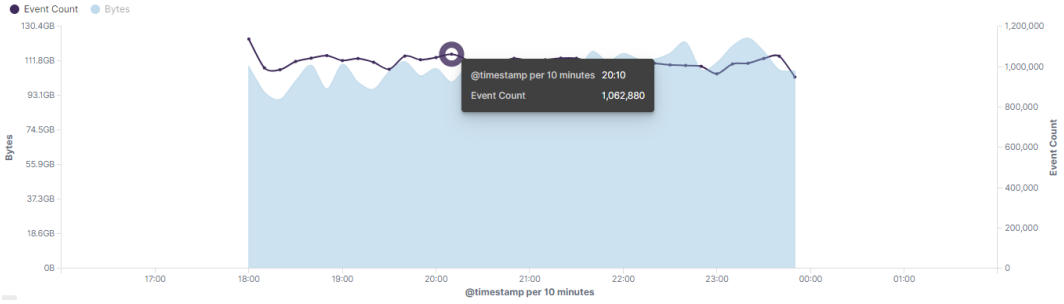

Event Count
Bytes
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 22:50
Bytes 113.9GB


Event Count
Bytes
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 20:40
Event Count 1,013,992


Event Count
Bytes
149GB
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 23:30
Bytes 132.6GB


Event Count
Bytes
149GB
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,200,000
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 19:40
Event Count 1,005,354


Event Count
Bytes
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,200,000
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 23:30
Bytes 127.2GB


Event Count
Bytes
130.4GB
111.8GB
93.1GB
74.5GB
55.9GB
37.3GB
18.6GB
0B
Bytes
1,200,000
1,000,000
800,000
600,000
400,000
200,000
0
Event Count
17:00
18:00
19:00
20:00
21:00
22:00
23:00
00:00
01:00
@timestamp per 10 minutes
@timestamp per 10 minutes 20:10
Event Count 1,062,880


tcp
udp
icmp
unknown (47)
unknown (89)
443
80
5228
5222
5223
853
8001
53
993
6800
8820

7.2 ANEXO 2 SCRIPT DE PYTHON

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

from elasticsearch import Elasticsearch
import datetime
import routeros_api

connection = routeros_api.RouterOsApiPool(
    host='IP_HOST',
    username='user_netflow',
    password='pass_netflow',
    port=8728,
    plaintext_login=True,

client = Elasticsearch(hosts=["localhost:9200"])

filter1 = {

  "version": true,
  "size": 10000,
  "sort": [
    {
      "@timestamp": {
        "order": "desc",
        "unmapped_type": "boolean"
      }
    }
  ],
  "stored_fields": [
    "*"
  ],
  "script_fields": {},
  "docvalue_fields": [
    {
      "field": "@timestamp",
      "format": "date_time"
    },
    {
      "field": "netflow.collection_time_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.exporter.timestamp",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_microseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_end_nanoseconds",
      "format": "date_time"
    },
   {
      "field": "netflow.flow_end_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.flow_start_microseconds",
      "format": "date_time"
    },
```

```
      {
        "field": "netflow.flow_start_milliseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.flow_start_nanoseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.flow_start_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.max_export_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.max_flow_end_microseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.max_flow_end_milliseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.max_flow_end_nanoseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.max_flow_end_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.min_export_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.min_flow_start_microseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.min_flow_start_milliseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.min_flow_start_nanoseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.min_flow_start_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.monitoring_interval_end_milli_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.monitoring_interval_start_milli_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.observation_time_microseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.observation_time_milliseconds",
        "format": "date_time"
      },
```

```
      {
        "field": "netflow.observation_time_nanoseconds",
        "format": "date_time"
      },
      {
        "field": "netflow.observation_time_seconds",
        "format": "date_time"
      },
      {
        "field": "netflow.system_init_time_milliseconds",
        "format": "date_time"
      }
    ],
    "_source": {
      "excludes": []
    },
    "query": {
      "bool": {
        "must": [
          {
            "query_string": {
              "query": "*",
              "analyze_wildcard": 'true',
              "time_zone": "America/Guayaquil"
            }
          }
        ],
        "filter": [
          {
            "match_all": {}
          },
          {
            "match_phrase": {
              "input.type": {
                "query": "netflow"
              }
            }
          },
       {
            "match_phrase": {
              "destination.port": 80
            }
       },
          {
            "range": {
              "@timestamp": {
                "gte": "2021-05-03T03:20:00.000Z",
                "lte": "2021-05-03T04:25:00.000Z",
                "format": "strict_date_optional_time"
              }
            }
          }
        ],
        "should": [],
        "must_not": []
       }
    }

}
filter2 ={

  "aggs": {
    "4": {
      "terms": {
        "field": "source.port",
        "order": {
          "2": "desc"
        },
        "size": 10000
      },
```

```
    "aggs": {
      "2": {
        "sum": {
          "field": "network.bytes"
        }
      },
      "3": {
        "sum": {
          "field": "network.packets"
        }
      }
    }
  }
},
"size": 0,
"stored_fields": [
  "*"
],
"script_fields": {},
"docvalue_fields": [
  {
    "field": "@timestamp",
    "format": "date_time"
  },
  {
    "field": "netflow.collection_time_milliseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.exporter.timestamp",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_end_microseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_end_milliseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_end_nanoseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_end_seconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_start_microseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_start_milliseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_start_nanoseconds",
    "format": "date_time"
  },
  {
    "field": "netflow.flow_start_seconds",
    "format": "date_time"
  },
  {
    "field": "netflow.max_export_seconds",
    "format": "date_time"
  },
  {
    "field": "netflow.max_flow_end_microseconds",
    "format": "date_time"
  },
```

```
    {
      "field": "netflow.max_flow_end_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.max_flow_end_nanoseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.max_flow_end_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.min_export_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.min_flow_start_microseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.min_flow_start_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.min_flow_start_nanoseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.min_flow_start_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.monitoring_interval_end_milli_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.monitoring_interval_start_milli_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.observation_time_microseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.observation_time_milliseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.observation_time_nanoseconds",
      "format": "date_time"
    },
    {
      "field": "netflow.observation_time_seconds",
      "format": "date_time"
    },
    {
      "field": "netflow.system_init_time_milliseconds",
      "format": "date_time"
    }
],
  "_source": {
    "excludes": []
  },
  "query": {
    "bool": {
      "must": [
```

```
        {
          "query_string": {
            "query": "*",
            "analyze_wildcard": 'true',
            "time_zone": "America/Guayaquil"
          }
        }
      ],
  "filter": [
          {
            "match_all": {}
          },
          {
            "match_phrase": {
              "input.type": {
                "query": "netflow"
              }
            }
          },
       {
            "match_phrase": {
              "destination.port": 80
            }
       },
          {
            "range": {
              "@timestamp": {
                "gte": "2021-05-03T03:20:00.000Z",
                "lte": "2021-05-03T04:25:00.000Z",
                "format": "strict_date_optional_time"
              }
            }
          }
        ],
        "should": [],
        "must_not": []
       }
      }
  }

  res= client.search(index='*', body=filter1)

  response_hits = res['hits']['hits']
  print ('\n, number of hits:', len(response_hits))
  for num, doc in enumerate(response_hits):
      print ('\n, num:', num)
      doc_id = doc['_id']
      print ('_id:', doc_id)
      print ('VIN:', doc['_source']['source']['ip'])
      api = connection.get_api()
      list_address.add(list='NETFLOW', address=doc['_source']['source']['ip'],
timeout='8d')

  #end of coding
```