UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

TÍTULO: Mitigación de ataques de denegación de servicio distribuidos en la nube

AUTORES: Cristian Daniel Barbaro y Mateo Durante

DIRECTOR: Nicolás Macia

CODIRECTOR: Alejandro Sabolansky

ASESOR PROFESIONAL: –

CARRERA: Licenciatura en Sistemas

Resumen

Los costos de los servicios de mitigación de ataques de DDoS generalmente resultan difíciles de afrontar para pequeñas y medianas organizaciones. Debido a esto, se propone el desarrollo de una plataforma utilizando estándares abiertos y software libre para proveer un servicio propio de mitigación de DDoS y que distintos tipos de organizaciones, de manera individual o coordinada, puedan utilizarlo. Esta plataforma se conforma de diversas componentes, entre las que sobresalen los nodos de limpieza de tráfico distribuidos administrables desde una única interfaz web.

Palabras Clave

Mitigación, ataques de denegación de servicio, ataques de denegación de servicio distribuido, plataforma distribuida, cooperativo, open source, seguridad, BGP, firewall, limpieza de tráfico de red, filtrado, scrubbing center, web, túnel, FlowSpec, ISP.

Conclusiones

Se logró la integración de diversas herramientas que permiten resolver el problema de un ataque de DDoS, evitando interacciones con ISPs y terceros. La componente de la plataforma denominada WebScrub simplifica las acciones del usuario evitándole los inconvenientes que conlleva ejecutar instrucciones en dispositivos y le presenta al usuario una interfaz gráfica con la información necesaria para que pueda actuar en consecuencia cuando lo considere oportuno.

Trabajos Realizados

Se desarrolló una plataforma de software para organizaciones que requieran adquirir servicios para mitigar ataques de DDoS. La misma se conforma de una interfaz web que permite administrar las acciones que requieran los usuarios finales, los nodos de limpieza de tráfico controlados remotamente, APIs para el intercambio de métricas del sistema, la abstracción del uso de los firewalls y creación semiautomática de túneles para la redirección de tráfico. Para validar el desarrollo, se diseñó e implementó una maqueta que permite instanciar la plataforma con un ataque de DDoS y así, probar su funcionalidad en un entorno controlado.

Trabajos Futuros

- Puesta en producción de la plataforma en Internet.
- Evaluar las capacidades de filtrado de posibles firewalls.
- Ampliar los tipos de reglas de filtrado de tráfico.
- Desplegar la plataforma entre miembros de redes de investigación para que cada miembro aproveche y colabore con el mecanismo de mitigación de ataques de DDoS.
- Extender las capacidades de accounting de uso del sistema.
- Determinar ubicaciones geográficas adecuadas para la instalación de los scrubbing center.

Fecha de la presentación: Mayo 2022

Mitigación de ataques de denegación de servicio distribuidos en la nube

Por

Cristian Daniel Barbaro
Mateo Durante

Facultad de Informática
Universidad Nacional de La Plata

Directores: Alejandro Sabolansky. Nicolás Macia

Tesina de Licenciatura en Sistemas

Mayo 2022

Tabla de Contenidos

CAPÍTULO 1

MOTIVACIÓN

Actualmente los ataques de DDoS (denegación de servicio distribuido o Distributed Denial of Service) pueden ser un gran problema para toda clase de organizaciones que sean víctimas de los mismos. Un ataque de DDoS puede impedir el normal desempeño de las actividades en Internet de una organización o provocar una baja reputación en la comunidad debido a la poca estabilidad de los servicios que presta, ya que se produce una sobrecarga de procesamiento y uso de recursos de los dispositivos involucrados provocando que se vea afectada la calidad del servicio a usuarios finales, incrementando los tiempos de acceso, en algunos casos hasta el punto de no poder acceder al mismo.

Los ataques tienen diferentes objetivos o motivaciones. Entre los más comunes se pueden destacar el ataque a una organización con el fin de causarle pérdidas monetarias, ocasionarle daños de imagen o prestigio, o simplemente tratarse de un daño colateral de un ataque mayor.

Existen diversas técnicas que pueden ser utilizadas, una de las más conocidas es la inundación de datagramas IP de diferentes orígenes generando grandes cantidades de tráfico hacia la víctima.

En la actualidad, las organizaciones susceptibles a ataques de DDoS pueden adquirir un servicio de limpieza de tráfico a un proveedor especializado en el tema a través del pago de un abono mensual. Este tipo de servicios que se brinda en diversas modalidades, se ocupa de mitigar los ataques dirigidos a la organización objetivo.

Los servicios comerciales existentes para mitigar los ataques de DDoS en la nube son provistos por compañías que tienen recursos de red y cómputo en distintas regiones del mundo. Estos servicios están enfocados en la protección de grandes empresas y organizaciones. El costo de estos servicios generalmente resulta inalcanzable para pequeñas y medianas empresas. Debido a esto, los autores de este texto se encuentran motivados en desarrollar un servicio propio de mitigación de DDoS para que distintos tipos de organizaciones, de manera individual o coordinada, puedan utilizarlo.

Objetivo

El objetivo de esta tesina es desarrollar una plataforma de software que sea económicamente accesible para todo tipo de organizaciones que requieran adquirir servicios para mitigar ataques de denegación de servicio distribuidos que pueden afectarles directa o indirectamente. Esta plataforma comprende un conjunto de aplicativos distribuidos que se alojarán en la red del usuario y en la nube. Las mismas se orquestarán utilizando estándares y tecnologías abiertas.

De esta manera, al desarrollar todas estas herramientas como software libre [1] se realizará un aporte a la comunidad y se espera que las mismas conformen una alternativa para pequeñas y medianas organizaciones a los servicios de filtrado de tráfico arancelados, los cuales generalmente tienen un costo prohibitivo de afrontar.

Para llevar a cabo este desarrollo, se analizará en el contexto de ciberseguridad las problemáticas que generan los ataques de denegación de servicio distribuidos a las organizaciones víctimas, las soluciones disponibles y las necesidades que estas requieren para que sean implementadas.

Organización del documento

La presente tesina está dividida por capítulos que abordan el marco teórico, el estado del arte y el desarrollo de la herramienta propuesta. A continuación se detalla de forma concisa la estructura y los contenidos de cada uno de ellos.

Capítulo 2, Marco teórico Se introducirán diferentes conceptos que se consideran necesarios abordar para poder avanzar en los temas tratados en los capítulos siguientes. Se definirá qué son los ataques de denegación de servicio, qué o quiénes son los Sistemas Autónomos, a quiénes se denominan Proveedores de Servicio de Internet, qué es un Firewall, a qué se hace referencia cuando se menciona ancho de banda y, finalmente, qué son los túneles de red.

Capítulo 3, Estado del arte En este capítulo se mencionarán y estudiarán herramientas, trabajos y soluciones existentes relacionados con los temas principales de esta tesina.

Capítulo 4, ScrubbingUNLP En este capítulo se explica la plataforma que se planea ofrecer a la comunidad, cómo es el desarrollo de la misma, su arquitectura, componentes y cómo estos últimos se comunican.

Capítulo 5, Entorno de pruebas En el capítulo 5 se desarrolla un entorno de simulación sobre los que se prueba el trabajo. Se mostrarán las métricas obtenidas y se describirán las ventajas de aplicar los sistemas de filtrado de la plataforma desarrollada.

Capítulo 6, Conclusiones y Trabajo futuro En este último capítulo se presentarán las conclusiones obtenidas durante el trabajo de investigación e implementación plasmado en la tesina y además se delinearán actividades que podrían llevarse adelante para continuar con la línea de trabajo.

CAPÍTULO 2

MARCO TEÓRICO

En el presente capítulo se detallarán los componentes importantes de Internet que se encuentran involucrados cuando ocurre un ataque de denegación de servicio distribuido. También se profundizará en los diferentes tipos de ataques y sus variantes. Basándose en estos conceptos, en los sucesivos capítulos se describirá con mayor precisión los componentes de la solución propuesta.

2.1 Puntos claves de la estructura de Internet

2.1.0.1 Sistema Autónomo

Un sistema autónomo (o AS por sus siglas en inglés, *Autonomous System*) es un grupo conectado de uno o más prefijos de IP administrados por uno o más operadores de red que tiene una política de enrutamiento única y claramente definida [2]. Los AS poseen un número denominado ASN (*Autonomous System Number*) que funciona como identificador y son asignados por los Registros Regionales de Internet (RIRs). Por ejemplo, la Universidad Nacional de La Plata posee el ASN 5692 y publica los prefijos IPv4 163.10.0.0/16 e IPv6 2800:340::/32.

Los AS pueden ser categorizados según las conexiones y políticas usadas sobre el tránsito del tráfico en tres clases:

- Los *stub AS* son aquellos que se conectan únicamente con otro AS para alcanzar la conexión a Internet.
- Los *multihomed AS* son aquellos que se conectan a más de un AS, sin permitir el tránsito entre ellos. Usualmente se utiliza esta configuración para tener redundancia de acceso a Internet con múltiples proveedores.
- Los *transit AS* son aquellos que se conectan con múltiples AS permitiendo el tránsito entre ellos. Esta es la configuración utilizada por la mayoría de los grandes proveedores de servicio de Internet.

2.1.0.2 Proveedores de servicio de Internet

Un proveedor de servicio de Internet (o ISP por sus siglas en inglés, *Internet Service Provider*) es una organización que brinda servicios de conexión a Internet a sus clientes. Los ISPs son categorizados por el *tier* al que pertenecen y esto representa con quiénes se conectan y a quiénes les brindan servicios. Existen tres niveles de *tiers*, donde el primer nivel representa a unas pocas organizaciones interconectadas a nivel global que brindan conectividad a otras organizaciones, generalmente regionales de *tiers* nivel dos. A su vez, estos *tiers* nivel dos se interconectan y brindan conectividad a otros de nivel tres. Estos últimos finalmente dan conectividad a los ISPs locales. Estos ISPs locales son quienes proveen conexión a Internet a los usuarios residenciales u organizaciones.

Cada uno de estos ISPs puede contar con uno o más ASNs para poder ser identificados en Internet.

2.1.0.3 Punto de intercambio de tráfico

Un Punto de Intercambio de Tráfico (IXP, del inglés Internet Exchange Point) [3][4] es un componente de la infraestructura de Internet que puede mejorar la asequibilidad y calidad de la Internet para las comunidades locales. Los IXPs permiten que redes locales y organizaciones, como pueden ser operadores, proveedores de acceso, organismos del gobierno, entidades académicas, etc., intercambien información de manera eficiente en un punto común dentro de un país, sin la necesidad de intercambiar el tráfico de Internet local en el extranjero. A su vez, esto también sirve para reducir costos y mejorar el servicio a los clientes.

Por ejemplo, el IXP permite reducir los costos y mejorar el intercambio de tráfico entre los miembros que se encuentran interconectados con una menor cantidad de intermediarios. Por otro lado, el IXP también admite la implementación de servicios de CDN [5] y caché en los puntos de intercambio beneficiando a todos los conectados al mismo sin necesidad de consumir tráfico de sus ISPs correspondientes.

2.1.0.4 Firewall

Los *firewalls* son sistemas informáticos que permiten permiten tomar acciones con el tráfico de red que lo atraviesa a partir de la definición de políticas y reglas específicas que pueden implementarse. Entre las principales acciones que un firewall puede realizar se encuentra el filtrado de tráfico no autorizado de un origen a un destino.

Existen diferentes tipos y generaciones de *firewalls*. Las primeras generaciones permitían definir reglas únicamente relacionadas con campos del datagrama IP y del protocolo de transporte. Sin embargo, a lo largo del tiempo, las capacidades de cómputo de los *firewalls* han evolucionado notoriamente y, junto con el avance en el análisis de tráfico, actualmente permiten implementar una gran cantidad de funcionalidades adicionales entre las que se pueden destacar el filtrado a nivel de protocolo de aplicación, seguimiento de conexiones con estado, soporte para NAT, terminadores de VPN, entre otras.

2.1.0.5 Ancho de banda

El ancho de banda o *bandwidth* es la máxima capacidad de transferencia de datos a través de un enlace o ruta determinada. La misma se suele medir en bits por segundo o bytes por segundo.

Un concepto altamente relacionado al ancho de banda es la tasa de transferencia efectiva, rendimiento o *throughput*. Es el nombre que se le da a la cantidad de datos que se pueden transferir dentro de un período de tiempo específico, y este será siempre igual o inferior al ancho de banda.

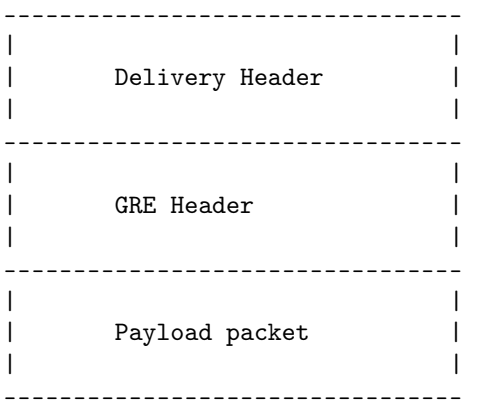
2.1.0.6 Túneles

El túnel (o *tunneling*) en redes informáticas es una técnica que permite que una Unidad de Dato de Protocolo (PDU, *Protocol Data Unit*) pueda ser encapsulada dentro de otra PDU, y esta pueda ser trasladada sin sufrir cambios de un extremo a otro del túnel. Estos extremos están conformados por nodos que suelen ser dispositivos de enrutamiento o computadoras. Un ejemplo de esto es la utilización de túneles SSH, donde a través del protocolo SSH [6] un dispositivo puede inyectar paquetes en otra red prácticamente como si éste estuviera en ella.

Los túneles tienen muchas capacidades; entre ellas permiten transmitir datagramas desde un nodo a otro a través de una red donde la red intermedia no sea capaz de interpretar la información de enrutamiento del mismo, como ocurre con los túneles IPv6 en IPv4, donde se encapsula la PDU (*Protocol Data Unit* o Unidad de Datos de Protocolo) de IPv6 dentro de un PDU con IPv4 para poder acceder a redes IPv6 a pesar de que no toda la red intermedia soporte el protocolo.

Uno de los protocolos más conocidos de túneles es el GRE (*Generic Routing Encapsulation*) que, como su nombre lo indica, es uno de los estándares definidos más genéricos para encapsular un protocolo en otro y de esta forma permitir, por ejemplo, transportar IP sobre IP con diversos propósitos con una estructura similar a la figura 2.1.

Figura 2.1: Estructura de un paquete GRE encapsulado



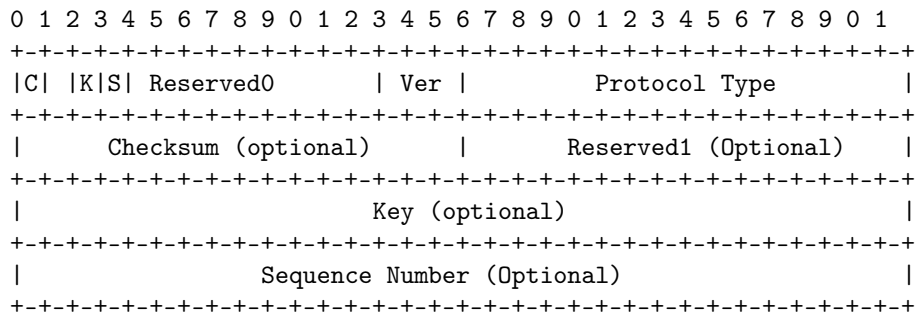
Como se observa en la figura, la cabecera de *delivery*, pudiendo ser por ejemplo IPv4, encapsulará la cabecera GRE [7] especificada en la figura 2.2.

2.1.0.7 BGP

Border Gateway Protocol [8] [9] [10] [11] (o BGP, por sus siglas en inglés) es un protocolo de enrutamiento entre Sistemas Autónomos. Se encuentra basado y dentro del grupo de protocolos EGP (*Exterior Gateway Protocol*). Se utiliza para intercambiar información de enrutamiento entre sistemas autónomos.

Los sistemas autónomos que hablan BGP tienen como función principal intercambiar información de las redes alcanzables a través de otros sistemas BGP. Esta información incluye los números de sistemas

Figura 2.2: Cabecera GRE



autónomos (ASNs, por sus siglas en inglés) que el tráfico debe atravesar para alcanzar estas redes. Al mismo tiempo que un AS informa a sus vecinos las redes alcanzables a través de otros AS, también publica los prefijos de red que posee.

De esta forma se puede construir un grafo de conectividad entre AS en donde se eliminen ciclos para evitar que el tráfico quede en bucle y se cumplan las políticas de tráfico definidas en cada AS. Este camino de ASN que debe atravesar el tráfico para alcanzar su AS destino se denomina ruta (o *path* en inglés) y, generalmente, se crea eligiendo el camino más corto. El tráfico para alcanzar el próximo AS del *path* deberá enrutarse hacia el llamado *next-hop*, o siguiente salto.

BGP se ejecuta sobre TCP, que es un protocolo de nivel de transporte confiable, y utiliza el puerto TCP 179 para establecer sus conexiones e intercambiar la información de rutas. Esta comunicación entre dos sistemas que hablan BGP se la denomina *peering* BGP.

2.1.0.8 FlowSpec [12]

Los *routers* IP modernos tienen la capacidad de retransmitir tráfico, clasificar, dar forma, limitar la velocidad, filtrar o redirigir paquetes en función de políticas definidas administrativamente, es decir, definidas por los administradores de los dispositivos. Estos mecanismos de definición de políticas de tráfico permiten al operador crear reglas de coincidencia que operan en múltiples campos del encabezado del paquete. Cuando el tráfico coincide con alguna de estas reglas, se le aplica las acciones definidas, similar al funcionamiento de un *firewall*.

Las reglas se componen por n-tuplas de criterios que definen especificaciones del flujo de tráfico o *Flow Specification*. Los criterios de selección pueden incluir elementos como son los prefijos de direcciones origen y destino, información del protocolo IP y números de puerto del protocolo de transporte.

Los *Flow Specifications* se distribuyen a través de BGP NLRI (Border Gateway Protocol Network Layer Reachability Information), es decir, se intercambia dicha información a través del puerto TCP 179 sobre la misma conexión establecida para BGP. Esta información también provee las acciones a aplicar sobre los *Flow Specifications* como las reglas de filtros de tráfico, comunidades extendidas o mecanismos de utilización de BGP para filtrado de tráfico dentro y entre proveedores con el fin de mitigar ataques de DoS y DDoS.

Al expandir la información de enrutamiento con *Flow Specifications*, el sistema de enrutamiento puede tomar ventaja de las *Access Control List* (ACL, por sus siglas en inglés), o de las capacidades del *firewall* del router durante la retransmisión.

Flow Specifications hace uso de los siguientes conceptos que pueden ser necesarios durante el desarrollo de la tesina:

- AFI: Address Family Identifier
- Loc-RIB: El Loc-RIB contiene las rutas que han sido seleccionadas por el Proceso de Decisión del sistema BGP local [11].
- NLRI: Network Layer Reachability Information
- PE: Provider Edge router
- RIB: Routing Information Base
- SAFI: Subsequent Address Family Identifier
- VRF: Virtual Routing and Forwarding

2.2 Denegación de servicio

2.2.1 Definición

La denegación de servicio [13] es un tipo de ataque a un sistema de computadoras o red que causa que un servicio o recurso sea inaccesible para los usuarios legítimos. Normalmente provoca la pérdida de la conectividad con la red por el consumo del ancho de banda de la víctima o por la sobrecarga de los recursos computacionales del sistema atacado.

2.2.2 Técnicas

Los ataques de denegación de servicio pueden clasificarse a partir del mecanismo utilizado para lograr su objetivo.

2.2.2.1 Ataques de vulnerabilidades

Este tipo de ataques usualmente se basan en enviar un mensaje o una secuencia de mensajes especialmente diseñados a una aplicación o sistema operativo vulnerable que estén ejecutándose en una máquina objetivo. Estos mensajes tienen como finalidad producir consumos de recursos excesivos en el sistema donde corren o excepciones no atendidas en los procesos y sistemas operativos de la víctima provocando que el proceso termine o sea detenido por el sistema operativo, o hasta que el propio sistema operativo no sea capaz de atender una excepción y genere el famoso *blue screen of death* (BSOD). En algunos casos, los sistemas son capaces de recuperarse de estos problemas, por ejemplo, reiniciando el proceso o el sistema completo. Esto lleva a que el atacante busque reiterar el ataque para que el sistema vuelva a caer y así lograr prevalecer la denegación del servicio. Estos ataques requieren considerablemente menos tráfico en comparación del *flooding* y, por lo tanto, no serán de interés en esta tesina.

2.2.2.2 Ataques de inundación o *flooding*

Los servidores son dispositivos que prestan servicios a través de los procesos que corren en él. Estos procesos hacen uso de los recursos de hardware que el servidor les otorgue. Los recursos de hardware pueden ser comparativamente bastos, pero al final, es evidente que estos en cierto punto son limitados y, por lo tanto, también las capacidades de los procesos que se ejecutan en el servidor. Para un proceso que corre en el servidor, consumir todos los recursos que tiene disponibles le imposibilita la capacidad de, por ejemplo, atender nuevas peticiones y, por ello, generar una denegación del servicio a nuevos clientes potenciales. Existen diversos métodos para realizar esta clase de ataques dependiendo de qué recurso del servidor es el que se busca saturar, pero la mayoría se obtiene a través del *flooding* de datagramas IP, el cual será uno de los tipos de ataques más nombrados en capítulos posteriores.

Existen diversos tipos de ataques de *flooding* y a continuación se detallará el funcionamiento de dos de los más usados que son “ataques de inundación de conexiones” y “ataques de inundación de ancho de banda”.

Ataques de inundación de conexiones

Uno de los protocolos más utilizados de capa de transporte es el protocolo TCP. Este protocolo requiere establecer una conexión a través del saludo de tres vías para empezar a enviar información de la capa de

aplicación.

Con el fin de concretar este tipo de ataque, se establece un gran número de conexiones TCP completamente abiertas o semi-abiertas con el host objetivo sin enviar datos de la capa de aplicación. El host puede llegar a bloquearse con estas conexiones fraudulentas, impidiéndose así que acepte las conexiones legítimas.

Ataques de inundación de ancho de banda

Son aquellos que buscan saturar los recursos del medio de comunicación del dispositivo víctima con su entorno y los usuarios legítimos. Este tipo de ataques suelen sobrecargar los enlaces a partir de mecanismos como el *flooding* (inundación) de paquetes IP.

El *flooding* es capaz de saturar los enlaces de red de menor capacidad que conectan a la víctima generando cuellos de botella capaces de impedir la circulación de otro tipo de tráfico a través de los mismos. A su vez, este tipo de ataques genera sobrecarga en los *buffers* y procesadores de dispositivos de red intermedios como podrían ser *routers* y *switches*, obligando a que no puedan realizar su funcionamiento normal de *buffering* y enrutamiento, y por lo tanto, terminen perdiendo paquetes.

Además, existen técnicas para incrementar la eficacia de inundación de los ataques de ancho de banda al aumentar el volumen de datos transmitidos. Estas técnicas serán detalladas en las próximas secciones.

2.3 Denegación de Servicio Distribuido

2.3.1 Definición

La denegación de servicio distribuido [14], o DDoS por sus siglas en inglés, extiende el concepto de DoS ya que requiere la utilización de un conjunto de varios dispositivos para realizar el ataque.

2.3.2 Técnicas de ataque

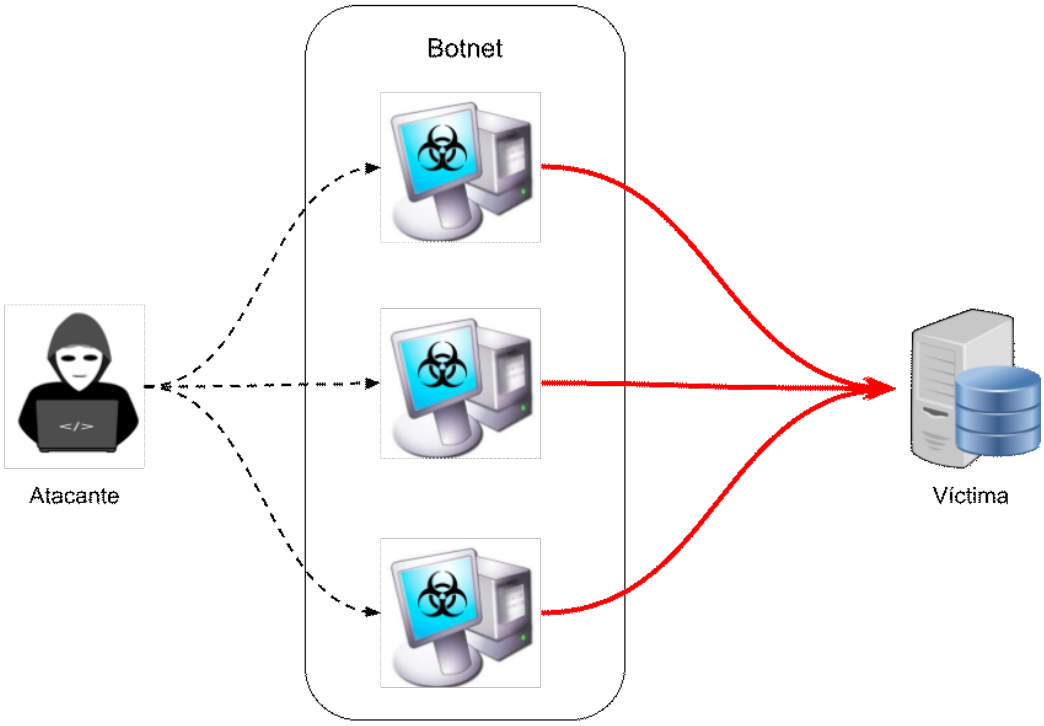
En muchos casos estos dispositivos suelen estar infectados por software malicioso, desconocido por el propietario del dispositivo que participa en dichos ataques. A estos dispositivos infectados se los denomina *bots* y suelen pertenecer a una red denominada *botnet*, coordinada por uno o más dispositivos centrales que definen los objetivos del ataque.

Como ya se mencionó, a través del *flooding* de paquetes los atacantes logran saturar los recursos del sistema objetivo, pero en este caso lo logran efectuando un ataque en simultáneo desde cada uno de los *bots* con un potencial mucho mayor que si fuera sólo uno de ellos. Aquí todos los *bots* se convierten en atacantes del sistema final, logrando con la suma de sus capacidades de *flooding* de paquetes alcanzar un ataque con mayor volumen de tráfico transmitido, por lo que a este tipo de ataques también se les llama **ataques volumétricos**.

Los ataques volumétricos suelen ser fáciles de realizar por una *botnet* ya que es sencillo sumar todos los anchos de banda de cada *bot* y alcanzar un *throughput* suficiente para saturar con tráfico malicioso el enlace de red de un usuario final o una organización pequeña, impidiendo que el tráfico legítimo llegue a destino.

Para un atacante, otra de las ventajas de realizar un ataque a través de una *botnet* es el anonimato del origen del ataque respecto al punto de vista de la víctima. Cuando una *botnet* recibe una orden de ataque a un objetivo desde su controlador central (el atacante original), esta información viaja en una conexión completamente diferente, a veces cifrada, y no vinculada a los datagramas luego utilizados para realizar el ataque desde cada *bot*. Esto le brinda al atacante anonimato desde la vista de la víctima siendo para esta última un ataque desde uno o varios dispositivos irreconocibles. A veces, la cantidad de *bots* llega a un número lo suficientemente grande que es imposible distinguir fehacientemente entre un tráfico legítimo de un ataque, lo cual no da la capacidad de, por ejemplo, identificar uno o varios orígenes y bloquear los mismos. A veces en estos casos no queda más que esperar a que el ataque cese.

Figura 2.3: Diagrama de ataque de tipo DDoS



2.4 Denegación de Servicio Distribuido Amplificado

2.4.1 Definición

Un ataque de Denegación de Servicio Distribuido Amplificado se basa en las técnicas de DDoS y le suma a éstas la capacidad de amplificación basada en servicios UDP abiertos que envían los servicios amplificadores respuestas hacia una víctima, a partir del *spoofing* de direcciones IP origen, con el IP de la víctima, realizados por el atacante y enviados hacia estos amplificadores.

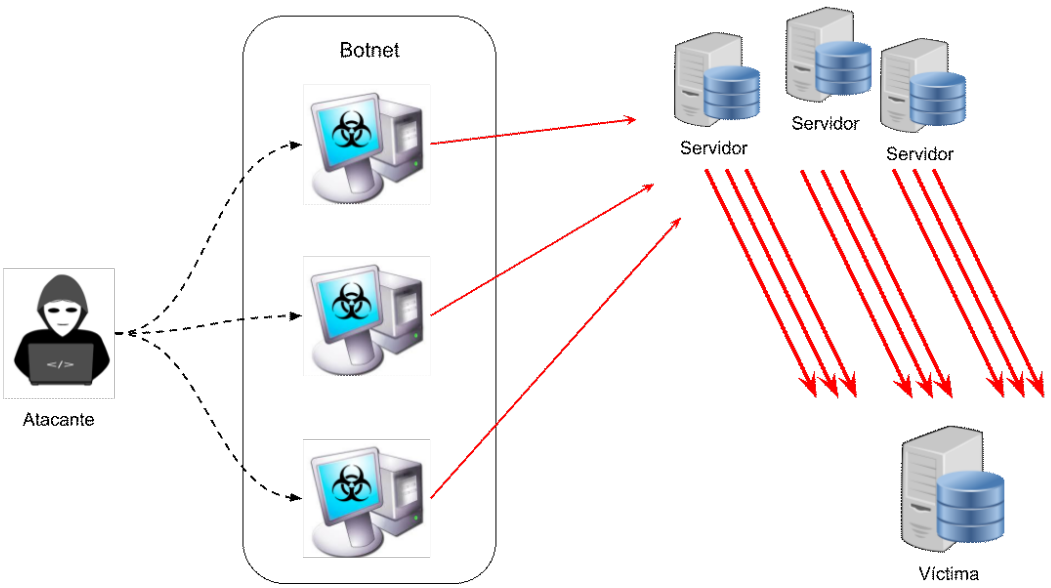
2.4.2 Técnicas de ataques de amplificación

Aquellos servicios que utilicen UDP y produzcan una respuesta ante determinado mensaje, tienen la capacidad de ser reflectantes. Dadas las características de UDP que trabaja sin conexión, el mismo al momento de generar una respuesta lo hará hacia la IP origen del datagrama sin necesidad de realizar comprobación alguna. Esto le permite a un atacante utilizar la técnica de *spoofing* que consiste en modificar la IP origen del datagrama por la de una IP que no sea la propia. De esta manera, la respuesta que dé el destinatario de dicho datagrama será hacia quien posea la IP puesta como origen por el atacante. Esta técnica se conoce también con el nombre de **ataque de reflexión**.

Al realizar este tipo de ataque hacia un protocolo sobre UDP que responda con un datagrama de mayor tamaño al enviado en la consulta, será un ataque amplificado ya que el atacante con un determinado consumo de bits de su ancho de banda producirá un consumo mayor en el ancho de banda de la víctima. A esta técnica se la llama **ataque de amplificación**.

Por consiguiente, un ataque realizado utilizando las técnicas de DDoS hacia servicios en Internet que sean amplificados contra una víctima se le llamará **DDoS Amplificado** y será un ataque potencialmente mucho más efectivo que todos los anteriores vistos.

Figura 2.4: Diagrama de ataque de tipo DoS amplificado



Para analizar la potencia de amplificación de las diferentes técnicas se utiliza la tasa de amplificación de ancho de banda, también llamada BAF (*Bandwidth Amplification Factor*). Existen diversos protocolos UDP que pueden ser amplificadores de ataques con distintos BAF [15] como muestra el cuadro de la tabla 2.1.

Tabla 2.1: Factor de Amplificación de Ancho de Banda por protocolo

Protocolo	**Factor de amplificación**	**Comando vulnerable**
DNS	**28 a 54**	ver: TA13-088A
NTP	**556.9**	ver: TA14-013A
SNMPv2	**6.3**	GetBulk request
NetBIOS	**3.8**	Name resolution
SSDP	**30.8**	SEARCH request
CharGEN	**358.8**	Character generation request
QOTD	**140.3**	Quote request
BitTorrent	**3.8**	File search
Kad	**16.3**	Peer list exchange
Quake Network Protocol	**63.9**	Server info exchange
Steam Protocol	**5.5**	Server info exchange
Multicast DNS (mDNS)	**2 a 10**	Unicast query
RIPv1	**131.24**	Malformed request
Portmap (RPCbind)	**7 a 28**	Malformed request
LDAP	**46 a 55**	Malformed request
CLDAP	**56 a 70**	-
TFTP	**60**	-
Memcached	**10000 a 51000**	-
WS-Discovery	**10 a 500**	-

A fines prácticos, a lo largo de esta tesina se tratará en los ejemplos tanto a los ataques de denegación de servicio como a los ataques de denegación de servicio amplificado por igual, dada su similitud en ciertas características y que ambos son ataques de tipo volumétricos.

2.5 Prevención y mitigación de ataques de denegación de servicio

No existe una solución infalible a los ataques de denegación de servicio. Sin embargo, existen diferentes acciones que se pueden tomar con el objetivo de mitigar estos problemas. La efectividad de tales acciones dependerá de la metodología empleada por el atacante para realizar la denegación de servicio. Además, es importante entender que la implementación de las acciones posibles para combatir estos problemas, no necesariamente las debe implementar la organización víctima de los ataques de denegación.

La mejor aproximación para evitar los ataques de DDoS es la prevención. Para ello, hay una serie de recomendaciones y buenas prácticas que se deberían implementar en todos los actores de Internet. Las recomendaciones son:

- **Aplicar las buenas prácticas** descriptas en BCP38/RFC3704 [16][17] para el filtrado en los ISP de tráfico con direcciones IP falsas (*spoofed*/*forged IP*).
- **Evitar exponer servicios reflexivos y amplificables** a Internet. Este objetivo es muy difícil de llevar a cabo, teniendo en cuenta que estos servicios no están solamente en PCs y servidores que pueden ser actualizados o configurados adecuadamente. Si estos servicios deben estar expuestos a Internet, es necesario mantenerlos actualizados para recibir las correcciones de seguridad pertinentes. Los dispositivos IoT son más difíciles de actualizar y, por ello, son comúnmente utilizados para la implementación de ataques de denegación de servicio [18].

A su vez, la organización *Mutually Agreed Norms for Routing Security* (MANRS) [19], propone:

- **Filtrado**: asegurar la corrección de sus propios anuncios y de los anuncios de los clientes a redes adyacentes con granularidad de prefijo y AS-path.
- **Anti-spoofing**: habilitar la validación de la dirección de origen para al menos redes de clientes auxiliares de alojamiento único, los propios usuarios finales e infraestructura.
- **Coordinación**: mantener información de contacto actualizada y accesible a nivel mundial.
- **Validación global**: publicar los datos para que otros puedan validar la información de enrutamiento a escala global.

Sin embargo, como se ha mencionado, la prevención depende de la aplicación de buenas prácticas por todos los actores de Internet: usuarios e ISPs. La no aplicación de estas buenas prácticas posibilitan la realización de diferentes tipos de ataques de denegación. Es por esta razón que, además de las acciones preventivas, se pueden considerar otros mecanismos y técnicas para mitigar los efectos de un ataque de DoS.

Entre las técnicas de mitigación posibles, se pueden mencionar:

- Aplicación de filtros del ataque en la red de la víctima. Esto, dependiendo del ataque puede incluso resultar insuficiente si el volumen del ataque es superior al ancho de banda del *downstream* que posee la organización.
- Coordinación con el ISP de la víctima para la implementación de los filtros.
- Localización del origen del ataque para el filtrado remoto del mismo. Esto, sumando a la posibilidad de enviar tráfico *spoofed*, hace que sea una tarea muy difícil.
- Implementación de técnicas coordinadas en el enrutamiento de Internet (BGP) para la mitigación de ataques de denegación.

CAPÍTULO 3

ESTADO DEL ARTE

En este capítulo se detallarán diferentes técnicas de prevención y herramientas existentes, algunas privativas y otras de código abierto, que permiten evitar o mitigar los ataques de denegación de servicios distribuidos. Para ello, se utilizará información que se encuentre disponible en Internet o en artículos publicados relacionados.

3.1 Medidas de prevención locales

El Instituto Nacional de Ciberseguridad de España (INCIBE) define que los ataques DDoS son uno de los ciberataques más comunes del mundo [20]. Por esta razón existen algunas técnicas comunes que las organizaciones pueden aplicar como medidas de prevención ante este tipo de ataques de manera local o interna a la organización.

Una de estas técnicas, considerada de las más comunes, consiste en **ubicar el servidor en una red desmilitarizada** (o DMZ). Este tipo de redes se encuentran aisladas de la red interna de la organización que publica el servidor de manera que si un intruso logra obtener acceso al servidor, no pueda ingresar a la red interna provocando mayores problemas. Esta es considerada una *medida de protección interna*. Esta técnica debería ser aplicada a cualquier servicio que sea expuesto a Internet.

Otra técnica que se puede aplicar es el uso de un *firewall*. Este *firewall* debería ser capaz de procesar todo el tráfico de la organización y filtrar, en caso necesario, aquél que se considere malicioso. Esta técnica puede combinarse con el uso de un IDS (un sistema de detección de intrusión) capaces de obtener información sobre las conexiones que se llevan a cabo en la organización y alertar en caso de que un ataque estuviera ocurriendo.

Dependiendo de la ubicación del *firewall*, se podrá llegar a ser más o menos efectivo. Si el *firewall* se coloca cerca del *router* de borde de la organización, el tráfico malicioso se descartaría apenas ingresara a la organización y evitaría que el resto de los dispositivos de red internos procesen dicho tráfico. Caso contrario, si el *firewall* se aplicase en el servidor del servicio afectado, todo el tráfico habría sido procesado por todos los dispositivos intermediarios internos de la organización, provocando el uso de recursos para

procesarlo.

Es por ello que contar con un mayor ancho de banda puede suponer una buena estrategia para aplicar, tanto si el servidor se encuentra alojado en la organización o en la nube (*hosting* proporcionado por un tercero). Esto implicaría mejorar los dispositivos con los que ya se cuenten o agregar nuevos, siempre y cuando esto sea posible llevar adelante, y de esta forma, lograr procesar una mayor cantidad de tráfico y evitar, aunque generalmente sea temporal, la sobrecarga de los dispositivos de red.

Un problema que acarrea este tipo de soluciones es que la cantidad de tráfico malicioso durante un ataque puede ser tan grande que los dispositivos que deben filtrarlo, en este caso el *firewall*, pueden verse sobrecargados hasta dejar de funcionar.

Además, este tipo de soluciones suelen ser insuficientes si el tipo de ataque de denegación de servicios es volumétrico y superior al ancho de banda del *downstream* de la víctima debido a la saturación del enlace de su conexión a Internet, provocando que el ataque sea exitoso.

Por esta razón, se puede contar con redundancia y balanceo de cargas. La redundancia implica tener el servicio duplicado en más de un servidor en diferentes locaciones y el balanceo de cargas permite que se asigne la petición del usuario a uno u otro servidor en función de la carga de trabajo que estén llevando a cabo. Al contar con más de un servidor para un mismo servicio, se reduce bastante la probabilidad de que se sufra un ataque de denegación de servicio, ya que si un servidor se sobrecarga, el o los otros servidores podrían seguir en funcionamiento. Otra ventaja de esta técnica es la tolerancia a fallos, ya que si un servidor queda fuera de servicio, los otros servidores se encargarían de continuar con el trabajo.

Hay muchos sistemas en Internet que no se encuentran correctamente actualizados como se desearía y, por ende, pueden ser vulnerables. Es por ello que este tipo de sistemas son objetivos de ataques de diversas índole. Por esta razón, es importante actualizar el sistema siempre que sea posible y, de esta manera, evitar estar bajo la lupa de los atacantes.

Estas son algunas soluciones que pueden aplicar los administradores de la organización para mitigar un ataque de denegación de servicio en su red interna.

Aunque estas soluciones pueden llegar a detener los ataques de denegación de servicio filtrando tráfico malicioso, la sobrecarga de los recursos físicos sigue siendo el mayor problema para la organización, incluso si se agregan o mejoran los que ya existen. Este problema podría provocar que servicios que no sean los objetivos principales del ataque se vean comprometidos también debido a la saturación de los equipos de red de la organización.

No existe certeza de que la red de la organización sea capaz de gestionar el flujo de tráfico durante un ataque de denegación de servicio. Y tampoco las técnicas vistas previamente aseguran que ello ocurra.

Y, aunque la organización sea capaz de soportar el tráfico malicioso, aún debe procesarlo y esto es uso de recursos que se traduce en consumo de energía, desgaste de equipos y poder de cómputo que podría utilizarse para el tráfico legítimo y no malicioso.

Por esta razón es necesario que cuando se esté ejecutando un ataque de denegación de servicio, la víctima del mismo pueda recurrir a otro tipo de técnicas de prevención o mitigación y evitar que el tráfico provoque mayores problemas a la red de su organización.

3.2 Agujero negro (*blackhole routing*) y RTBH (*Remotely Triggered Black Hole*)

La técnica de *blackhole routing* [21] implica redirigir el tráfico hacia un agujero negro [22], es decir, a una ruta donde el tráfico es descartado. Esta es una contramedida que suele utilizarse cuando ocurre un ataque de denegación de servicio. Generalmente, todo el tráfico dirigido a la IP del host o red víctima, tanto legítimo como malicioso, es enviado a una ruta llamada **nula** (*null route*) donde es descartado de la red.

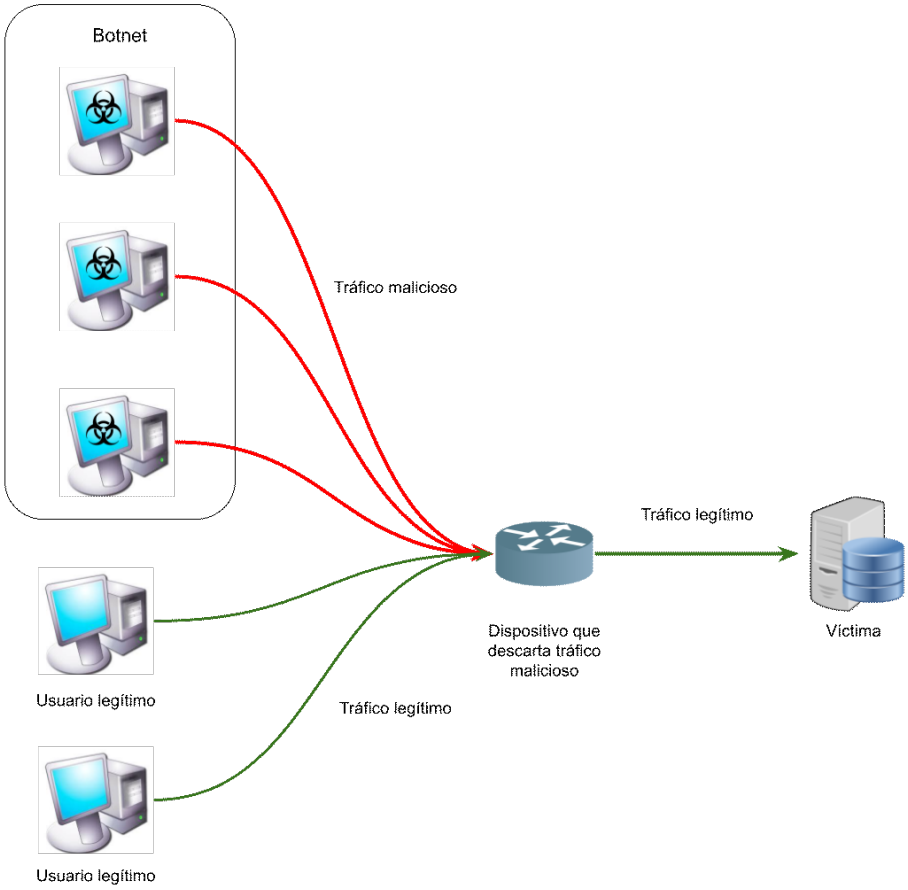
Son los proveedores de servicio de Internet los encargados de implementar estas rutas y de descartar el tráfico hacia el host víctima antes de que sea dirigido hacia la organización afectada. De esta manera, la organización ya no tendría que procesar todo el tráfico malicioso o filtrarlo. Este servicio suele proveerse como un servicio adicional por parte de los proveedores de servicio de Internet hacia sus clientes.

Para que esta técnica sea aplicada, es necesario realizar configuraciones específicas en el protocolo BGP y coordinar aspectos técnicos con el ISP que brinda conectividad a Internet. Por esta razón es importante contar con la administración de la red de la organización para ejecutar los comandos necesarios para su publicación o que el proveedor de servicios pueda hacerlo a pedido de sus clientes.

Con la técnica de RTBH, un tipo de implementación de *blackhole routing*, se puede lograr la mitigación del ataque y se evita la congestión de la red completa de la organización, pero el servicio objetivo de los atacantes queda fuera de línea dado que todo el tráfico, tanto malicioso como legítimo, es enviado al agujero negro o ruta nula. Suponiendo que el servidor atacado posee un servidor HTTP y un servidor de Mail y el servicio de correo es el que está siendo atacado, este tipo de técnicas no evita dejar fuera de línea a todos los servicios corriendo en el host. A pesar de ello, este tipo de soluciones son de mucha utilidad para evitar problemas mayores ya que de esta forma se evita que el tráfico malicioso llegue a la red destino y los enlaces sean saturados por tráfico malicioso.

Dado que el tráfico legítimo se encontraría afectado, se deberá buscar otro tipo de solución a los ataques DDoS que elimine el tráfico malicioso pero permita que el tráfico legítimo pueda llegar a su destino (ver figura 3.1).

Figura 3.1: Diagrama filtrado de tráfico malicioso



3.3 Unwanted Traffic Removal (UTRS)

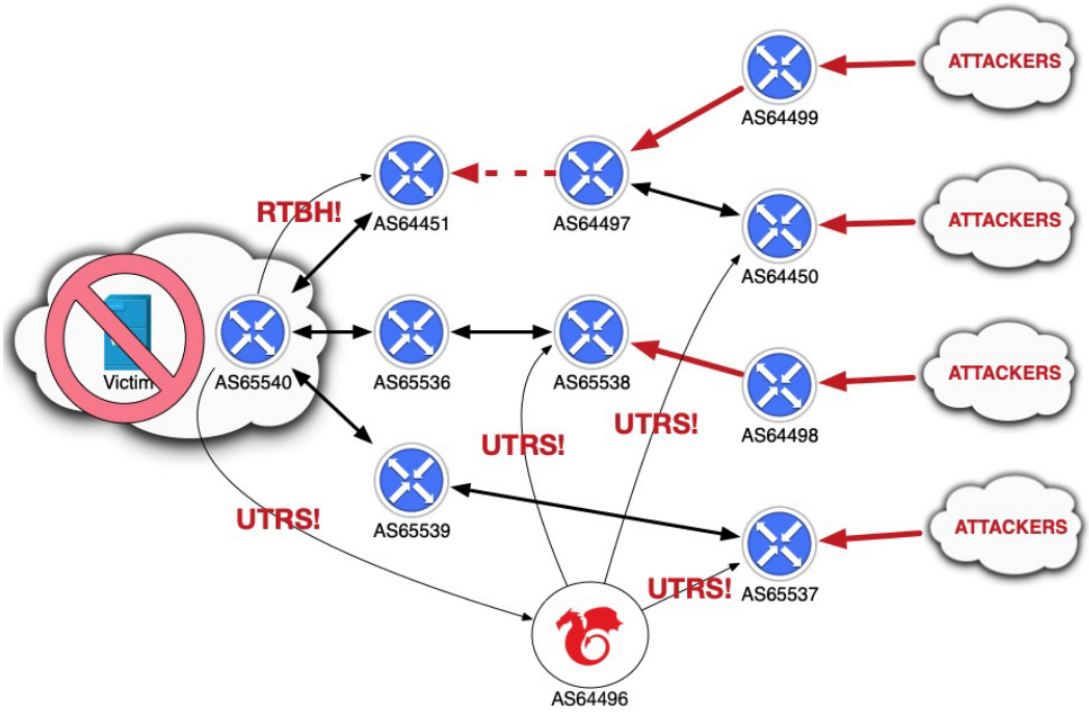
Team Cymru es un equipo de profesionales que trabajan en conjunto con equipos de seguridad (CSIRTs) alrededor del mundo, investigadores, sector de la educación, proveedores de servicio de Internet (ISP), entre muchos otros [23] con el objetivo de rastrear y evitar que atacantes o infraestructuras maliciosas lleven a cabo sus actividades. Este equipo ha llevado a cabo el desarrollo de diferentes tipos de herramientas y servicios, algunas sin costo de índole comunitarios y otras soluciones comerciales, para lograr dicho objetivo. Entre estas herramientas, se encuentra Unwanted Traffic Removal o UTRS.

Esta herramienta permite reducir el impacto de los ataques DDoS a nivel global. Los usuarios (organizaciones o ISP), deben completar un formulario donde, entre otra información, se les solicitará el ASN y una dirección IP para establecer un *peering* BGP, necesario para el funcionamiento de esta herramienta [24].

La manera en la que UTRS mitiga los ataques de denegación de servicio es bloqueando el tráfico malicioso a partir de los anuncios enviados por los colaboradores mediante el uso del protocolo BGP. De esta manera, los ISP, proveedores de *hosting* e instituciones educativas, entre otras organizaciones, pueden automáticamente aplicar las reglas de filtro de tráfico basadas en BGP recibidas.

Cuando un sistema autónomo se encuentra bajo ataque, anuncia su IP afectada a través de una sesión BGP. El resto de los participantes crearán una ruta que redirigirá todo el tráfico hacia un agujero (RTBH) que tenga como destino el bloque anunciado. Esto provocará que el tráfico malicioso pueda ser filtrado lo más cerca del origen posible y, de esta forma, reducir la cantidad de tráfico malicioso a través de Internet [25] (ver figura 3.2). Por ello es importante que haya la mayor cantidad de participantes posibles para lograr una mayor efectividad.

Figura 3.2: Esquema de funcionamiento de UTRS de Team Cymru [26]



Cabe destacar que UTRS actualmente (al momento de escribir este texto) solo soporta IPv4, pero se pretende que en el futuro sea capaz de soportar IPv6.

Como en la técnica anteriormente detallada, utilizando este mecanismo, todo el tráfico dirigido al servicio afectado es descartado, sea legítimo o malicioso. Existe una técnica que permitirá descartar el tráfico malicioso y, también, permitirá que el tráfico legítimo llegue al servicio afectado.

3.4 Scrubbing center

La técnica de *scrubbing center* [27] [28] permite que el tráfico dirigido a la organización pueda ser procesado y "limpiado" antes de ser reenviado a su destino. De esta manera, el tráfico malicioso es descartado inmediatamente y el legítimo puede continuar hacia el servicio afectado. Por lo tanto, a diferencia de otras técnicas, con el *scrubbing center* se puede mitigar un ataque de denegación de servicio y el servicio afectado puede continuar operativo.

Para que esto pueda llevarse a cabo, es necesario contar con dispositivos de red (*routers*) que sean capaces de comunicarse mediante el protocolo BGP alrededor del mundo estableciendo *peerings* BGP con

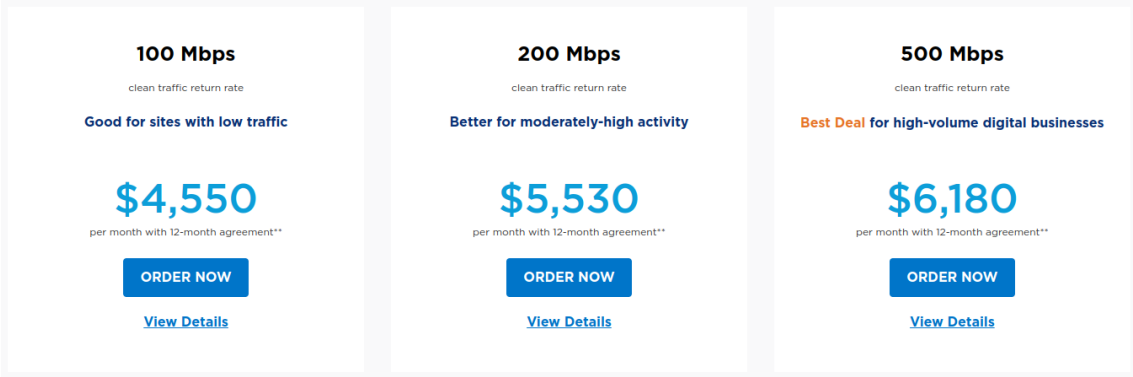
sus vecinos. Estos dispositivos son los encargados de recibir todo el tráfico dirigido hacia la organización utilizando como técnica la publicación del prefijo IP afectado del Sistema Autónomo víctima.

Una vez recibido este tráfico, el dispositivo deberá limpiarlo, removiendo el tráfico malicioso utilizando alguna técnica de filtrado. Como opción, se podría utilizar un *firewall* administrado de manera remota por el administrador del *scrubbing center* o por el mismo cliente.

Por último, se deberá reenviar el tráfico legítimo hacia su destino mediante una comunicación directa, por ejemplo mediante el uso de técnicas de *tunneling*, privado o público, a través de Internet. Esto se debe a que si el tráfico fuera enrutado directamente hacia Internet desde el *scrubbing center* por BGP, regresaría al mismo provocándose un bucle, ya que se encuentra anunciando el prefijo de red afectado.

Las organizaciones que brindan este tipo de solución en forma comercial cuentan con recursos suficientes para procesar una gran cantidad de volúmenes de tráfico. Los clientes de este tipo de servicios son los proveedores de Internet y las grandes empresas dado que tienen un costo elevado (ver figura 3.3). Además, en muchos casos, estos servicios no se contratan porque las organizaciones consideran que no lo necesitan ya que no sufrieron ataques de DoS en el pasado.

Figura 3.3: Tabla de precios mensuales en dólares del servicio de mitigación DDoS de la empresa Lumen por tráfico limpio entregado a la organización objetivo [29]



Se considera que esta es la solución definitiva a un ataque DDoS, junto con las técnicas mencionadas a lo largo de este apartado.

3.5 Algunas herramientas o servicios sobre mitigación de ataques DDoS

3.5.1 nScrub

nScrub [30] es una herramienta desarrollada con el objetivo de mitigar los ataques DDoS [31]. Se basa en **PF_RING y Zero Copy** [32] permitiéndole ser capaz de operar en sistemas *low-end* (*hardware* que no necesariamente debe ser dedicado y que suelen tener un alto costo económico) a velocidades de 10 Gigabits/s. nScrub ha sido diseñado como una plataforma extensible, es decir, cuenta con la posibilidad de definir algoritmos o funcionalidades adicionales para la mitigación de ataques incrementando sus posibilidades.

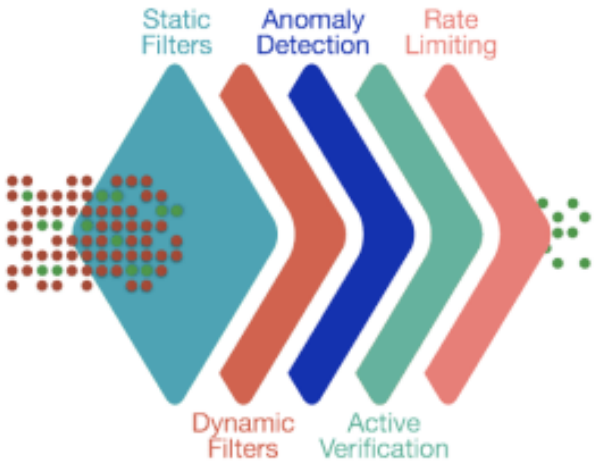
Además, provee de una API REST para configurarlo y combinarlo con una herramienta CLI con autocompletado.

Algunas de las funcionalidades más importantes con las que cuenta **nScrub** son:

Aplicación en tráfico multicapas (figura 3.4):

- Verificación de sesiones para protocolos como TCP y DNS.
- Listas blancas y negras para subredes.
- Verificación de DNS (prevención de ataques mediante técnicas que utilizan este protocolo).
- Listas de control de acceso (ACLs) con políticas basadas en campos UDP/TCP o ICMP.
- Filtrado basado en contenido y filtrado mediante requerimientos HTTP.
- Detección de anomalías basadas en el comportamiento del tráfico de la red.
- Limitación del tráfico, los paquetes que estén por debajo del valor máximo configurado se reenvían, en caso contrario, se descartan. Esto garantiza que el tráfico no deseado tendrá un flujo máximo limitado. Posibilidad de modificar estos límites basados en protocolo y origen o destino.

Figura 3.4: Diagrama multicapa de nScrub



Aplicación multi-tenancy (figura 3.5):

- El tráfico entrante se divide en varios mitigadores que se basan en la dirección IP de destino, de esta manera es posible especificar políticas de tráfico por subred de destino.
- Cada mitigador virtual se asocia a perfiles de tráfico (*default, white, black, gray*). Cada perfil contiene configuración de tráfico y se aplica a las direcciones IP de acuerdo a las listas definidas en los perfiles.

Aplicación en modo de puente transparente (*Transparent Bridge Mode*):

Figura 3.5: Diagrama multitenancy de nScrub

- Si se lo ejecuta en este modo (ver figura 3.6) no requiere de ningún tipo de configuración, de este modo se coloca entre el tráfico de Internet y el destino en el borde de la organización.

Figura 3.6: nScrub en modo puente transparente

Aplicación en modo de enrutamiento (*Routing Mode*):

- En este modo (ver figura 3.7), es posible mitigar los ataques según la necesidad del usuario y en ubicaciones remotas. Esta es la forma similar en la que funciona un *scrubbing center* y la única que implica su instalación fuera de la red de la organización que lo implementa.

Figura 3.7: nScrub en modo de enrutamiento

Esta herramienta cuenta con características muy interesantes, sin embargo, su instalación, la cual depende del usuario, y puesta a punto puede verse dificultada si no se cuenta con experiencia previa ya que no es simple de llevar a cabo.

Aunque existen en el mercado algunos dispositivos de *hardware* que ya cuentan con la solución instalada y configurada lista para utilizarse, estos equipos pueden llegar a valer alrededor de 12000 dólares en Reino Unido [33]. Además, en ese mismo enlace se detallan las especificaciones del hardware. Esta información es útil para tener un punto de comparación sobre los precios del mercado de este tipo de productos y soluciones.

3.5.2 Telecom

Telecom ofrece un servicio de mitigación de ataques DDoS para empresas [34] y es, además, una de las pocas compañías que otorga una breve descripción de los servicios que brinda. Como se ha mencionado antes, este tipo de servicios son prácticamente cajas negras sobre las cuales no se sabe cómo funcionan ni cómo están implementados, solamente se sabe lo que hacen.

En este caso, se puede leer en su sitio web para registrarse que proporciona un monitoreo permanente y continuo del tráfico de una empresa y así detectar de manera inmediata ataques de denegación de servicio distribuido.

Prometen contar con la infraestructura necesaria para mitigar los ataques mediante filtrados inteligentes que les permite diferenciar el tráfico malicioso del tráfico legítimo. Con ello, solamente filtrarían el tráfico malicioso y redirigirían el tráfico legítimo al sitio del cliente.

Esta es una de las características que ScrubbingUNLP implementa y, al ser *opensource*, cualquiera puede utilizarlo y ver su código fuente.

3.5.3 Cloudflare

Por su parte **Cloudflare** ofrece también sus servicios pero no detallan ni resumen ningún tipo de información sobre cómo funciona o se debe implementar. Sin embargo, cuenta con un formulario que debe completar el cliente para obtener el servicio [35].

3.5.4 Imperva

Imperva ofrece diferentes soluciones a distintos tipos de ataques DDoS. Para los ataques de red utiliza un sistema similar a un scrubbing center, donde el tráfico dirigido a la organización es recibido por Imperva, el mismo es "limpiado" y reenviado a la organización víctima a través de un túnel GRE [36].

3.5.5 Lumen

Lumen también ofrece sistemas de mitigación de DDoS utilizando como herramienta los *scrubbing centers*, que reciben el tráfico, lo limpian y reenvían el mismo a su destino original a través de túneles [29].

CAPÍTULO 4

SCRUBBINGUNLP

Esta sección está dedicada al objetivo de la tesina, la plataforma desarrollada de mitigación de ataques DDoS en la nube, la cual los autores denominan ScrubbingUNLP. Para organizarla se ha dividido en diferentes secciones.

La primer parte está destinada al diseño de la plataforma, los sistemas y al trabajo de investigación realizado. Allí se hará referencia a las herramientas específicas que se utilizan para su desarrollo.

En la segunda parte se verá el desarrollo de la herramienta en sí, desde las dependencias utilizadas, el modo de instalación, funcionamiento, ataques disponibles y el modo de uso con ejemplos.

4.1 Objetivos y funcionamiento

Como ya se ha mencionado, al igual que otros *scrubbing centers*, el ScrubbingUNLP tiene como objetivo principal limpiar el tráfico que llega a la organización destino removiendo la parte que constituya un ataque y reenviar el que sea legítimo a su destino. Para esto será necesario en primera instancia que el usuario, administrador de red de una organización partícipe del sistema, identifique ese tráfico que constituye parte del ataque. Por ejemplo, a partir de la red origen, los puertos destino del servidor atacado y la IP del servidor o los servidores destino bajo su control.

Una vez identificado el flujo de datos que constituyen el ataque, el usuario podrá definir a través de un sitio web de la plataforma los parámetros necesarios para que el tráfico sea filtrado por ScrubbingUNLP. Para ello, primero deberá ingresar con su usuario y contraseña provistos por los administradores al sitio web donde tendrá visión de lo que ocurre en los *scrubbing centers*, podrá observar los volúmenes de tráfico que transitan por ellos y las limpiezas que estos realizan en tiempo real si es que se encuentran activos.

El usuario, a su vez, tendrá control para realizar ciertas acciones relacionadas a publicaciones para su propio ASN y bloques de red relacionados. Así, el usuario como primera instancia para filtrar el ataque deberá, a través del sitio web de ScrubbingUNLP, realizar una publicación del bloque de red perteneciente a su AS que se encuentra bajo ataque. Esto efectuará una publicación BGP en los nodos de limpieza pertenecientes a la plataforma y, de esta manera, atraerá todo el tráfico cercano en Internet dirigido a ese

bloque.

Cuando el tráfico ha sido recibido por el nodo de limpieza, este necesitará reenviarlo mediante túneles GRE hacia la red destino. Para ello, los nodos de limpieza crean túneles GRE automáticamente hacia un host definido previamente por el administrador de red de la organización y asignado en la interfaz web por un administrador del ScrubbingUNLP. Por otro lado, el otro extremo del túnel será configurado por el administrador de red de la organización para recibir el tráfico y procesarlo hacia su destino.

Hasta el momento, se ha logrado redirigir el tráfico que circula por Internet de modo tal que atraviese algún nodo cercano del ScrubbingUNLP debido a las publicaciones de rutas BGP creadas desde la interfaz web. En este punto aún no se ha aplicado ninguna clase de limpieza, por lo que no se verán cambios para clientes que interactúen con los servicios ofrecidos en la organización.

Una vez existente la publicación del prefijo IP desde el ScrubbingUNLP el usuario podrá, a través de la interfaz web, realizar acciones de limpieza sobre el respectivo bloque de red publicado. Estas acciones dependerán del tipo de ataque identificado por el usuario antes mencionado y las mismas se propagarán a todos los nodos del ScrubbingUNLP a través del protocolo FlowSpec, que a grandes rasgos, es capaz de definir reglas similares a las de un *firewall* de capa de transporte, que serán interpretadas por los nodos y los mismos aplicarán en sus respectivos *firewalls* las reglas, por ejemplo, de denegación para cierto tipo de tráfico, provocando que el mismo no se dirija por el túnel GRE previamente creado y sea selectivamente descartado. De este modo, sólo el tráfico que el administrador de la red desee llegará a través del túnel GRE a su organización.

4.2 Herramientas

4.2.1 Criterios de selección de herramientas

La selección de herramientas tiene como principales criterios la licencia con la que se distribuye el software, la facilidad de uso basada en la experiencia de los autores de esta tesina y la comunidad que lo utiliza.

4.2.1.1 La licencia

Se ha optado por utilizar herramientas de diversas licencias [37] mientras las mismas permitan siempre la redistribución y uso. Contemplando que este será un software abierto y de uso libre se dará, cuando corresponda según la licencia, crédito a los creadores de las herramientas. El uso de las herramientas contempla un posible cambio de licencia y restricciones de uso de la misma, por lo que, gracias a un desarrollo modular, cualquiera de las herramientas podrá ser fácilmente reemplazada por otra o un eventual desarrollo propio.

4.2.1.2 La experiencia

Se ha considerado bajo la experiencia de los autores en el área de redes, infraestructura y seguridad informática, apropiado utilizar las herramientas seleccionadas porque han sido altamente fiables para los propósitos buscados o similares. También ellas sitúan a los autores en la tranquilidad ya que ofrecen la más amplia capacidad y variedad de utilidades frente a otras existentes. Esto da una ventaja frente a una futura necesidad de requerir nuevas capacidades que otras no posean y deban adaptar.

4.2.1.3 La comunidad que lo soporta

Es de gran importancia la existencia de una gran comunidad de usuarios y, por igual, una gran comunidad de desarrolladores que aporten a la continuidad y avances de los productos. Los usuarios dan una pauta del buen desempeño de las herramientas, su facilidad de uso y correcto funcionamiento, por ejemplo, cuando se habla de protocolos. A su vez la existencia de múltiples usuarios demanda una mejor documentación debido a la mayor cantidad de consultas, respuestas y finalmente se logra aportar ideas, documentación, *testing*, problemas y soluciones a estos problemas. Los desarrolladores son un indicador del soporte que tiene el sistema en la actualidad. De este modo, cuanto más desarrolladores hayan aportando a la herramienta, más probabilidad habrá de conseguir una solución a un posible problema que aparezca, o menos posible que el software no se actualice ante los avances de sus dependencias y sea descontinuado.

4.2.2 Productos

Los productos seleccionados para el desarrollo fueron elegidos en base a los motivos ya mencionados y a su compatibilidad entre ellos. También es importante considerar que toda la plataforma funciona sobre máquinas virtuales y no hace uso de hardware específico, por eso mismo este sistema puede ser utilizado dentro de un emulador de redes con sistemas operativos basados en Debian. A continuación se darán unas breves definiciones oficiales de las herramientas que se utilizarán en la próxima sección.

4.2.2.1 Debian Linux

Hay muchas razones para elegir Debian Linux [38] como sistema operativo: como usuario, como desarrollador e incluso en entornos empresariales. La mayoría de los usuarios aprecian la estabilidad y los procesos de actualización fluidos de ambos paquetes y de toda la distribución. Debian también es ampliamente utilizado por desarrolladores de software y hardware porque se ejecuta en numerosas arquitecturas y dispositivos, ofrece un rastreador de errores público y otras herramientas para desarrolladores. Las versiones LTS e imágenes en la nube que poseen son ideales para trabajar en entornos profesionales.

4.2.2.2 Bash

Bash es la *shell* del Proyecto GNU [39], el Bourne Again SHell. Esta es una *shell* compatible con *sh* que incorpora características útiles del *shell Korn (ksh)* y el *shell C (csh)*. Está diseñado para cumplir con el estándar IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools. Ofrece mejoras funcionales sobre *sh* tanto para programación como para uso interactivo. Además, Bash puede ejecutar la mayoría de los *scripts sh* sin modificaciones.

Las mejoras que ofrece Bash incluyen:

- Edición de línea de comandos.
- Historial de comandos de tamaño ilimitado.
- Control de trabajo.
- Funciones de shell y alias.

- Matrices indexadas de tamaño ilimitado.
- Aritmética entera en cualquier base de dos a sesenta y cuatro.

4.2.2.3 Python

Python es un lenguaje de programación interpretado [40], interactivo y orientado a objetos. Incorpora módulos, excepciones, tipos dinámicos, tipos de datos dinámicos de muy alto nivel y clases. Admite múltiples paradigmas de programación más allá de la programación orientada a objetos, como la programación procedimental y funcional. Python combina una potencia notable con una sintaxis muy clara. Tiene interfaces para muchas bibliotecas y llamadas al sistema, así como para varios sistemas de ventanas, y es extensible en C o C++. También se puede utilizar como lenguaje de extensión para aplicaciones que necesitan una interfaz programable. Finalmente, Python es portátil: se ejecuta en muchas variantes de Unix, incluidos Linux y macOS, y en Windows.

4.2.2.4 Django

Django es un *framework* web Python [41] de alto nivel que fomenta un desarrollo rápido y un diseño limpio y pragmático. Creado por desarrolladores experimentados, se ocupa de gran parte de las molestias del desarrollo web, por lo que puede concentrarse en escribir su aplicación sin necesidad de reinventar la rueda. Es gratis y de código abierto.

4.2.2.5 ExaBGP

ExaBGP [42] proporciona una forma conveniente de implementar SDN (Software Defined Networks o Redes Definidas por Software) mediante la transformación de mensajes BGP (Border Gateway Protocol) en texto plano o JSON, que luego se pueden manejar fácilmente mediante *scripts* simples o un BSS/OSS. Se utiliza habitualmente para mejorar la resiliencia del servicio y brindar protección contra fallas en la red o en el servicio. Por ejemplo, gracias al *backend* de estabilidad incluido, las fallas del servicio DNS *anycast* se pueden detectar y reducir las fallas.

Además, solo o junto con FastNetMon o WanGuard, proporciona a los operadores de red una solución de protección DDoS rentable.

Gracias al balanceo de flujo de los *routers* modernos, ExaBGP también se puede usar para ahorrar dinero en balanceadores de carga.

Otros usos incluyen vigilar los cambios de red realizados como lo hizo RIPE u otras redes con GIXLG.

4.2.2.6 iptables

iptables es una utilidad de línea de comando para configurar el *firewall* [43] del kernel de Linux implementado como parte del proyecto Netfilter. El término iptables también se usa comúnmente para referirse a dicho *firewall* del kernel. Puede configurarse directamente con iptables, o usando uno de los muchos *frontend* existentes de consola y gráficos. El término iptables se usa para IPv4, y el término ip6tables para IPv6. Tanto iptables como ip6tables tienen la misma sintaxis, pero algunas opciones son específicas de IPv4 o de IPv6.

4.2.3 fprobe

fprobe [44] es una herramienta basada en libpcap [45] que recopila datos de tráfico de red y lo convierte en flujos de NetFlow [46] enviándolo hacia un determinado colector de este tipo de datos para procesarlos.

NetFlow es un protocolo de red perteneciente a Cisco que es utilizado para recolectar información sobre tráfico IP. Además de los dispositivos de Cisco, existen otros dispositivos con sus respectivos sistemas operativos capaces de generar flujos NetFlow, entre ellos Linux, usando la herramienta fprobe, mencionada previamente.

4.2.4 Elasticsearch

Elasticsearch es un motor de analítica y análisis distribuido [47] para todos los tipos de datos, incluidos textuales, numéricos, geoespaciales, estructurados y no estructurados. Elasticsearch está desarrollado a partir de Apache Lucene. Conocido por sus API REST simples, naturaleza distribuida, velocidad y escalabilidad, Elasticsearch es el componente principal del Elastic Stack, un conjunto de herramientas gratuitas y abiertas para la ingesta, el enriquecimiento, el almacenamiento, el análisis y la visualización de datos.

4.2.5 Kibana

Kibana es una interfaz de usuario abierta que permite la visualización de datos de Elasticsearch mediante gráficos de diferentes índoles [48]. Esta herramienta brinda, entre otras, histogramas en tiempo real y mapas. También es capaz de generar visualizaciones en mapas a partir de datos geoespaciales o de direcciones IP.

4.2.6 Filebeat

Filebeat es un agente de recolección de datos en servidores [49]. Es capaz de monitorizar los registros o las ubicaciones del sistema que el usuario defina y reenviar los datos obtenidos a servidores Elasticsearch. Contiene una gran diversidad de módulos que permiten la integración con una gran cantidad de herramientas, entre ellas NetFlow. En este caso, es capaz de permanecer escuchando en un puerto UDP definido por el usuario y recibir y procesar flujo de NetFlow y reenviarlos a un servidor con Elasticsearch.

4.3 Desarrollo

4.3.1 Componentes y arquitectura

La plataforma está compuesta de tres componentes: la **interfaz** llamada WebScrub que se provee al usuario, el **scrubbing center** que limpia el tráfico y el **túnel** donde sale el tráfico limpio. Las componentes pueden estar ubicadas en máquinas diferentes. Entre estas tres componentes, la plataforma se encargará por medio del **scrubbing center** de limpiar el tráfico pedido por el usuario a través de la **interfaz** y liberarlo por el extremo del **túnel** ubicado en un dispositivo dentro de su organización.

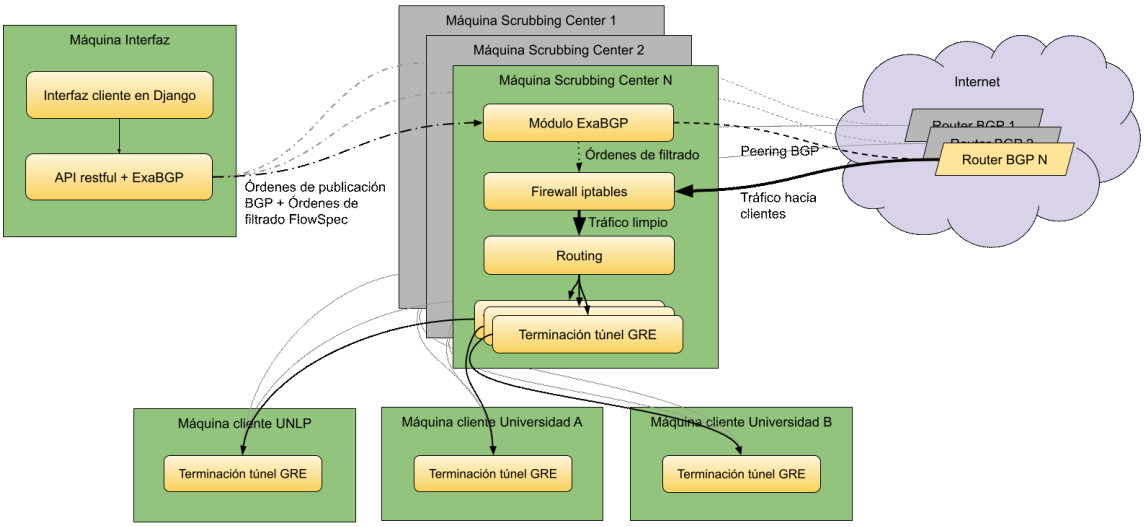
Como se muestra en la figura 4.1, la plataforma debe contar con una máquina donde se instale y se sirva la interfaz web disponible al usuario. Este sitio, en su *backend*, se comunica con una *API restful* que recibe órdenes a enviar a través de ExaBGP con mensajes BGP hacia el otro lado del *peering* BGP donde

están las máquinas del *scrubbing center* ubicadas en un lugar distante, siendo deseable, lo más cercano a un posible ataque.

Como se mencionó, estas máquinas del *scrubbing center* poseen el otro par de la conexión BGP, atendida por una instancia del módulo ExaBGP. Este módulo es capaz de tomar acciones BGP como la publicación de un bloque de red a través del *peering* BGP con un *router* ubicado en Internet, o también, de correr comandos dentro de la misma máquina donde se ejecuta. La publicación del bloque de red provocará que el tráfico destinado a esa red (perteneciente a uno de los clientes) sea encaminado a través del *scrubbing center*. Y los comandos ejecutables en la máquina serán principalmente acciones al *firewall* para aplicar un determinado filtro de red a ese tráfico que atraviesa el *scrubbing center*.

Finalmente, ese tráfico es direccionado a través de las rutas del *scrubbing center* a un túnel específico que conecta al mismo con un dispositivo que soporta túneles GRE, situado dentro de la organización del cliente. Por ahí es donde saldrá el tráfico limpio y sin que el ataque alcance a su destino.

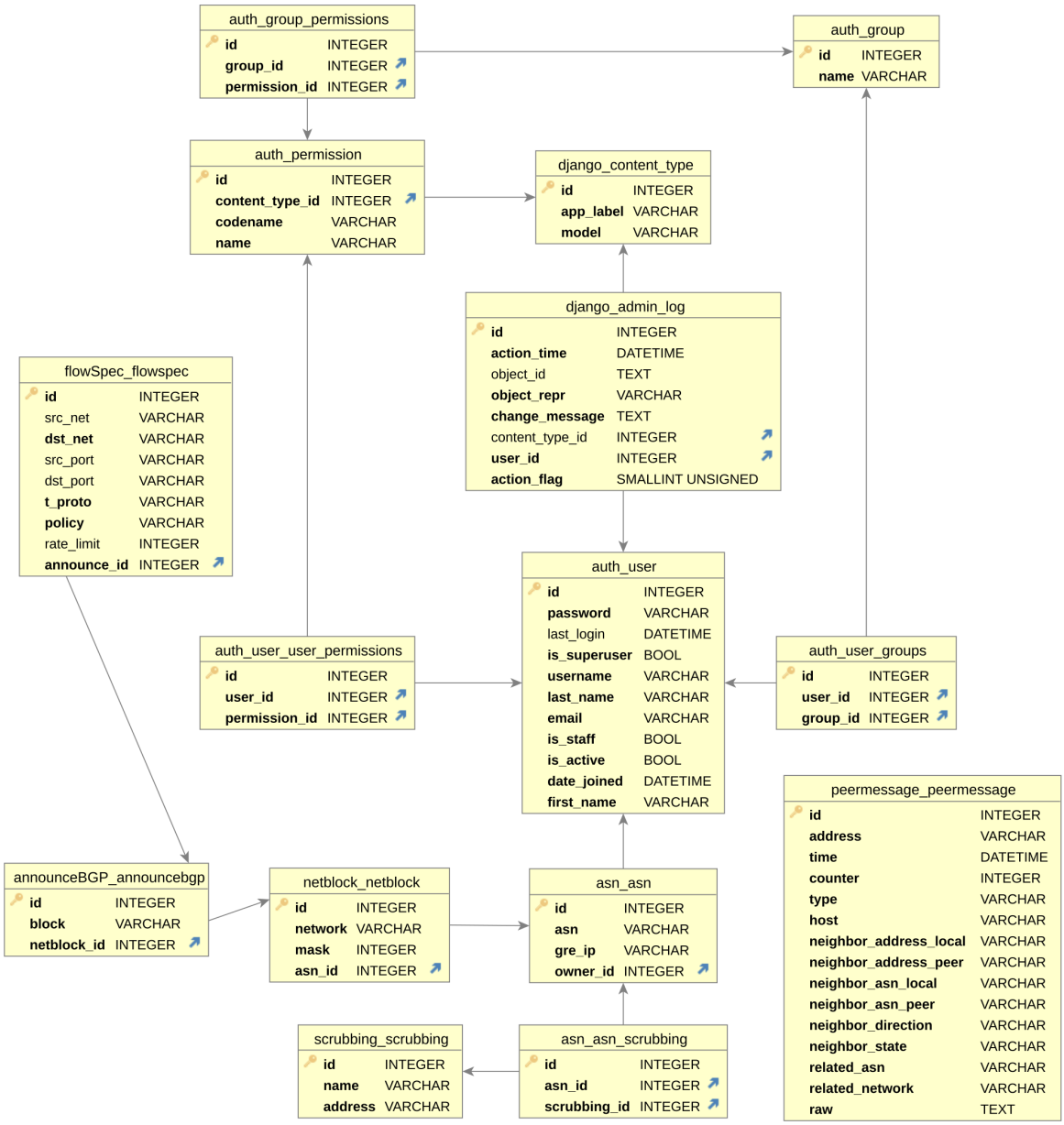
Figura 4.1: Diagrama de componentes del ScrubbingUNLP



4.3.1.1 Interfaz

La interfaz del usuario, o WebScrub, es un sitio web desarrollado con el *framework* Django y capaz de administrar tanto los usuarios del sistema como las características del *scrubbing center*. Se tuvo como premisa desarrollar respetando el modelo MVC [50], siguiendo el concepto de DRY [51], y según el paradigma de desarrollo con orientación a objetos. En la figura 4.2 se muestra el diagrama UML usado de relación de tablas de la base de datos Sqlite [52]. Vale considerar que en las pruebas se utilizó Sqlite por simplicidad y porque resuelve perfectamente las necesidades actuales, pero gracias al *framework* podría ser reemplazada fácilmente por otra base de datos como MySQL [53] o PostgreSQL [54].

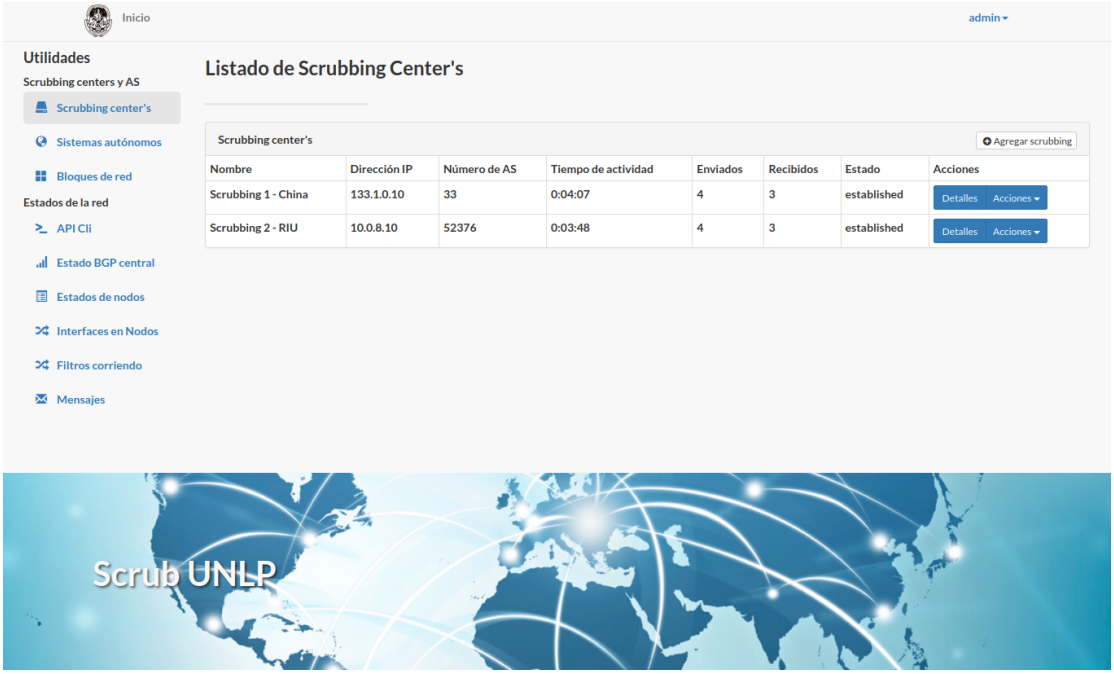
Figura 4.2: Diagrama de la base de datos del WebScrub



El *framework* provee facilidades para la creación del sistema de usuarios, roles y permisos, siendo los roles objetos constituidos dinámicamente por permisos definidos en el panel de administración del sitio, pero se pueden clasificar por defecto dos tipos de roles de usuarios: administradores y clientes.

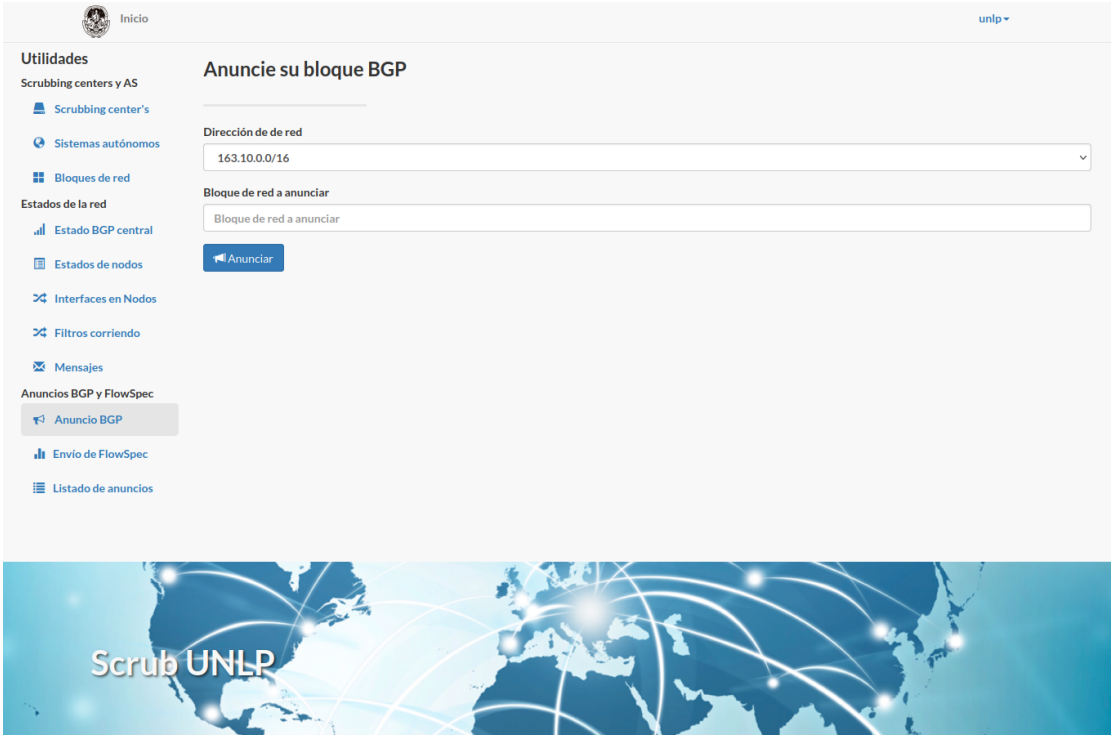
Los administradores del sitio son usuarios con todos los privilegios que, además de poder administrar otros usuarios, pueden crear, editar y eliminar *scrubbing centers*, Sistemas Autónomos y Bloques de red. La figura 4.3 muestra un ejemplo de la vista del administrador para la lista de *scrubbing centers* y en la parte superior derecha cómo crear nuevos.

Figura 4.3: Página de creación de listado de *scrubbing centers* para el administrador



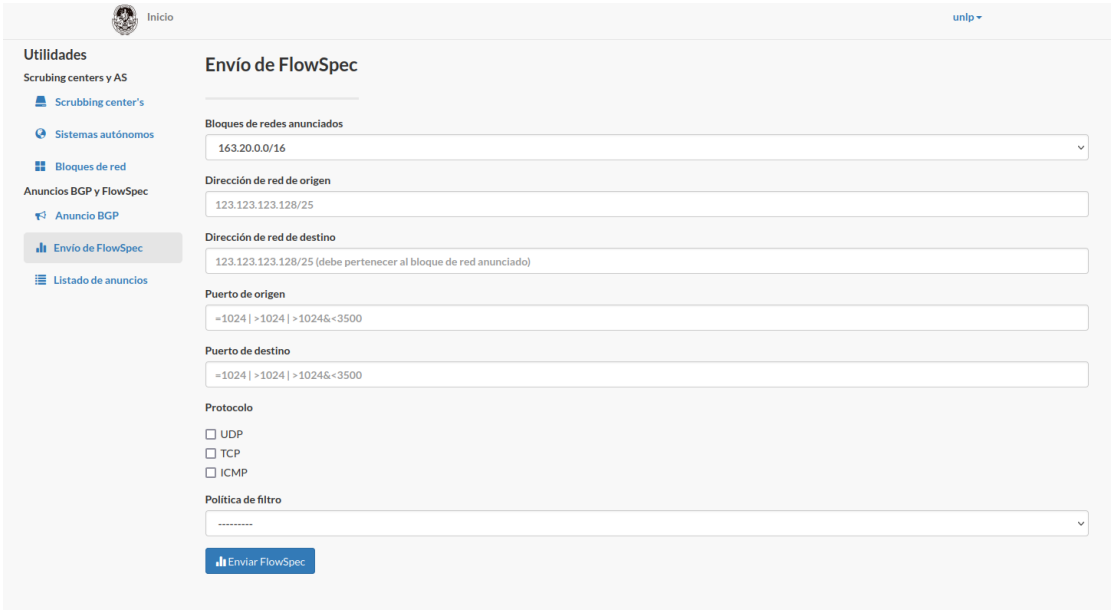
A diferencia de los administradores, los clientes son usuarios del rol con menos privilegios; ellos solo son capaces de administrar publicaciones para los bloques de red asignados a su AS. En otras palabras, un administrador primero debe crear y asignar un AS a un cliente para que este luego tenga capacidad de actuar sobre el mismo. Sus capacidades constan de realizar publicaciones BGP y Flowspec, y eliminarlas. Por otro lado, puede visualizar *scrubbing centers*, estados de red de los mismos, AS y bloques de red pero no editarlos. Estas publicaciones son creadas en la interfaz web y el sistema se encarga de enviarlas al *scrubbing center* para ejecutarlas. La figura 4.4 muestra la vista de creación de anuncios BGP. Una vez conseguido crear un anuncio de rutas BGP, el bloque de red del cliente será publicado por los *scrubbing centers* asignados y por ellos atravesará el tráfico cercano a los mismos con direcciones IP destino pertenecientes al bloque de red del cliente.

Figura 4.4: Página de creación de anuncios BGP para el cliente



El tráfico que atraviesa los *scrubbing centers* será limpiado por las reglas creadas a través de FlowSpec que el cliente defina de acuerdo a sus necesidades. Por ejemplo, puede bloquear el tráfico a determinados *hosts* con prefijos hasta longitud 32, rangos de puertos o puertos específicos, bloques de red origen, o también aplicar *ratelimit* por cantidad de tráfico. Estas opciones se muestran en la figura 4.5.

Figura 4.5: Página de creación de anuncios FlowSpec para el cliente



De esta forma, un cliente puede crear anuncios BGP para bloques de red pertenecientes a su red, con longitudes de máscara entre 16 y 24 bits inclusive. Los anuncios serán enviados a la *API restful* que actúa de repetidor, retransmitiendo el mensaje HTTP a través de ExaBGP. Esta API pertenece al entorno de trabajo de ExaBGP y se la ha definido a través de su configuración como se muestra en las líneas 1 a 5 de la figura 4.6. Luego, en la misma figura entre las líneas 7 a 26, se puede visualizar las configuraciones del propio AS donde corre la interfaz de administración, con el ASN y otras características de BGP. Finalmente, a partir de la línea 28, se observa la configuración de los vecinos del *peering BGP* que son los *scrubbing centers*, en este caso dos.

Figura 4.6: Archivo de configuración de ExaBGP en la interfaz

shared > ExaBGP-ScrubbingCenter > ConfExaBGP > exabgpCentral.ini

```
# Control pipe
process httpAPI {
    run /usr/bin/python3 /opt/exabgp/scripts/http_api.py;
    encoder json;
}

# Changes on the neighbor states
process processNeighborChanges {
    run /usr/bin/python3 /opt/exabgp/scripts/webscrubmessages.py;
    encoder json;
}

# IPv4 template
template {
  neighbor unlp {
    local-as 5692;
    hold-time 180;
    group-updates false;

    capability {
      graceful-restart 120;
    }
    family {
      ipv4 unicast;
      ipv4 flow;
    }
    api {
      processes [ httpAPI ];
      neighbor-changes;
    }
    api {
      processes [ processNeighborChanges ];
      neighbor-changes;
      receive {
        parsed;
        update;
        keepalive;
        operational;
        packets;
        open;
        consolidate;
        refresh;
      }
    }
  }
}

# Neighbours
neighbor 133.1.0.10 {
  inherit unlp;
  peer-as 33;
  local-address 163.10.252.2;
  router-id 163.10.252.2;
  description "ExaBGP China";
}

neighbor 10.0.8.10 {
  inherit unlp;
  peer-as 52376;
  local-address 163.10.252.2;
  router-id 163.10.252.2;
  description "ExaBGP CABASE";
}
```

Es importante destacar en este punto que sobre el WebScrub se han puesto todas las medidas de seguridad pertinentes para que los usuarios sean los únicos capaces de realizar acciones sobre sus sistemas autónomos de modo que no exista forma de realizar publicaciones BGP, filtrado o limpieza de tráfico en los *scrubbing centers* que no haya sido validado por el sistema a partir de que el usuario ingresó al sistema con su usuario y contraseña correctamente para ser identificado.

A su vez, tanto clientes como administradores poseen distintos paneles para ver el estado y la carga de la red en los *scrubbing centers* (figuras 4.7, 4.8, 4.9, 4.10 y 4.11). Esto les brinda un panorama global del sistema y saber si las reglas aplicadas sobre la plataforma para determinados AS son funcionales (para los clientes su propio AS, figura 4.12), en otras palabras, poder ver si el tráfico del ataque atraviesa al *scrubbing center*, a cuál *scrubbing center*, y si la regla de filtrado para limpiar el tráfico está actuando como lo esperan.

Figura 4.7: Vista de salida de comandos BGP en la red, ejemplo de “adj-rib out”

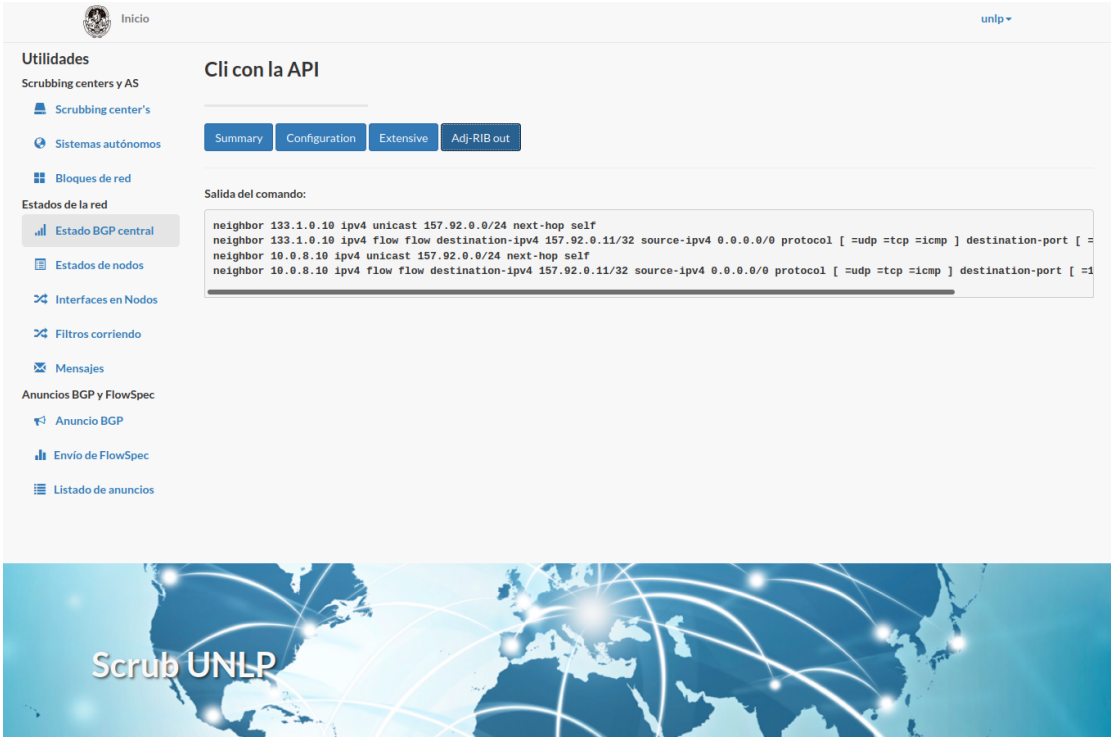


Figura 4.8: Listado de salidas de comandos de red disponibles en nodos

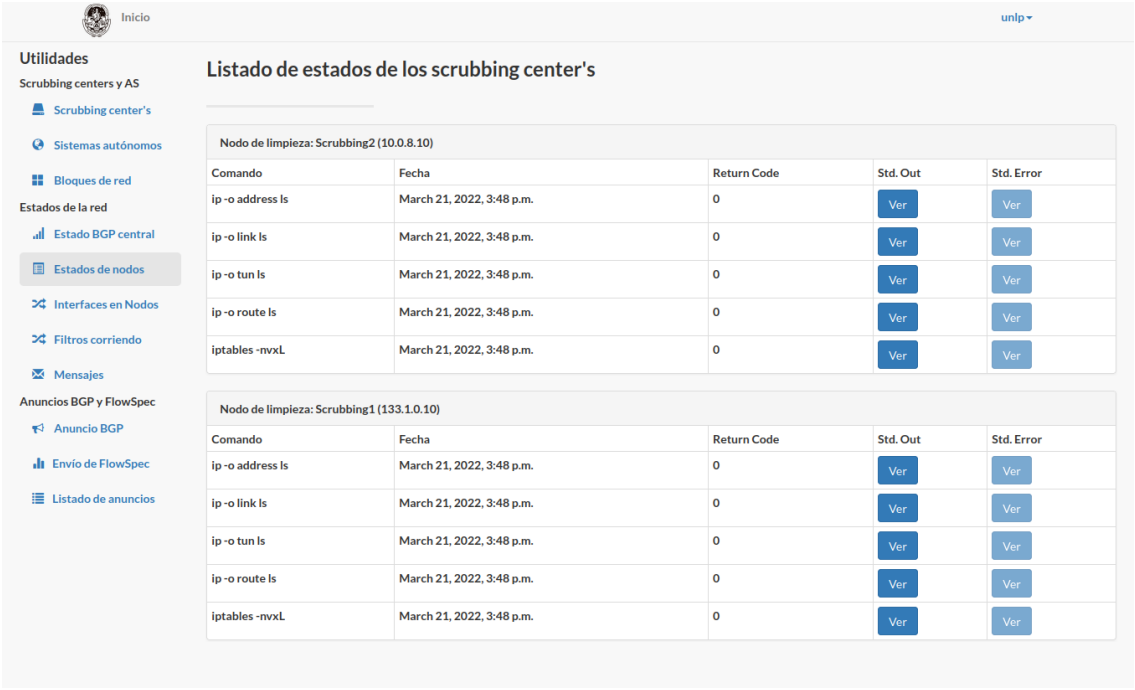


Figura 4.9: Ejemplo de vista del comando “iptables -nvxL”

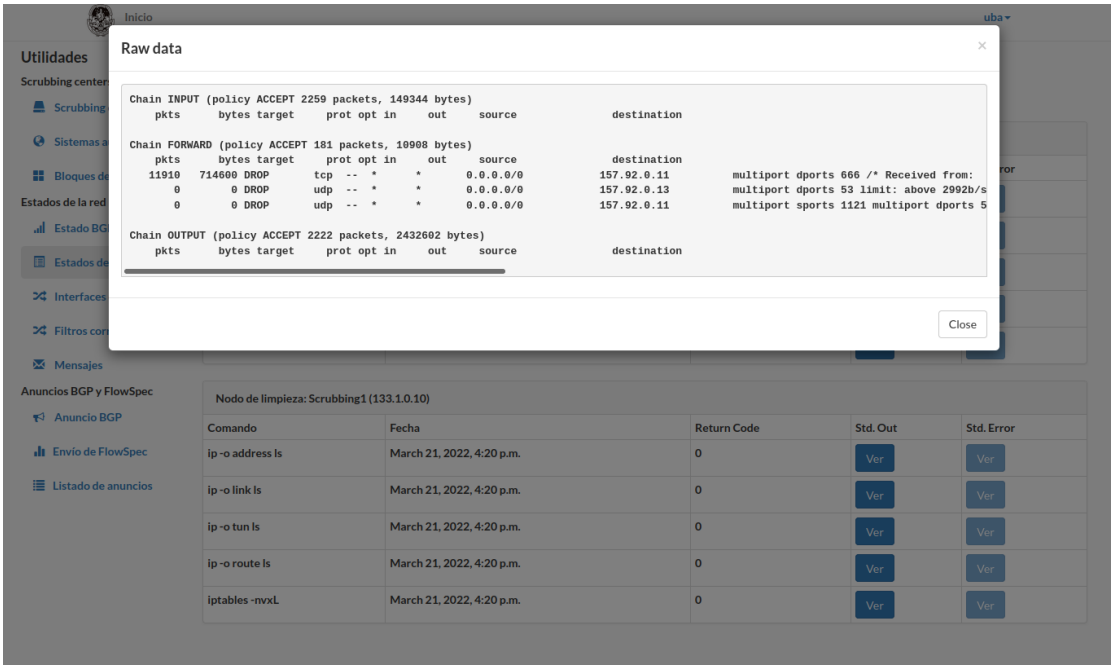
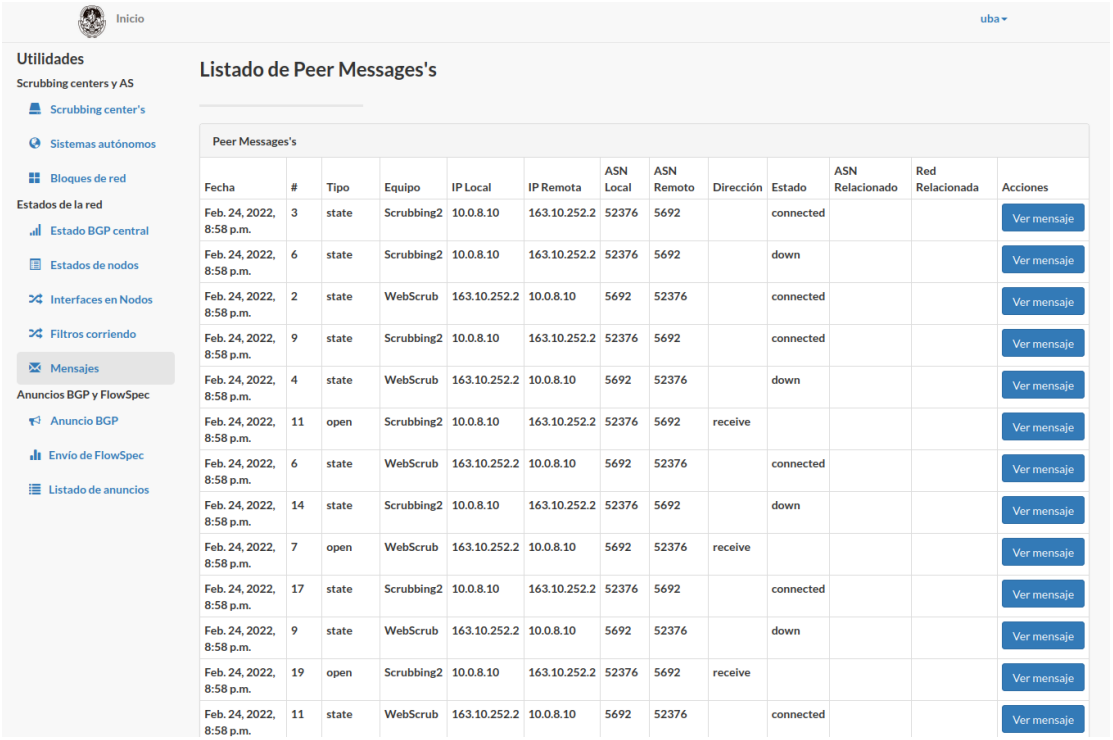
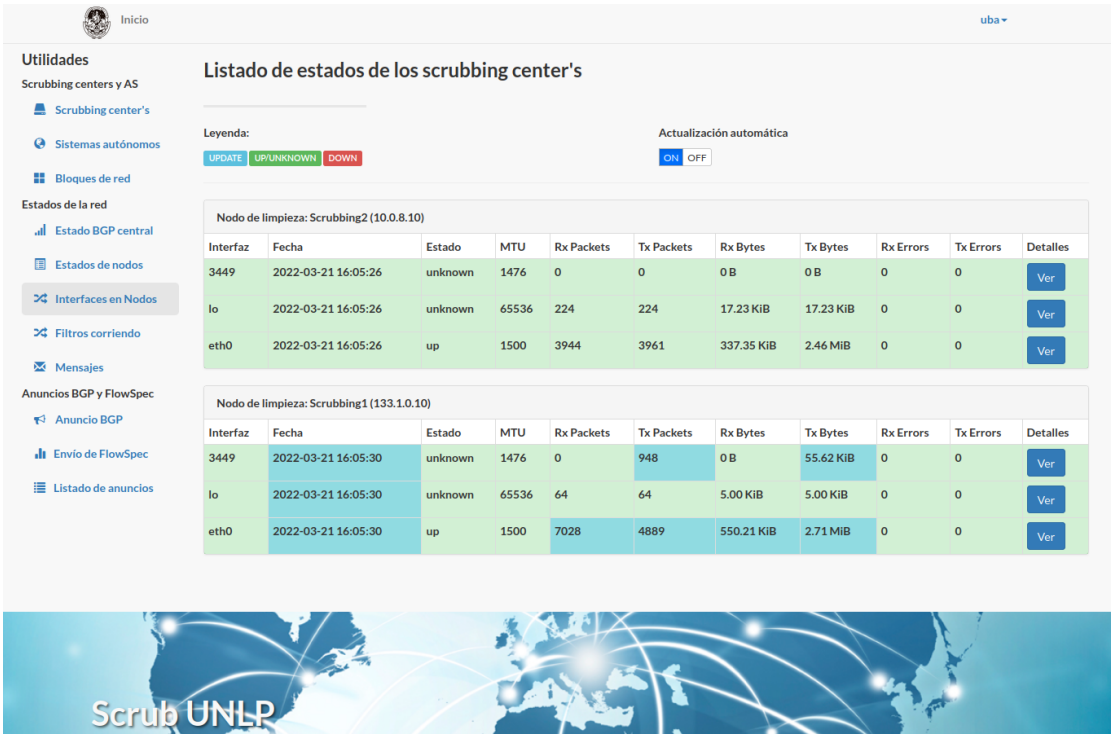


Figura 4.10: Listado de mensajes FlowSpec intercambiados en la red



Fecha	#	Tipo	Equipo	IP Local	IP Remota	ASN Local	ASN Remoto	Dirección	Estado	ASN Relacionado	Red Relacionada	Acciones
Feb. 24, 2022, 8:58 p.m.	3	state	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692		connected			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	6	state	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692		down			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	2	state	WebScrub	163.10.252.2	10.0.8.10	5692	52376		connected			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	9	state	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692		connected			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	4	state	WebScrub	163.10.252.2	10.0.8.10	5692	52376		down			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	11	open	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692	receive				Ver mensaje
Feb. 24, 2022, 8:58 p.m.	6	state	WebScrub	163.10.252.2	10.0.8.10	5692	52376		connected			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	14	state	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692		down			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	7	open	WebScrub	163.10.252.2	10.0.8.10	5692	52376	receive				Ver mensaje
Feb. 24, 2022, 8:58 p.m.	17	state	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692		connected			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	9	state	WebScrub	163.10.252.2	10.0.8.10	5692	52376		down			Ver mensaje
Feb. 24, 2022, 8:58 p.m.	19	open	Scrubbing2	10.0.8.10	163.10.252.2	52376	5692	receive				Ver mensaje
Feb. 24, 2022, 8:58 p.m.	11	state	WebScrub	163.10.252.2	10.0.8.10	5692	52376		connected			Ver mensaje

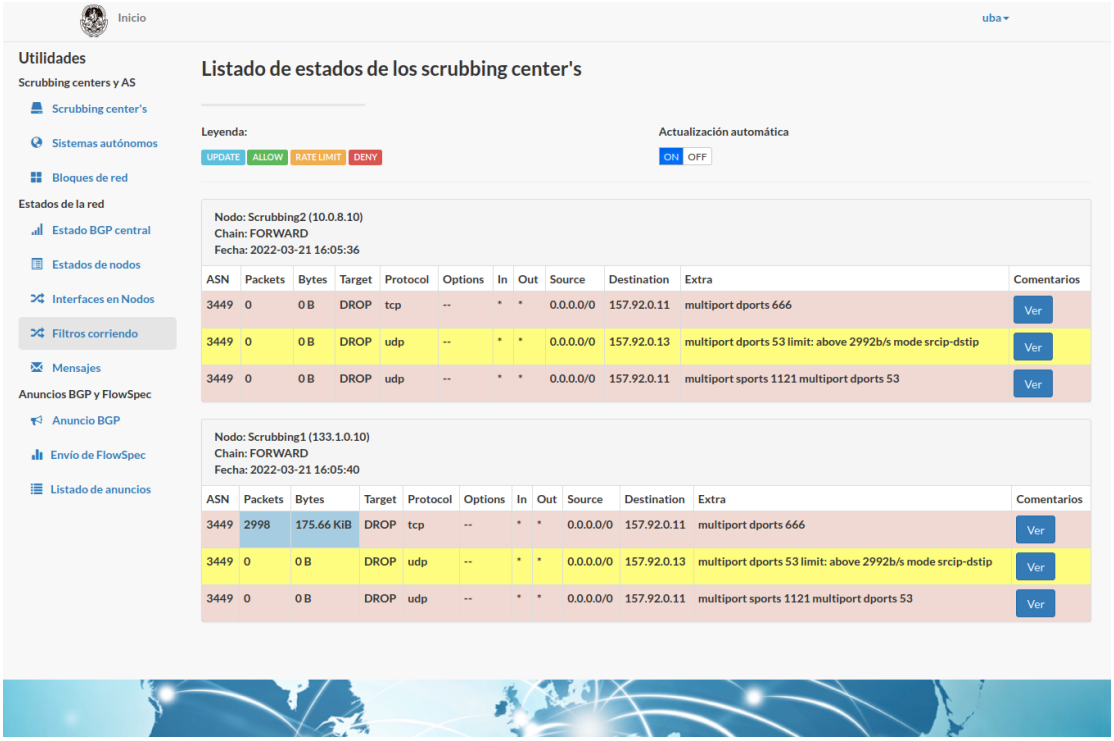
Figura 4.11: Vista de interfaces importantes de nodos *scrubbing center*; en azul se remarcan en tiempo real los cambios de valores para cada interfaz



Interfaz	Fecha	Estado	MTU	Rx Packets	Tx Packets	Rx Bytes	Tx Bytes	Rx Errors	Tx Errors	Detalles
3449	2022-03-21 16:05:26	unknown	1476	0	0	0 B	0 B	0	0	Ver
lo	2022-03-21 16:05:26	unknown	65536	224	224	17.23 KiB	17.23 KiB	0	0	Ver
eth0	2022-03-21 16:05:26	up	1500	3944	3961	337.35 KiB	2.46 MiB	0	0	Ver

Interfaz	Fecha	Estado	MTU	Rx Packets	Tx Packets	Rx Bytes	Tx Bytes	Rx Errors	Tx Errors	Detalles
3449	2022-03-21 16:05:30	unknown	1476	0	948	0 B	55.62 KiB	0	0	Ver
lo	2022-03-21 16:05:30	unknown	65536	64	64	5.00 KiB	5.00 KiB	0	0	Ver
eth0	2022-03-21 16:05:30	up	1500	7028	4889	550.21 KiB	2.71 MiB	0	0	Ver

Figura 4.12: Vista de reglas de filtrado aplicadas en nodos *scrubbing center*; en rojo se remarcan las reglas *DENY*, en amarillo las reglas *RATE LIMIT* y en azul los campos actualizados en tiempo real cuando el tráfico coincide con la regla



4.3.1.2 Scrubbing Center

Los *scrubbing centers* son máquinas remotas que preferentemente tendrán que estar ubicados lo más cercano posible a un Tier 1 en Internet para poder procesar la mayor cantidad de tráfico posible de diferentes orígenes. Pero esto no es sumamente necesario, también pueden haber múltiples en diferentes niveles y ubicaciones geográficas permitiéndoles conseguir el mismo o hasta mejor resultado. Lo importante es considerar los costos entre las diferentes posibilidades. Que existan más máquinas distribuidas también generaría mayor probabilidad de frenar un posible ataque cerca del origen lo que vuelve a la solución más eficiente en costos de tráfico. El principal inconveniente a afrontar en este caso es que no es sencillo para una organización pequeña ubicar máquinas con *peering* BGP a un costo asequible.

Las ubicaciones geográficas de los ataques son un factor muy importante ya que un ataque con orígenes distribuidos en todo el mundo no va a generar las mismas cargas en los *scrubbing centers* que si el ataque proviene en su completitud desde el "otro extremo" de Internet o, más difícil de prevenir aún, si el ataque viene de muy muy cerca. Los *scrubbing centers*, por el comportamiento de BGP, absorben el tráfico dirigido hacia la red que ellos publican y por lo tanto cuanto más cerca atraviese el ataque por Internet cerca de las rutas BGP publicadas por los *scrubbing centers*, serán más efectivas para atraer ese tráfico y limpiarlo.

Los *scrubbing centers* a partir de anuncios FlowSpec creados en la interfaz, absorberán tráfico malicioso del ataque y retransmitirán el tráfico limpio hacia el destino. Como ya se mencionó, esto se logra primero creando en la interfaz un anuncio BGP para un bloque de red que es transmitida por el *peering* BGP de ExaBGP hacia los *scrubbing centers*, estos se encargan de interpretar el mensaje y realizar la publicación

BGP. En la figura 4.13 se muestra un ejemplo de un mensaje de creación de ruta BGP para el ASN 52376, se le pide publicar la red 163.10.10.0/24 perteneciente al AS 5692; en la primer línea se muestra el log y al final el formato JSON utilizado por ExaBGP entre nodos del *peering*.

Figura 4.13: Log del mensaje transmitido para la publicación de rutas BGP

```
23:33:23 | 43     | api           | route added to neighbor 133.1.0.10 local-ip 163.10.252.2 local-as 5692 pee
r-as 33 router-id 163.10.252.2 family-allowed in-open, neighbor 10.0.8.10 local-ip 163.10.252.2 local-as 5692
peer-as 52376 router-id 163.10.252.2 family-allowed in-open : 163.10.10.0/24 next-hop self as-path [ 5692 ]
```

Luego de atraer el tráfico del ataque, un *scrubbing center* tiene como objetivo efectuar una limpieza del tráfico definida por el cliente a través del anuncio FlowSpec definido en la interfaz del WebScrub.

Las reglas FlowSpec, diseñadas para detener tráfico de ataques DDoS, tienen un formato y campos similares a las reglas de un *firewall*. Los campos que el cliente debe definir son:

- **Bloque de red anunciado**: Obligatorio. Debe efectuar la regla FlowSpec sobre un bloque de red previamente anunciado en BGP por el *scrubbing center*, sobre el cual se aplicará el filtro.
- **Dirección de red origen**: Opcional. Selecciona las IP origen sobre las cuales el filtro actuará. Si es ausente se aplica sobre cualquier IP origen.
- **Dirección de red de destino**: Obligatorio. Subconjunto de IP destino, perteneciente al bloque de red anunciado, sobre las cuales se aplican los filtros.
- **Puerto de origen**: Opcional. Rango de puertos origen sobre los cuales se aplica el filtro. Si es ausente se aplica sobre cualquier puerto origen.
- **Puerto de destino**: Opcional. Rango de puertos destino sobre los cuales se aplica el filtro. Si es ausente se aplica sobre cualquier puerto destino.
- **Protocolo**: Obligatorio. Se debe seleccionar al menos una opción y define los protocolos sobre los cuales el filtro de aplica; UDP, TCP y/o ICMP.
- **Política de filtro**: Obligatorio. Es un campo de opción y define la política a ejecutar para el tráfico que coincida con la regla. Las opciones son: aceptar el tráfico, descartar el tráfico o aplicar un límite de tasa. Si se selecciona este último se agregará un campo más que define la cantidad de tráfico máximo a permitir para la regla en bytes/segundo.

De esta forma, si se desea descartar todo el tráfico UDP al destino 163.10.0.199/32 con origen 123.123.123.123/32 se aplicará el siguiente anuncio FlowSpec:

```
announce flow route {
  match {
    source 123.123.123.123/32;
    destination 163.10.0.199/32;
    protocol UDP;
  } then {
    discard;
```

```
  }
}
```

Si se tiene un ataque de tipo DDoS amplificado hacia el host 163.10.0.155 utilizando el protocolo Memcached [55] se deberá realizar un anuncio FlowSpec como el siguiente:

```
announce flow route {
  match {
    source 0.0.0.0/0;
    destination 163.10.0.155/32;
    source-port =11211;
    protocol ['UDP'];
  } then {
    discard;
  }
}
```

Si, por ejemplo, se cuenta con un servicio esencial como es DNS en la IP 163.10.0.2, el cual no se puede bloquear por completo ya que impediría el funcionamiento de otros sistemas pero el ataque viene por ese medio, se puede aplicar una regla de tipo *rate limit* para limitar la cantidad de tráfico considerando que el funcionamiento de DNS usualmente es de una tasa en bits baja, un ataque volumétrico sería fácil de diferenciar por la utilización de arriba de los cientos de megabytes por segundo para una IP origen. Entonces se aplicaría una regla como la siguiente:

```
announce flow route {
  match {
    source 0.0.0.0/0;
    destination 163.10.0.2/32;
    protocol ['UDP' 'TCP' 'ICMP'];
  } then {
    rate-limit 100000;
  }
}
```

Otro caso posible es que, combinado al caso anterior, el ataque esté dirigido a toda la red del servidor de DNS 163.10.0.2, por ejemplo, toda la red 163.10.0.0/24, saturando el enlace por completo. En ese caso la regla anterior no alcanzaría. Siendo que en los nodos las reglas se aplican en orden inverso a las definidas, será necesario previamente aplicar otra regla que deniegue todo el tráfico hacia 163.10.0.0/24, que se ejecutará última en el nodo. Al aplicar las reglas de este modo, el nodo primero analizará sobre el tráfico la regla con destino 163.10.0.2/32 (con *rate limit*) y luego la regla con destino 163.10.0.0/24 (con *discard*).

```
announce flow route {
  match {
    source 0.0.0.0/0;
```

```
    destination 163.10.0.0/24;
    protocol ['UDP' 'TCP' 'ICMP'];
  } then {
    discard;
  }
}

announce flow route {
  match {
    source 0.0.0.0/0;
    destination 163.10.0.2/32;
    protocol ['UDP' 'TCP' 'ICMP'];
  } then {
    rate-limit 100000;
  }
}
```

Estas reglas FlowSpec son recibidas en los *scrubbing centers* por ExaBGP. ExaBGP provee una configuración sobre la cual se pueden definir procesos a ejecutar a partir de determinados eventos recibidos de los *peerings*. Se ha creado sobre esta estructura los procesos que proveen a la plataforma toda la funcionalidad de actuar como una red de *scrubbing centers* distribuidos.

Como muestra la figura 4.14 entre las líneas 1 a 26 se crean los procesos que ejecutan diferentes actividades dentro de los nodos y en la figura 4.15, que es la segunda parte del mismo archivo, se muestra cuándo estos procesos son utilizados. A continuación se describe su funcionamiento y el momento de su ejecución:

- **Proceso "receive-routes-central"**: Es uno de los procesos más importantes a ejecutar en el nodo ya que aplica las reglas de filtrado sobre el mismo para el tráfico que lo atraviesa y funciona a partir de traducir las reglas FlowSpec a reglas ACL que luego serán convertidas al *firewall* real implementado en el sistema y finalmente ejecutadas. El proceso se ejecuta al recibir mensajes de tipo *parsed*, *update*, *operational*, *packets*, *open*, *consolidate* y *refresh*.

- **Proceso "parse-routes-central"**: Este proceso se encarga de crear, actualizar y bajar las interfaces, túneles y redes necesarias en el nodo para el correcto funcionamiento de la redistribución del tráfico filtrado por los túneles hacia los destinos. El proceso se ejecuta al recibir mensajes de tipo *update*.

- **Proceso "httpAPI"**: Se encarga de crear una API HTTP para ejecutar comandos ExaBGP y así proveer otro medio de comunicación con la plataforma. Se ejecuta en todo tipo de mensajes ExaBGP.

- **Proceso "processNeighborChanges"**: Este proceso envía al WebScrub a través de su API los mensajes ExaBGP recibidos en el nodo para tener control y conocimiento de todo lo transmitido en la red. El proceso se ejecuta al recibir mensajes de tipo *parsed*, *update*, *operational*, *packets*, *open*, *consolidate* y *refresh*.

- **Proceso "processNodeStatus"**: Es el proceso encargado de mantener el WebScrub con el estado actual del nodo al enviarle la salida de varios comandos de red necesarios para verificar su estado, junto al estado de todas las interfaces, que en sistemas Linux se ofrece a través de la ruta "/sys/class/net/". No se ejecuta al percibir un evento sino que permanece activo actualizando el WebScrub cada diez segundos.

Figura 4.14: Primera parte de la definición de ejecución de procesos por tipos de mensajes FlowSpec

shared > ExaBGP-ScrubbingCenter > ConfExaBGP > exabgpScrubbing1.ini

```
process receive-routes-central {
  run /usr/bin/python3 /opt/exabgp/scripts/acl.py;
  encoder json;
}

process parse-routes-central {
  run /usr/bin/python3 /opt/exabgp/scripts/parseroute.py;
  encoder json;
}

process httpAPI {
    run /usr/bin/python3 /opt/exabgp/scripts/http_api.py;
    encoder json;
}

# Changes on the neighbor states
process processNeighborChanges {
    run /usr/bin/python3 /opt/exabgp/scripts/webscrubmessages.py;
    encoder json;
}

# Status of the scrubbing center node
process processNodeStatus {
    run /usr/bin/python3 /opt/exabgp/scripts/webscrubstatus.py;
    encoder json;
}

# IPv4 template
template {
  neighbor scrubbing {
    local-as 33;
    hold-time 180;
    group-updates false;
    capability {
      graceful-restart 120;
    }
    family {
      ipv4 unicast;
      ipv4 flow;
    }
  }
}

template {
  neighbor QuaggaPeer {
    local-as 33;
    peer-as 33;
    hold-time 180;
    group-updates false;
    capability {
      graceful-restart 120;
    }
    family {
      ipv4 unicast;
      ipv4 flow;
    }
    api{
    processes [ httpAPI ];
    neighbor-changes;
    }
  }
```

Figura 4.15: Segunda parte de la definición de ejecución de procesos por tipos de mensajes FlowSpec

shared > ExaBGP-ScrubbingCenter > ConfExaBGP > exabgpScrubbing1.ini

```
64
65  # ExaBGP peering to Central ExaBGP
66  neighbor 163.10.252.2 {
67    inherit scrubbing;
68    peer-as 5692;
69    router-id 133.1.0.10;
70    local-address 133.1.0.10;
71    description "ExaBGP to ExaBGPCentral";
72    api {
73      processes [ parse-routes-central ];
74      neighbor-changes;
75      receive {
76        update;
77      }
78    }
79    api {
80      processes [ receive-routes-central ];
81      neighbor-changes;
82      receive {
83        parsed;
84        update;
85        operational;
86        packets;
87        open;
88        consolidate;
89        refresh;
90      }
91    }
92    api {
93      processes [ processNeighborChanges ];
94      neighbor-changes;
95      receive {
96        parsed;
97        update;
98        operational;
99        packets;
100       open;
101       consolidate;
102       refresh;
103     }
104   }
105   api {
106     processes [ processNodeStatus ];
107   }
108 }
109
110 neighbor 133.1.0.1 {
111   inherit QuaggaPeer;
112   router-id 133.1.0.10;
113   local-address 133.1.0.10;
114   description "Peering with quagga router to announce";
115 }
116
```

Figura 4.16: Sección de código relevante de parseroute.py definido para interpretar mensajes *update* BGP

shared > ScrubbingUNLP > scripts > parseroute.py > ...

```
159
160 while True:
161     try:
162         line = stdin.readline().strip()
163
164         # When the parent dies we are seeing continual newlines, so we only access so many before       #stopping
165         if line == "":
166             counter += 1
167             if counter > 100:
168                 break
169             continue
170         counter = 0
171
172         message = json.loads(line)
173
174         if message["type"] == "state":
175             if message['neighbor']['state'] in ["down", "connected"]:
176                 logger.warning(f"Peer {message['neighbor']['state']}. Limpiando rutas")
177                 send_cmd('clear adj-rib')
178                 tuneles = get_tunnels()
179                 logger.info(f'Eliminando tuneles: {tuneles}')
180                 for tun in tuneles:
181                     logger.info(f'Eliminando tunel {tun}')
182                     remove_gre(tun, force=True)
183
184         elif message["type"] == "update":
185             try:
186                 neighbor = message['neighbor']
187                 update = neighbor['message']['update']
188                 if 'announce' in update.keys():
189                     peer = neighbor['address']['peer']
190                     local = neighbor["address"]["local"]
191                     array  = update['announce']['ipv4 unicast'][peer]
192                     net = array[0]['nlri']
193                     message_type = "announce"
194                     action = "add"
195                     asn_remote = update['attribute']['as-path'][0]
196                     create_or_up_gre(local, asn_remote)
197                     value = f"announce route {net} next-hop self origin igp as-path [{asn_remote}]"
198                     aplicar_ruta(value, action, net, asn_remote)
199                 elif 'withdraw' in update.keys():
200                     array  = update['withdraw']['ipv4 unicast']
201                     net = array[0]['nlri']
202                     action = "del"
203                     message_type = "withdraw"
204                     asn_remote = extraer_asn_de_ruta(net)
205                     value = f"withdraw route {net}"
206                     aplicar_ruta(value, action, net, asn_remote)
207                     if asn_remote:
208                         down_gre(asn_remote)
209                     # remove_orphan_gre()
210
211
212
213             except KeyError as detail:
214                 pass
215     except Exception as e:
216         logger.exception('Exception in main loop')
217
218
219
```

Por ejemplo, un mensaje BGP de anuncio de rutas es transmitido en ExaBGP con el siguiente formato JSON:


```
{
  'exabgp': '4.0.1',
  'time': 1643733595.2644334,
  'host': 'n36',
  'pid': 33,
  'ppid': 1,
  'counter': 11,
  'type': 'update',
```

```
  'header': '0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF003002',
  'body': '0x00000014400101004002060201000016 3C400304A314FC0219A3140880',
  'neighbor': {
    'address': {
      'local': '133.1.0.10',
      'peer': '163.10.252.2'
    },
    'asn': {
      'local': 33,
      'peer': 5692
    },
    'direction': 'receive',
    'message': {
      'update': {
        'attribute': {
          'origin': 'igp',
          'as-path': [5692],
          'confederation-path': []
        },
        'announce': {
          'ipv4 unicast': {
            '163.10.252.2': [{'nlri': '163.10.8.128/25'}]
          }
        }
      }
    }
  }
}
```


Y luego es convertido por 'parseroute.py' a una ejecución en el sistema del siguiente comando:

```
ip route add 163.10.8.128/25 dev 5692
```

El *script* "acl.py" modulariza el proceso de aplicación de reglas FlowSpec en el sistema de *firewall* que el nodo esté utilizando. Esto se realiza a partir de la utilización de objetos de Python, con un patrón de diseño llamado *Strategy* se instancia la clase que se desea utilizar del traductor FlowSpec-iptables, así puede ser extensible e intercambiable el conjunto de firewalls a usar en función de las capacidades de cada nodo.

Finalmente, con el tráfico ya limpio a partir de las reglas del *firewall*, el último punto es enrutar el tráfico hacia el cliente. Esto se logra a partir del uso del túnel entre el *scrubbing center* y el cliente, el cual fue creado previamente cuando se realizó la publicación de redes por BGP ejecutadas por el *script* "parseroute.py" antes mencionado.

4.3.1.3 Túnel

Los túneles, creados manualmente, son del tipo GRE y permiten un encapsulamiento liviano de baja sobrecarga. El tráfico limpio del cliente ingresa a través del *scrubbing center* y sale en un nodo previamente creado ubicado en la organización del cliente, con el otro extremo del túnel. Sólo tráfico dirigido a la organización saldrá por aquí gracias a las reglas de publicación de rutas BGP previamente definidas por el *script* "parseroute.py" y creadas por el cliente a través de la interfaz web. De esta manera, se provee al cliente un túnel dedicado para el tráfico filtrado hacia su organización.

4.3.2 Instalación y dependencias

A continuación se detalla la instalación y dependencias de WebScrub y ScrubbingUNLP por separado y cómo configurar ambas herramientas para que puedan trabajar en conjunto.

4.3.2.1 ScrubbingUNLP

Se considera que la instalación se realiza sobre un sistema operativo basado en Debian. Para otros sistemas operativos es necesario analizar las opciones equivalentes de cada herramienta utilizada. Este servicio debe ser instalado en todos los nodos que ejecuten ExaBGP y deben contar con las configuraciones y *peerings* BGP adecuados.

Para la instalación de ScrubbingUNLP se debe contar con **Python3** (viene instalado por defecto en la mayoría de los sistemas operativos basados en Linux), **git**, **python3-pip** y **ExaBGP**.

```
sudo apt-get -y install python3-pip git
```

Para instalar ExaBGP, se debe clonar su repositorio de GitHub (https://github.com/Exa-Networks/exabgp) e instalarlo mediante el uso del módulo zipapp de python3:

```
sudo git clone https://github.com/Exa-Networks/exabgp /opt/exabgp/
cd /opt/exabgp/
sudo git checkout 4.2.11
sudo python3 -m zipapp -o /usr/local/sbin/exabgp \
       -m exabgp.application:main  -p "/usr/bin/env python3" lib
```

Luego de instalar las dependencias, se continúa con la instalación de ScrubbingUNLP. Para ello, se debe clonar el repositorio de Github (`https://github.com/mateodurante/ScrubbingUNLP`) e instalar las dependencias necesarias para el funcionamiento de los *scripts* y la API en python3 que utiliza ExaBGP:

```
cd /opt/
sudo git clone https://github.com/mateodurante/ScrubbingUNLP
sudo pip3 install -r /opt/ScrubbingUNLP/requirements.txt
```

El siguiente paso consiste en mover los *scripts* a la carpeta de ExaBGP para que este sea capaz de ejecutarlos.

```
sudo cp /opt/ScrubbingUNLP/scripts/*.py /opt/exabgp/scripts/
```

Finalmente, se cede permisos de ejecución al script llamado **start.sh** que, luego de configurar los parámetros necesarios que se explicarán en la siguiente sección, ejecutará el servicio ScrubbingUNLP en el nodo.

```
sudo chmod +x /opt/ScrubbingUNLP/start.sh
```

4.3.3 WebScrub

En el nodo que se vaya a ejecutar WebScrub, se debe instalar python3-pip y git. Luego, se clona el repositorio de WebScrub, instala dependencias de la aplicación web, también es necesario aplicar las migraciones para la base de datos del sitio:

```
git clone https://github.com/mateodurante/WebScrub.git /opt/WebScrub
cd /opt/WebScrub
cp db.sqlite3.initial db.sqlite3
sudo pip3 install -r requirements.txt
python3 manage.py makemigrations
python3 manage.py migrate
```

Finalmente, se ejecuta el servicio (en este momento, se ejecuta en modo desarrollo):

```
sudo python3 /opt/WebScrub/manage.py runserver 0.0.0.0:80
```

Una vez iniciado el servicio, este ya se encuentra listo para ser usado accediendo mediante la URL http://<IP_NODO_WEB>:80. Las credenciales de usuario administrador son por defecto: *administrador:administrador*.

Si el *scrubbing center* central está ejecutando en la misma máquina que la web, no es necesario realizar ningún cambio. Si, por otro lado, está ejecutando en un host diferente, es necesario actualizar la variable HTTP_API del settings.py antes de iniciar el servicio.

4.3.4 Configuraciones de ScrubbingUNLP

Para poder ejecutar correctamente el proyecto ScrubbingUNLP será necesario preparar las configuraciones necesarias del nodo central y los *scrubbing centers* a través de sus archivos de extensión ".ini.default" como muestran los ejemplos definidos en la carpeta "./config_examples" del repositorio. Los archivos de configuración deben presentar la extensión ".ini", por lo que se deberá remover el sufijo ".default" y dar la ruta del archivo como parámetro al momento de ejecutar ExaBGP sobre el nodo. En la carpeta se ubican tres archivos de ejemplo, uno para el nodo central y otros dos para nodos de *scrubbing center*. Para el nodo central se parte del archivo "exabgpCentral.ini.default", sobre el que se deben modificar todos los campos encerrados entre los símbolos "<" y ">".

- <ASN-Central>: Es el número de AS utilizado en el nodo Central, éste puede ser un ASN del rango reservado para uso privado [56].
- <IP-Central>: Es la IP del nodo central que habla ExaBGP.
- <ASN-ScrubbingX>: Es el número de AS utilizado en el *scrubbing center* en cuestión para el *peering*, para este caso el valor "X" pero para cada *scrubbing center* se utilizará un valor distinto.

- <IP-ScrubbingX>: Es la IP del *scrubbing center* en cuestión para el *peering*, para este caso el valor "X" pero para cada *scrubbing center* se utilizará un valor distinto.

Las secciones de ASN-ScrubbingX e IP-ScrubbingX se repetirán la cantidad de *scrubbing centers* existentes en la red reemplazando X por un valor diferente en cada caso. Del mismo modo ocurrirá al definir los archivos de configuración, donde cada *scrubbing center* tendrá su propio "exabgpScrubbingX.ini" definido. Dentro del archivo "exabgpScrubbingX.ini.default" se deberá reemplazar bajo los mismos criterios que el "exabgpCentral.ini.default" los campos mencionados.

4.3.5 Uso

Se ha simplificado la instalación de la plataforma en cada nodo a través del archivo "install.sh" definido en la raíz del repositorio. El mismo se encarga de instalar entre otras cosas los requerimientos de la plataforma que son ExaBGP y módulos de Python, pero como ya se ha mencionado, para poder utilizarla correctamente además sobre el nodo central o alguna máquina conectado a este, será necesario instalar WebScrub para que el administrador y los clientes puedan interactuar con la plataforma.

Para la ejecución se ha creado un *script* llamado "start.sh" que corre el módulo ExaBGP con parámetros para cada nodo, minimizando la posibilidad de fallas y errores. Los parámetros son:

- -b|–bind-ip: La IP a donde estará ligado el proceso ExaBGP, se recomienda utilizar la IP pública del nodo sobre el que se ejecuta.
- -c|–config-file: Ruta al archivo ".ini" donde se levantarán las configuraciones ya mencionadas.
- -w|–webscrub-url: Es la URL donde se ubica el sitio de WebScrub, se utiliza para enviar la información del estado de los nodos.
- -f|–fprobe: Es opcional e indica la IP donde enviar los datagramas de NetFlow para visualizar los flujos de tráfico que atraviesan las interfaces del nodo. El nodo debe tener **fprobe** instalado previamente.

CAPÍTULO 5

ENTORNO DE PRUEBAS

Una vez detalladas todas las características de la plataforma propuesta se está en condiciones de ver un ejemplo del funcionamiento. Se verá a través de esta demostración un caso de ataque de denegación de servicio distribuido en Internet sobre una topología posicionada geográficamente en puntos similares a los reales pero extremadamente simplificada para facilitar su lectura.

Con el ataque efectuado se procederá a activar el sistema para realizar una limpieza de tráfico, con lo cual se pretende demostrar la efectividad del sistema de limpieza de tráfico de red.

5.0.1 Herramientas de la simulación

A continuación se darán breves definiciones oficiales de las herramientas utilizadas para el entorno de pruebas de la tesina.

5.0.1.1 CORE

CORE (Common Open Research Emulator) es una herramienta que permite construir redes virtuales [57]. Como emulador, CORE crea una representación de una red informática real con sus dispositivos involucrados (computadoras, *routers*, *switchs*, *hubs*, etc.) que se ejecuta en tiempo real, a diferencia de la simulación, donde se utilizan modelos abstractos. La emulación de ejecución en vivo se puede conectar a redes físicas y *routers* permitiéndole interactuar con los mismos. Proporciona un entorno para ejecutar aplicaciones y protocolos reales, aprovechando las herramientas proporcionadas por el sistema operativo Linux sobre el que se instala.

CORE se utiliza normalmente para la investigación de redes y protocolos, demostraciones, pruebas de aplicaciones y plataformas, evaluación de escenarios de redes, estudios de seguridad y aumento del tamaño de las redes de pruebas físicas. El código fuente de CORE consta de varios lenguajes de programación diferentes por razones históricas. El desarrollo actual se centra en los módulos y *daemons* escritos en el lenguaje Python.

5.0.1.2 hping3

hping3 [58] es una herramienta de red capaz de enviar paquetes personalizados ICMP/UDP/TCP y mostrar respuestas de destino como lo hace *ping* con respuestas ICMP. Maneja la fragmentación, el cuerpo y tamaño arbitrario de los paquetes, y se puede usar para transferir archivos bajo protocolos compatibles. Con hping3 se puede probar las reglas del *firewall*, realizar *scans* de puertos (falsificados), probar el rendimiento de la red usando diferentes protocolos, descubrir MTU de ruta, realizar acciones similares a *traceroute* bajo diferentes protocolos, huellas dactilares de sistemas operativos remotos, auditar pilas TCP/IP, etc. hping3 es programable utilizando el lenguaje TCL.

5.0.1.3 FRRouting

FRRouting (FRR) es un conjunto de protocolos de enrutamiento de Internet gratuito y de código abierto para plataformas Linux y basadas en Unix en general [59]. Implementa BGP, OSPF, RIP, IS-IS, PIM, LDP, BFD, Babel, PBR, OpenFabric y VRRP, con soporte alfa para EIGRP y NHRP.

La integración de FRR con el *stack* IP nativo de Linux/Unix lo convierte en un sistema de enrutamiento de propósito general aplicable a una amplia variedad de casos de uso, incluida la conexión de *hosts*/VM/contenedores a la red, publicidad de servicios de red, conmutación y enrutamiento de LAN, enrutadores de acceso a Internet, y emparejamiento de Internet.

FRR tiene sus raíces en el proyecto Quagga. De hecho, fue iniciado por muchos desarrolladores de Quagga desde hace mucho tiempo que combinaron sus esfuerzos para mejorar la base bien establecida de Quagga para crear una mejor pila de protocolos de enrutamiento disponible.

Esta herramienta permite que una computadora con Linux (o *Unix-like*) instalado, pueda entender y administrar protocolos de enrutamiento convirtiendo la máquina en un *router* más en la red física.

5.0.1.4 Docker

Docker es una plataforma abierta para desarrollar, enviar y ejecutar aplicaciones [60]. Permite separar sus aplicaciones de infraestructura para entregar software rápidamente. También se puede administrar infraestructura de la misma manera que administra aplicaciones. Al aprovechar las metodologías de Docker para enviar, probar e implementar el código rápidamente, se puede reducir significativamente la demora entre escribir el código y ejecutarlo en producción.

Docker brinda la capacidad de empaquetar y ejecutar una aplicación en un entorno aislado llamado contenedor. El aislamiento y la seguridad le permiten ejecutar muchos contenedores simultáneamente en un host determinado. Los contenedores son livianos y contienen todo lo necesario para ejecutar la aplicación, por lo que no se necesita depender de lo que está instalado actualmente en el host. Es posible compartir contenedores fácilmente mientras se trabaja y asegurar de que todas las personas con las que se comparte obtengan el mismo contenedor que funciona de la misma manera.

5.0.1.5 Vagrant

Vagrant es una herramienta para construir y administrar entornos de máquinas virtuales en un solo flujo de trabajo [61]. Con un flujo de trabajo fácil de usar y un enfoque en la automatización, Vagrant reduce el tiempo de configuración del entorno de desarrollo, aumenta la paridad de producción y hace que la excusa "funciona en mi máquina" sea una reliquia del pasado.

Vagrant proporciona entornos de trabajo fáciles de configurar, reproducibles y portátiles construidos sobre la base de la tecnología estándar de la industria y controlados por un único flujo de trabajo coherente para ayudar a maximizar la productividad y la flexibilidad.

Para lograr su objetivo, Vagrant se para sobre los hombros de gigantes. Las máquinas se aprovisionan sobre VirtualBox, VMware, AWS o cualquier otro proveedor. Luego, las herramientas de aprovisionamiento estándar de la industria, como *scripts* de *shell*, Chef o Puppet, pueden instalar y configurar automáticamente el software en la máquina virtual.

5.0.2 Construcción de las pruebas

La topología en sí, así como todas las pruebas, fueron desarrolladas sobre la herramienta de emulación CORE en su versión 8.1.0 que provee mecanismos de creación de nodos de tipo *hubs*, *switchs*, *routers* y *hosts*, con sus propios servicios ejecutándose, como también los respectivos enlaces que los interconectan entre sí. La misma soporta los protocolos de ruteo dinámico mediante el uso de la herramienta FRRouting, siendo el protocolo principal BGP, y se ha utilizado para simular la interconexión realista de Internet, dada mayormente por la combinación de IP y BGP.

Para crear el entorno de pruebas se trabajó utilizando CORE sobre una máquina virtual (VM por sus siglas en inglés) creada mediante Vagrant, que provee la capacidad de definir las herramientas, archivos, paquetes, etc., necesarios para la instalación y funcionamiento adecuado del entorno. El proyecto está alojado en un repositorio ubicado en `https://github.com/cristianbarbaro/vagrant-coreemu`. Allí se observa dentro del archivo **Vagrantfile** la definición de las instalaciones para la máquina virtual basada en la *box* (definición de máquina dada por Vagrant) de Ubuntu Focal 20.04 LTS de 64 bits. En ese mismo archivo, al levantar la máquina virtual se define la ejecución del archivo **route.sh** que crea las entradas en la tabla de ruteo del anfitrión (máquina sobre la cual se ejecutan las máquinas virtuales) para lograr, entre otras cosas, llegar a través del navegador del mismo a visualizar el sitio web del WebScrub ejecutado dentro de la VM en el emulador CORE.

Durante el proceso de creación de la topología se ha observado ciertas fallas del funcionamiento de CORE, por lo que se procedió a realizar aportes a la comunidad del software libre a través de la creación de los llamados *pulls requests* (PR), que son aportes sobre proyectos de *git*. Al momento de escribirse estas líneas, uno de ellos ya ha sido corregido en el repositorio oficial de CORE, pero otros dos aún no, por lo que se procedió a agregar a través de la creación de la máquina virtual la modificación del código de CORE previa instalación así se aplican los parches y se trabaja sobre un sistema optimizado.

5.0.3 Aportes realizados al proyecto de CORE

Al utilizar CORE, se ha encontrado ciertos errores que fueron necesarios corregir y aplicado optimizaciones que mejora el funcionamiento de la herramienta, por lo que se procedió a realizar los siguientes *pulls requests* y discusiones de aportes:

- Nombre: Changed CoreNode inheritance and CoreInterface class call on physical interfaces #647
  - Descripción: Se encontró sobre los nodos de conexión RJ-45 virtuales del emulador que los mismos presentaban fallas de implementación y no funcionaban correctamente.
  - Tipo: Core Bug

  - Estado: El fix fue tomado en cuenta para la versión 8.1.0 https://github.com/coreemu/core/releases/tag/release-8.1.0
  - Enlace: https://github.com/coreemu/core/pull/647

- Nombre: Wallpaper zoom performance improvement and redrawing nodes #666
  - Descripción: Se detectó una falla en el procesamiento de imágenes de fondo utilizadas que no permite el uso de imágenes grandes y acercamientos en profundidad. Con esta corrección se puede utilizar grandes imágenes y realizar acercamientos a cualquier escala.
  - Tipo: Pygui Bug
  - Estado: Aún en revisión
  - Enlace: https://github.com/coreemu/core/pull/666

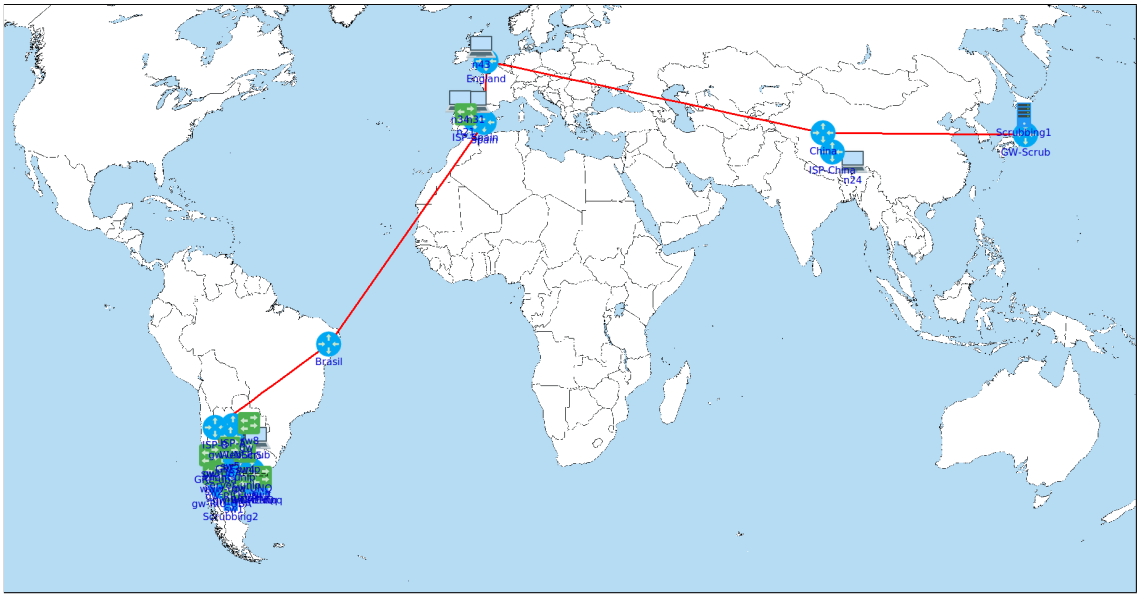
- Nombre: Node and label distance correct with redraw #674
  - Descripción: El nombre visible en la interfaz de CORE de los nodos se superpone o se aleja en distancias visuales de píxeles de los iconos. Se hizo fija la distancia en píxeles.
  - Tipo: Pygui Bug
  - Estado: Aún en revisión
  - Enlace: https://github.com/coreemu/core/pull/674

- Nombre: [FEATURE] Command line helper feature to work with the session #675
  - Descripción: Es un ejecutable creado en Bash para interactuar fácilmente con los elementos de la topología, así como también la creación automática de nombres en el archivo "/etc/hosts" al momento de iniciar y detener la topología, con los nombres de los nodos, para que los mismos sean accesibles en todo el entorno.
  - Tipo: Feature Discusion
  - Estado: Aún en discusión
  - Enlace: https://github.com/coreemu/core/discussions/675

5.0.4 Topología de pruebas

Se diseñó una topología en base a un mapa global para poder ejemplificar su utilización a gran escala en un entorno similar a uno real, donde los bloques de redes y ASNs elegidos para representar los nodos de la topología son los mismos que se observan realmente. De esta manera, se pretende lograr verosimilitud de geoposicionamiento que facilite la comprensión de la herramienta en su funcionamiento óptimo así como también para comprender las herramientas de monitoreo que se observarán en secciones posteriores. Del mismo modo, cabe destacar que es una demostración con políticas de enrutamiento definidas que no representan las reales de cada organización, por lo que en los próximos párrafos se explicarán ciertos detalles necesarios para comprender la topología y, además, no se llevó a cabo ninguna clase de ataque real. En la figura 5.1 se muestra el mapa completo realizado sobre la herramienta CORE.

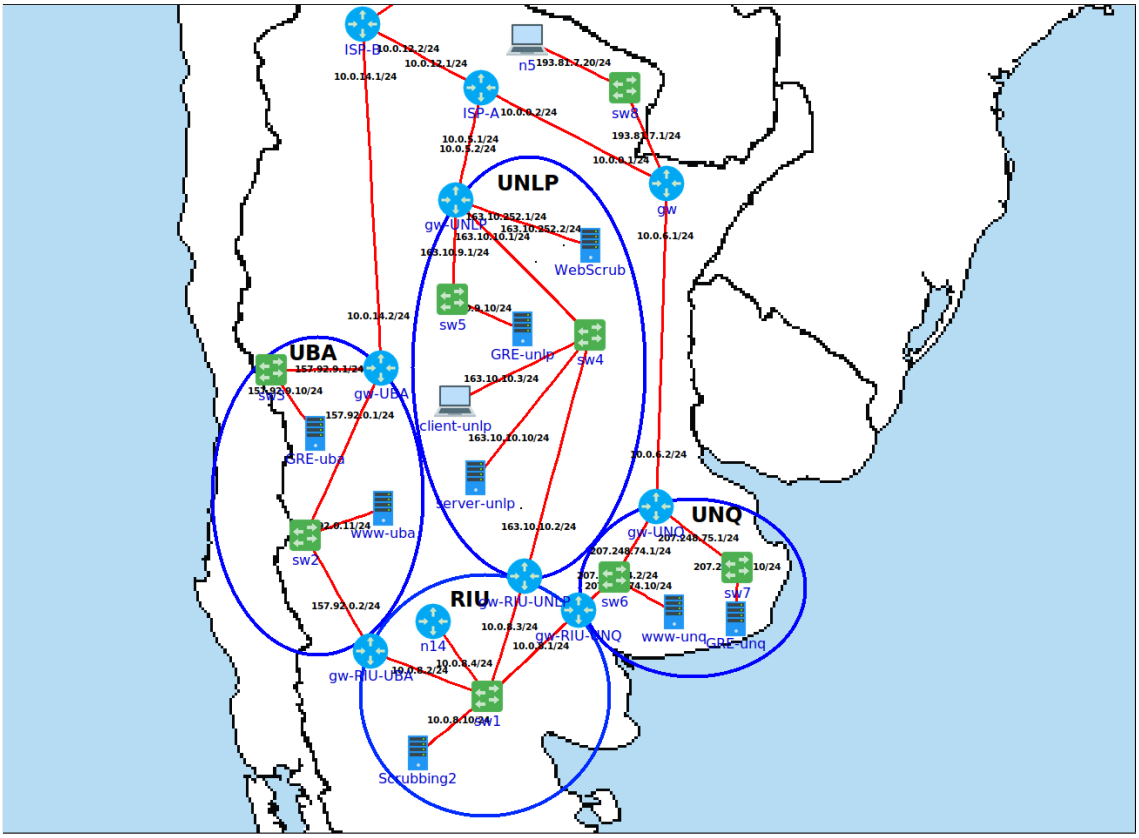
Figura 5.1: Vista global de la topología, en Argentina se ubican las organizaciones partícipes de ScrubbingUNLP



Se propone como organizaciones participantes de la plataforma a tres universidades locales que, elegidas intencionalmente, pertenecen a la Red de Interconexión Universitaria (RIU) de Argentina [62]. Estas universidades disponen de bloques de redes y ASNs propios, conectadas a sus ISPs y a su vez interconectadas entre ellas por la RIU, que en el caso de esta demostración actúa específicamente como un IXP y no provee de acceso a Internet a las universidades. En la figura 5.2 se muestra el mapa de los clientes del ScrubbingUNLP, junto a sus respectivos ISPs y la red de interconexión. Los datos de los clientes son:

- UNLP:
  - Red: 163.10.0.0/16
  - ASN: 5692
  - ASs Adjacentes: ISP-A, RIU
- UBA:
  - Red: 157.92.0.0/16
  - ASN: 3449
  - ASs Adjacentes: ISP-B, RIU
- UNQ:
  - Red: 207.248.74.0/23
  - ASN: 61486
  - ASs Adjacentes: ISP-A, RIU

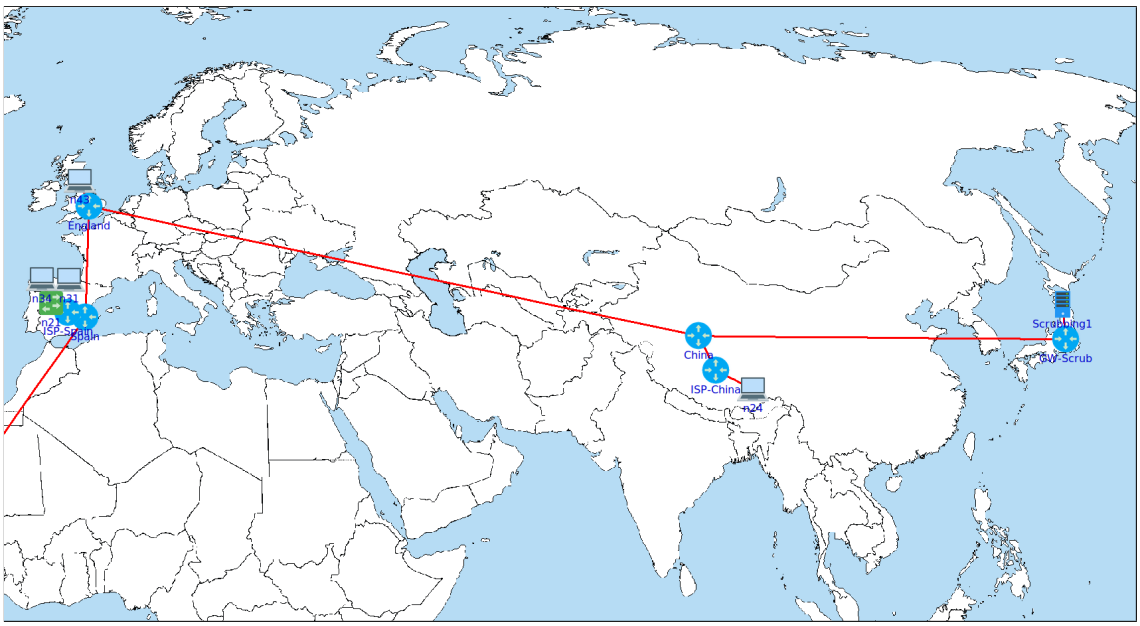
Figura 5.2: Interconexión de Argentina



Sobre la topología se dispone la red del ScrubbingUNLP lista para trabajar. En la figura 5.2 se observa dentro de la red de la UNLP el nodo WebScrub, que como su nombre indica, tiene la interfaz web disponible para que los clientes se conecten e interactúen con la plataforma del ScrubbingUNLP. También se puede observar en cada uno de los clientes la existencia de los nodos **GRE-uba**, **GRE-unlp** y **GRE-unq** necesarios para implementar las terminaciones de los túneles GRE de la **UBA**, la **UNLP** y la **UNQ**, respectivamente. Por último, en el grupo de dispositivos llamado RIU, que es el IXP en esta demo, existe un nodo llamado **Scrubbing2** que es uno de los múltiples *scrubbings centers* del ScrubbingUNLP capaces de recibir el tráfico, a partir de la publicación del ASN, como se ha mencionado previamente, y limpiarlo para enviarlo luego a la organización destino a través del túnel GRE. Para la topología propuesta, este *scrubbing center* sería el encargado de recibir y filtrar tráfico si el ataque proviene de uno de los AS miembros de la RIU.

Como se puede observar con más detalle en la figura 5.3, para el resto de la topología se dispone de otro nodo de *scrubbing center* llamado **Scrubbing1** ubicado en Japón. Este nodo, por su cercanía en la topología, sería capaz de atraer el tráfico proveniente de España, Inglaterra y China, destinado a cualquiera de las organizaciones miembros de ScrubbingUNLP.

Figura 5.3: Interconexión de Europa y Asia



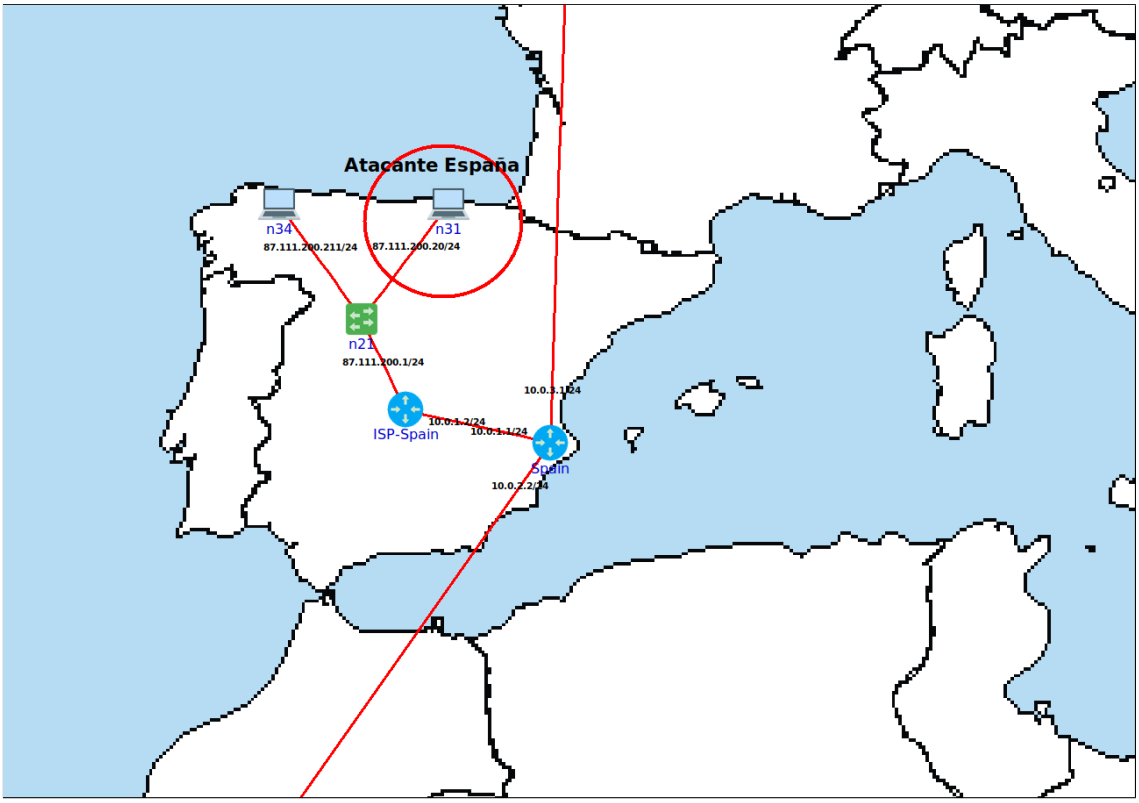
5.0.5 Simulación de un ataque

En esta sección se describirá cómo se implementa y cómo se lleva a cabo un ataque realizado por tres diferentes *hosts* de la topología hacia un servidor alojado dentro de la red de la UBA de manera detallada.

5.0.5.1 Implementación de un ataque de DDoS

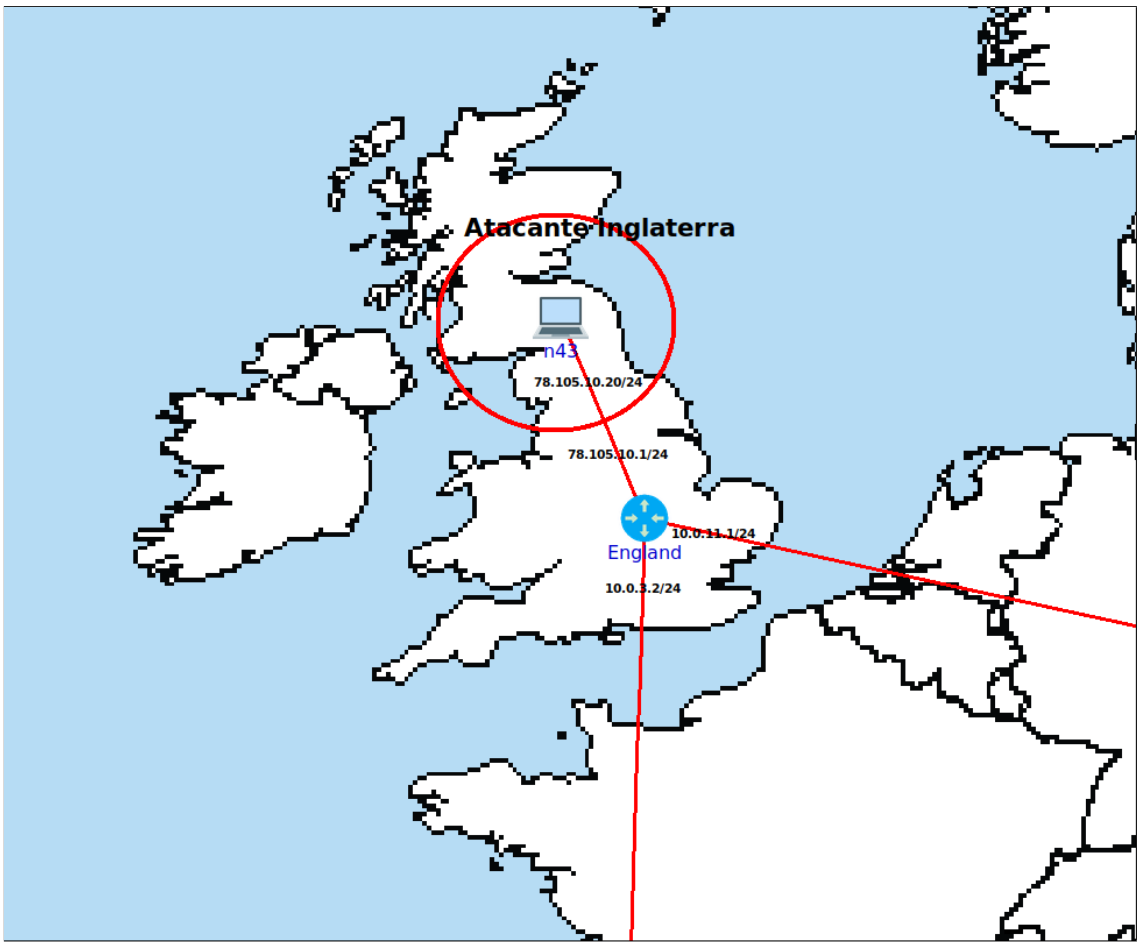
Los atacantes se hallan distribuidos geográficamente en diferentes locaciones de la topología de CORE. El primero de ellos es el nodo **n31** ubicado en la zona de España (ver figura 5.4) y está configurado con la dirección IP 87.111.200.20/24 que pertenece al ASN 3352. Este ASN es propietario de la red 87.111.0.0/16 y dicha red corresponde realmente a TELEFÓNICA DE ESPAÑA.

Figura 5.4: El nodo n34 ubicado en la región de España, es uno de los atacantes



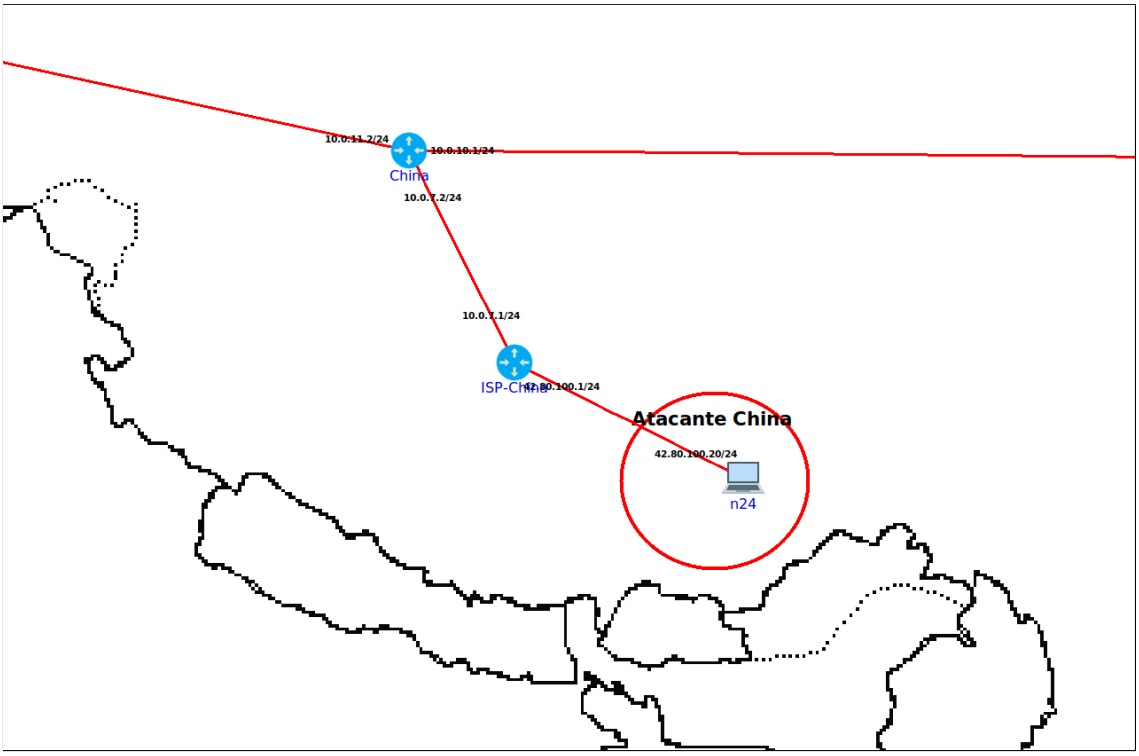
Otro de los nodos atacantes se encuentra ubicado en la zona de Inglaterra (ver figura 5.5), tiene asignado el nombre **n43** y tiene configurado la dirección IP 78.105.10.20/24. Este bloque de red pertenece al ASN 5607, correspondiente a SKY UK Limited y propietario del bloque de red 78.105.0.0/16.

Figura 5.5: El nodo n43 ubicado en la región de Inglaterra, es otro de los atacantes



El tercer atacante, llamado n24, se encuentra ubicado en China (ver figura 5.6) y tiene configurada la dirección IP 42.80.100.20/24 que pertenece al ASN 17638, correspondiente a TIANJIN Provincial Net of CT y propietario del bloque de red 42.80.0.0/16.

Figura 5.6: El nodo n24 ubicado en la región de China, es también uno de los atacantes



De forma coordinada, estos dispositivos enviarán tráfico hacia el servidor llamado **www-uba**, ubicado dentro de la red de la universidad de la UBA como puede verse en la figura 5.7 y configurado con la IP 157.92.0.11/24. Además tiene configurado un servicio web que escucha en el puerto 80/TCP.

En la figura 5.8 se muestra la página web de la víctima construida a partir de un HTML muy sencillo que posee una etiqueta *img* con referencia a una imagen dentro del mismo servidor de aproximadamente 1,6 Megabytes de tamaño. Esta imagen provoca que, cuando se haga una petición HTTP al servidor por esta página web, exista cierta carga de red durante el acceso al sitio web debido al incremento de tráfico.

El tamaño de la imagen y el HTML en su totalidad generan un tráfico de red muy leve a lo que un sitio suele ofrecer, del mismo modo respecto a la carga de procesamiento del servidor web que es prácticamente nula por ser un sitio plano. Esto ocurre en un comportamiento normal por parte de los usuarios.

Figura 5.7: El servidor www-uba es la víctima del ataque en cuestión

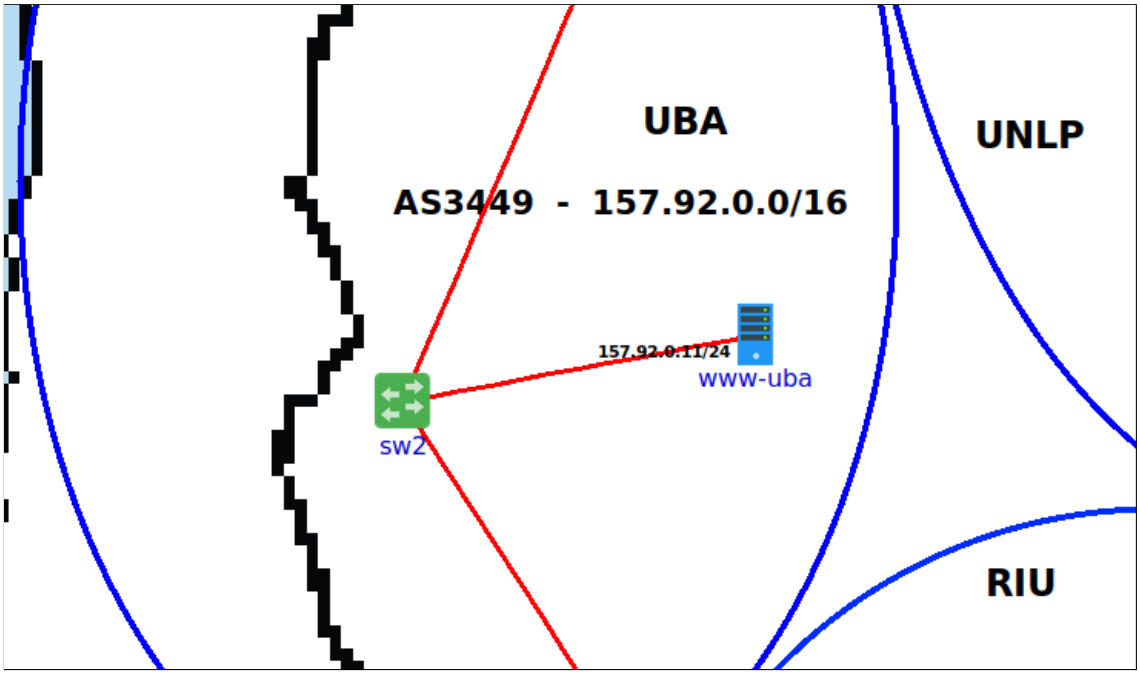
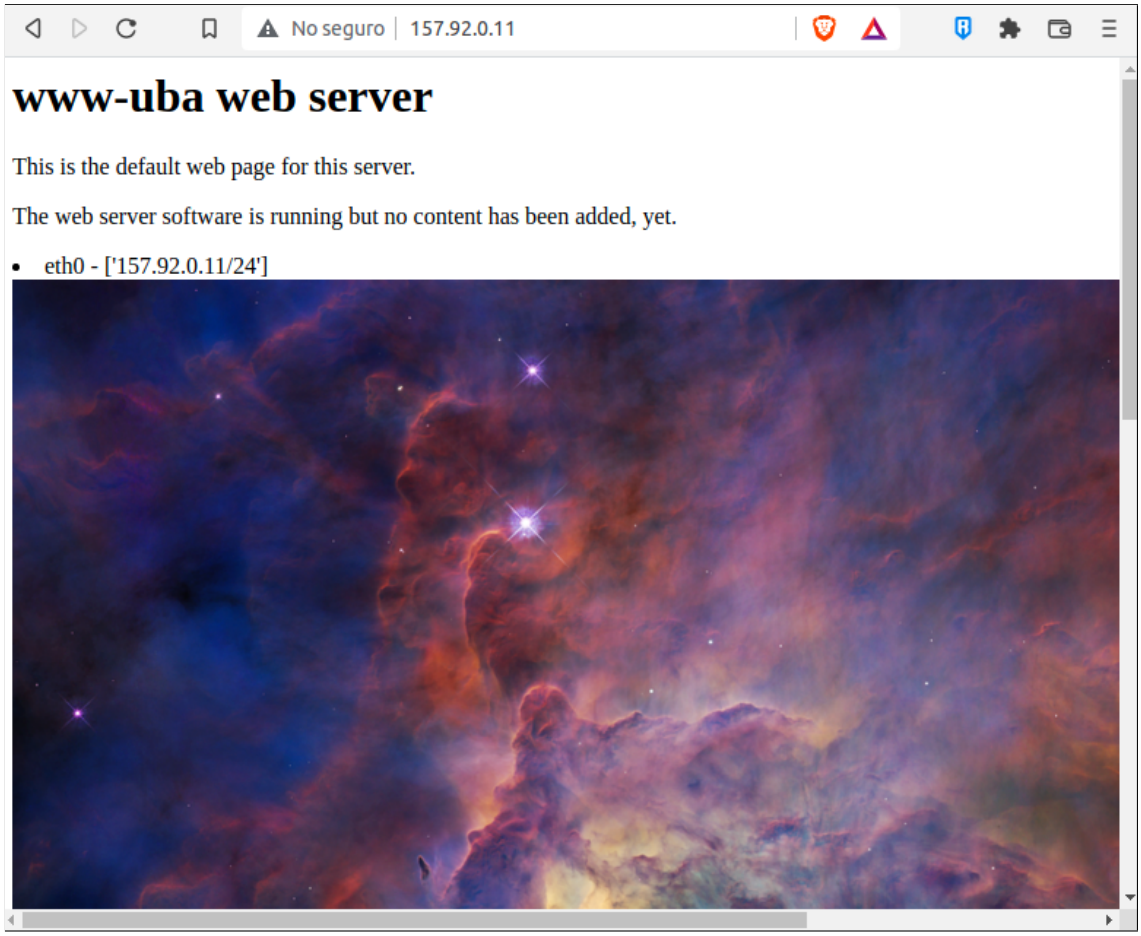


Figura 5.8: Sitio web del servidor víctima



Para llevar a cabo este ataque de denegación de servicio, se utilizará la herramienta **wget**. Esta herramienta realiza peticiones HTTP a servidores web y descarga sus recursos web en una carpeta del cliente. El ataque consiste en crear una cierta cantidad de procesos que se conecten y soliciten la página de inicio de la víctima de manera paralela y continua. De esta manera, el servidor comenzará a saturarse al intentar resolver cada una de las peticiones solicitadas por los atacantes. A continuación se muestra el *script* que ejecutan los ataques utilizado para generar grandes cantidades de tráfico.

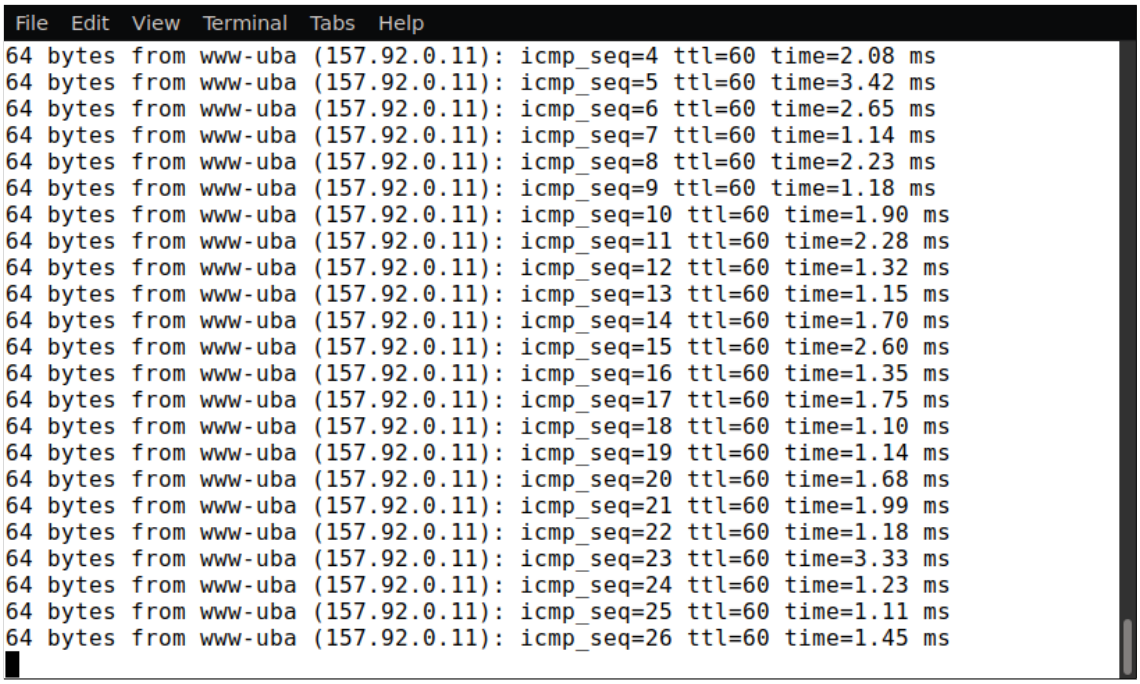
```
#!/bin/bash

for i in ‘seq 1 300‘; \
    do while true;
        do wget -r -q -k http://www-uba >/dev/null;
    done & \
done;
```

Este *script* tiene como finalidad crear hasta 300 procesos simultáneos que descargan el sitio web completo (con sus imágenes y recursos extras), y al finalizar una descarga, vuelve a iniciar un nuevo

proceso que nuevamente solicita el contenido de la página web completo. Esto se ejecuta constantemente, provocando que haya altas cantidades de tráfico en el servidor víctima.

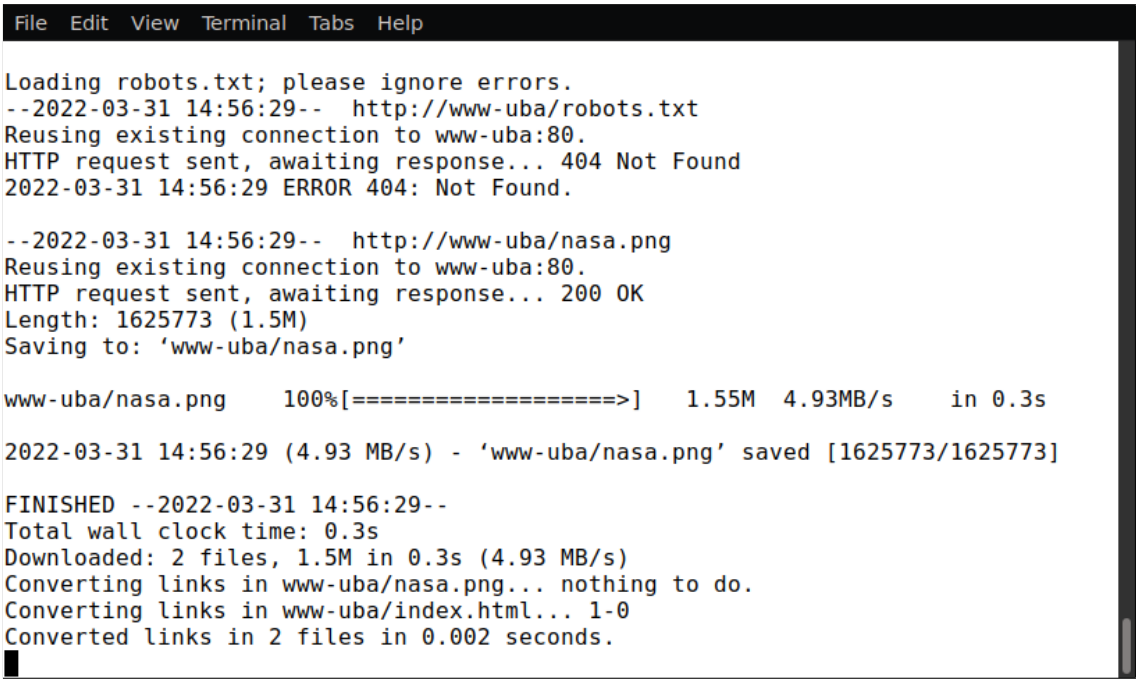
Por otro lado, el cliente **n5** se encuentra ejecutando un *ping* hacia el servidor víctima y permite ver el tiempo que tarda en obtener respuesta. Previamente al ataque el resultado del comando ping ronda alrededor de un milisegundo, como puede observarse en la figura 5.9.

Figura 5.9: Salida de comando ping de cliente n5 antes de iniciar el ataque DDoS



También, uno de los nodos ubicados en España, el nodo **n34**, se encuentra realizando peticiones HTTP al servidor de la UBA. En la figura 5.10 se puede ver la salida del comando curl que se ejecuta en un bucle pero manteniendo un intervalo de espera para simular lo mejor posible las acciones de un usuario cliente final. De esta manera se simula tráfico legítimo en la red de la demostración. Se observa que el tiempo que tarda en completarse una petición al servidor de la UBA ronda en los 0.3 segundos.

Figura 5.10: Peticiones HTTP realizadas por el cliente n34



Por otro lado, se cuenta con un *script* que realiza peticiones web al servidor víctima y calcula el tiempo que demora desde que se realiza la petición HTTP hasta que se obtiene la respuesta a la misma. Este *script* se ejecuta en los nodos **n34** (España) y **n5** (Argentina). En la imagen 5.11 se observan que los tiempos en ambos nodos ronda en alrededor de 0.01 segundos en completarse la petición. La figura se realizó a partir del cálculo según la siguiente fórmula, donde 'time_connect' indica el tiempo hasta que se establece la conexión TCP, 'time_transfer' indica el tiempo desde que se establece la conexión hasta que se obtiene la respuesta completa del servidor y 'time_total' representa el tiempo total transcurrido:

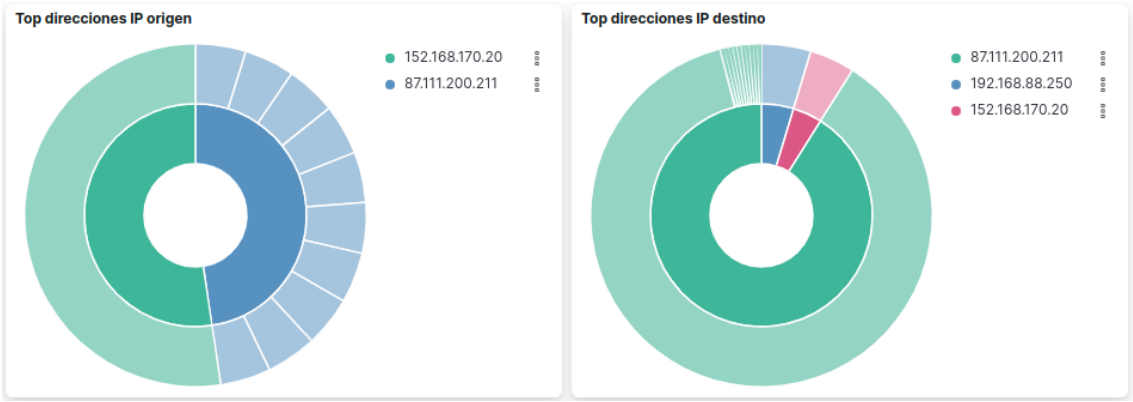
$$time\_connect + time\_transfer = time\_total$$

Figura 5.11: Tiempo que tarda el servidor víctima en devolver una página web antes de que ocurra el ataque

```
n34: 0.009225 + 0.002128 = 0.011353
n34: 0.024827 + 0.025364 = 0.050191
n5: 0.022996 + 0.036727 = 0.059723
n5: 0.011190 + 0.005467 = 0.016657
n34: 0.003424 + 0.004886 = 0.008310
n5: 0.025882 + 0.015376 = 0.041258
n5: 0.003851 + 0.019338 = 0.023189
n5: 0.017827 + 0.005091 = 0.022918
n34: 0.005060 + 0.025852 = 0.030912
n34: 0.003702 + 0.011248 = 0.014950
n34: 0.011041 + 0.008640 = 0.019681
n34: 0.002173 + 0.010183 = 0.012356
n5: 0.004098 + 0.004669 = 0.008767
n34: 0.004027 + 0.004015 = 0.008042
n34: 0.003536 + 0.002605 = 0.006141
n5: 0.001603 + 0.002752 = 0.004355
n5: 0.035135 + 0.007515 = 0.042650
n5: 0.006467 + 0.010408 = 0.016875
n5: 0.004503 + 0.002946 = 0.007449
n34: 0.007910 + 0.006976 = 0.014886
n5: 0.011878 + 0.003377 = 0.015255
n34: 0.005368 + 0.006444 = 0.011812
n5: 0.007827 + 0.021862 = 0.029689
n5: 0.039497 + 0.013118 = 0.052615
n5: 0.002463 + 0.003501 = 0.005964
n5: 0.005429 + 0.003354 = 0.008783
n5: 0.012656 + 0.008352 = 0.021008
n34: 0.006359 + 0.013329 = 0.019688
```

Para poder visualizar de una manera gráfica los eventos que ocurren en la red, se utilizan herramientas de monitoreo y gráficos en Kibana que permiten obtener una idea general de qué ocurre. Este tráfico está siendo monitoreado en el *router* de borde del sistema autónomo de la UBA, **gw-UBA**, por ende, se visualiza todo lo que entra o sale de la red de la UBA. La figura 5.12 representan el *top* de direcciones IP de origen y destino de todo el tráfico analizado en los últimos 30 minutos. Puede observarse que las direcciones IP que predominan son las direcciones de los nodos que tienen ejecutando los comandos ping y curl constantemente (la dirección IP 87.111.200.211 pertenece al nodo **n34** y la dirección IP 152.168.170.20 pertenece al nodo **n5**). En estos gráficos se decide omitir la dirección IP del servidor **www-uba** para que sea más fácil visibilizar los clientes externos. Sin embargo, es necesario tener en cuenta que está también dentro del tráfico procesado en estos gráficos.

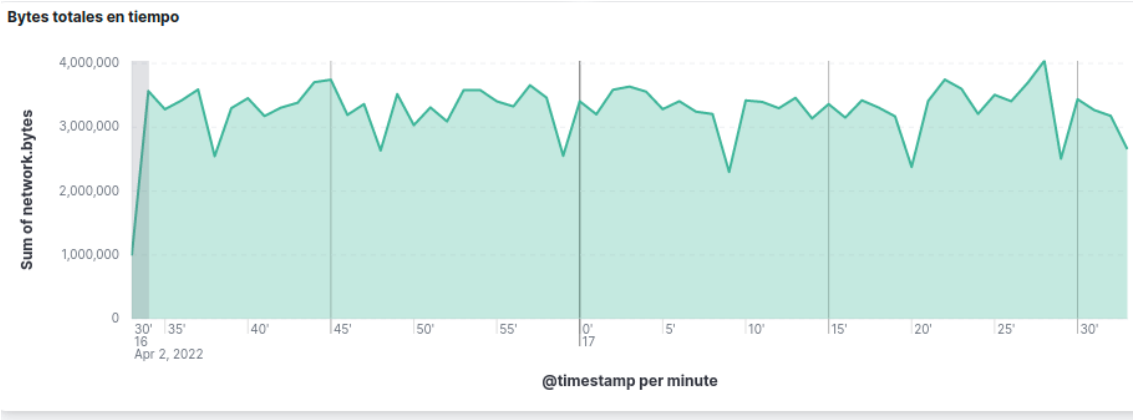
Figura 5.12: Top de direcciones IP observadas sin considerar la dirección del servidor www-uba



En el gráfico 5.13 se puede contemplar la cantidad de tráfico medido en bytes que atraviesan el *router*

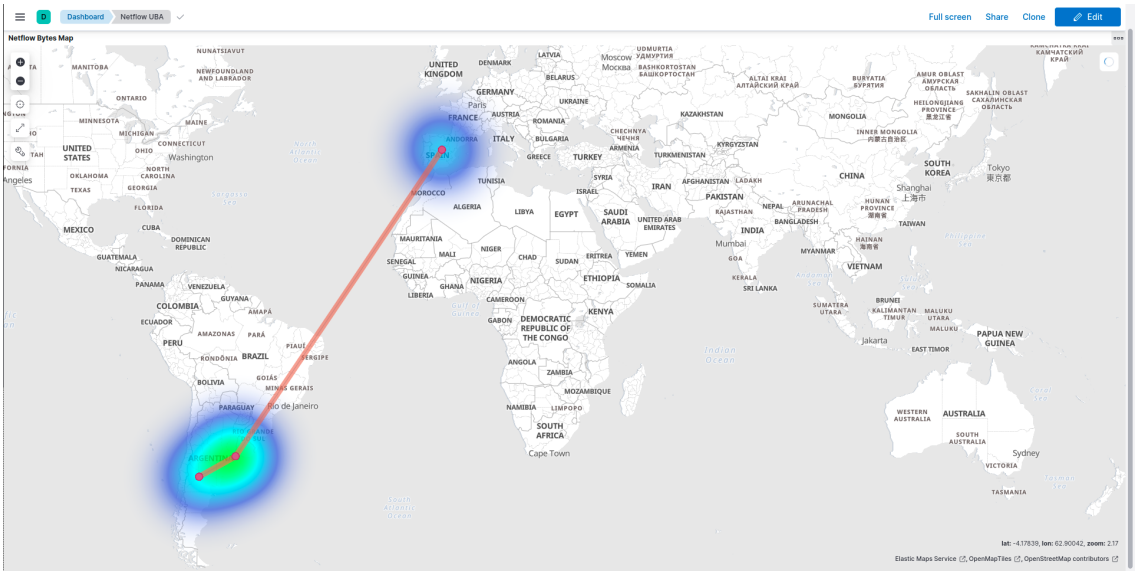
de borde de la UBA cada 30 segundos en los últimos 30 minutos. Se puede observar que los picos de tráfico están alrededor de los 3.000.000 bytes antes de que ocurriese el ataque y los mismos se mantienen dentro de esos valores. Este gráfico representa el funcionamiento normal y estable del flujo de tráfico de la red monitoreada.

Figura 5.13: Cantidad de tráfico observado en el tiempo



En el gráfico de la imagen 5.14 se observa la cantidad de volumen analizado basado en las geolocalizaciones de las direcciones IP vistas. En este caso, se observa que una de las zonas más vistas está en España, esto es debido al tráfico generado por el nodo **n34** y el otro punto se encuentra cerca del nodo víctima, **www-uba**, el nodo **n5**, ubicado también en Argentina. En secciones posteriores se observará cómo estos valores irán cambiando según la etapa en la que se encuentre la topología.

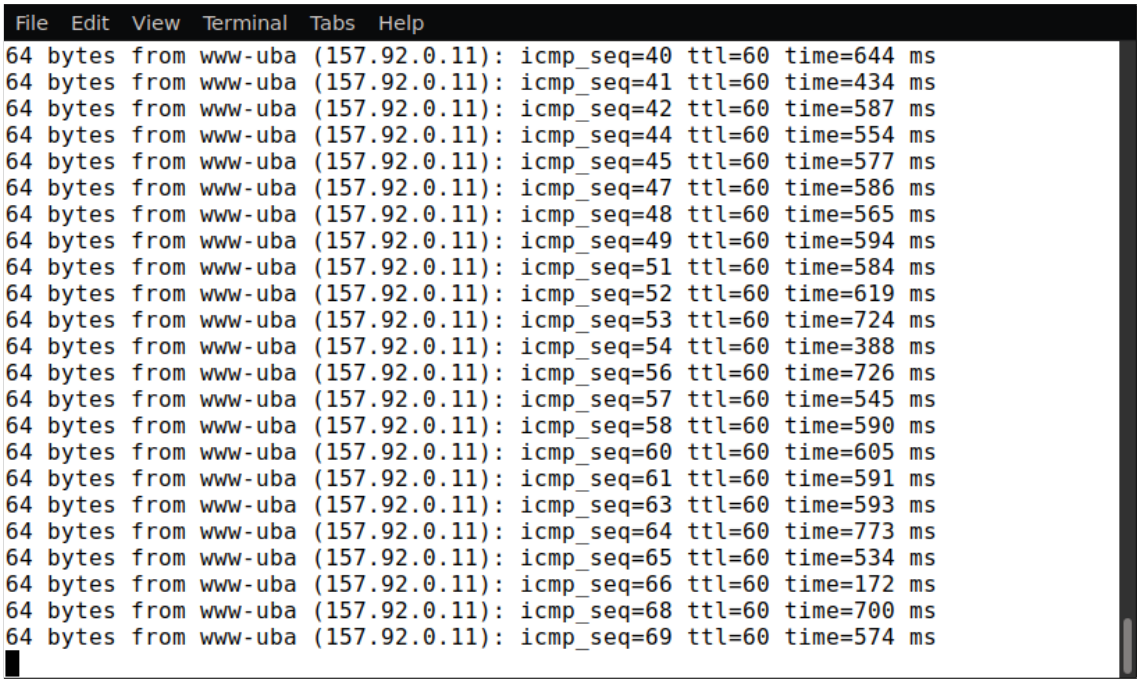
Figura 5.14: Cantidad de volumen de tráfico observado según la geolocalización antes de iniciar el ataque DDoS



5.0.5.2 Ejecución de un ataque de DDoS

Una vez que se inicia el ataque inmediatamente los tiempos de respuesta en el ping del cliente **n5** se ve afectado aumentando considerablemente, pasando de un tiempo de respuesta de alrededor de **1 ms** a un tiempo de respuesta de **500 ms** en promedio, como se puede observar en la figura 5.15.

Figura 5.15: Salida del ping del cliente durante el ataque



También, si se mira el *script* que calcula los tiempos necesarios para obtener una página web por parte de los clientes **n5** y **n34**, se notará que los tiempos necesarios para que el servidor web resuelva la tarea ahora ronda alrededor de los 60 segundos (ver figura 5.17).

Figura 5.16: Tiempos en segundos de respuesta del servidor web a los clientes durante el ataque, con valores cercanos al minuto

```
n34: 0.551745 + 53.083294 = 53.635039
n5: 0.561836 + 53.268703 = 53.830539
n34: 1.544219 + 51.709081 = 53.2533
n34: 1.556118 + 54.364639 = 55.920757
n34: 0.57829 + 52.742578 = 53.320868
n5: 0.556681 + 53.436090 = 53.992771
n5: 7.733396 + 55.763870 = 63.497266
n5: 0.566568 + 56.339869 = 56.906437
n34: 3.574554 + 54.757330 = 58.331884
n5: 7.651038 + 59.542072 = 67.19311
n34: 7.742056 + 58.550011 = 66.292067
n34: 7.625007 + 53.563360 = 61.188367
n34: 7.817222 + 53.914061 = 61.731283
n5: 3.568997 + 52.806735 = 56.375732
n5: 16.026754 + 55.372925 = 71.399679
n34: 0.555832 + 67.972080 = 68.527912
n34: 1.563242 + 61.214800 = 62.778042
n5: 0.533384 + 52.959735 = 53.493119
n5: 0.555181 + 58.647538 = 59.202719
n34: 7.711118 + 61.728973 = 69.440091
n5: 1.541232 + 67.312122 = 68.853354
n5: 7.771965 + 55.512563 = 63.284528
n5: 15.992103 + 66.216883 = 82.208986
n5: 0.558024 + 54.240277 = 54.798301
n5: 0.543184 + 53.190919 = 53.734103
n34: 31.965109 + 53.352757 = 85.317866
n34: 0.560267 + 54.166160 = 54.726427
n34: 0.540477 + 55.596642 = 56.137119
n5: 0.54302 + 80.457165 = 81.000185
n34: 7.676893 + 60.117938 = 67.794831
```

Por otro lado, se puede observar cómo los tiempos de descarga de la página web desde el cliente n34 pasó de 0.3 segundos a 43 segundos aproximadamente y una velocidad de descarga de 5MB/s a 87KB/s aproximadamente (ver figura 5.17).

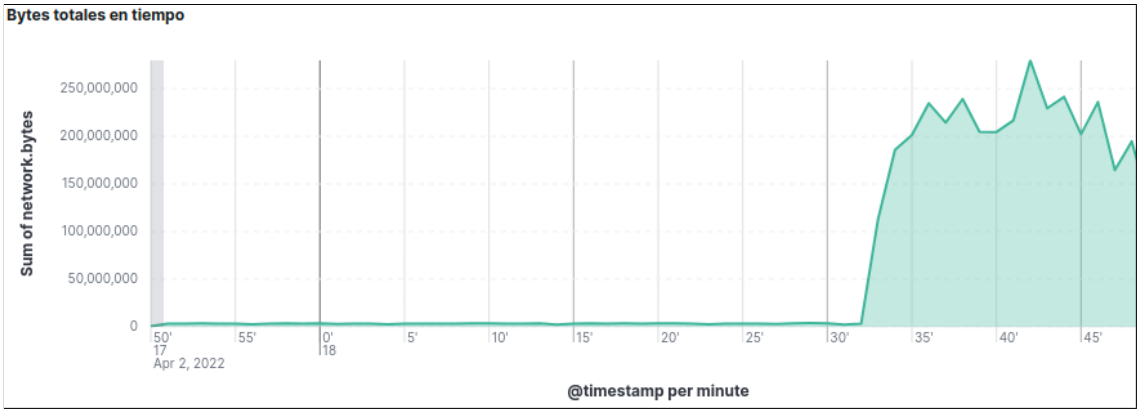
Figura 5.17: Salida del comando curl durante el ataque

```
File  Edit  View  Terminal  Tabs  Help
Loading robots.txt; please ignore errors.
--2022-04-02 21:33:44--  http://www-uba/robots.txt
Reusing existing connection to www-uba:80.
HTTP request sent, awaiting response... 404 Not Found
2022-04-02 21:33:45 ERROR 404: Not Found.

--2022-04-02 21:33:45--  http://www-uba/nasa.png
Reusing existing connection to www-uba:80.
HTTP request sent, awaiting response... 200 OK
Length: 1625773 (1.5M)
Saving to: ‘www-uba/nasa.png’

www-uba/nasa.png    100%[===================>]   1.55M  86.5KB/s    in 43s

2022-04-02 21:34:30 (36.7 KB/s) - ‘www-uba/nasa.png’ saved [1625773/1625773]

FINISHED --2022-04-02 21:34:30--
Total wall clock time: 2m 2s
Downloaded: 2 files, 1.5M in 43s (36.7 KB/s)
Converting links in www-uba/nasa.png... nothing to do.
Converting links in www-uba/index.html... 1-0
Converted links in 2 files in 0.008 seconds.
--2022-04-02 21:34:45--  http://www-uba/
Resolving www-uba (www-uba)... 157.92.0.11
```

Estos resultados están dados por la cantidad de dispositivos realizando el ataque, si se incrementara la cantidad de atacantes el ataque sería incluso más efectivo. Y, por otro lado, no aparenta en primera instancia ser una denegación de servicio ya que el servicio aún funciona, pero los tiempos de respuestas que se obtienen a partir de este momento, en muchos sistemas podrían generar un *timeout* o ser muy difícil de trabajar para un usuario que deba esperar casi un minuto para cargar cada imagen que la página posea (tener en cuenta que esta página web de pruebas solo cuenta con una imagen). De esta manera se puede observar cómo el servicio se ve degradado cuando el ataque comienza a ejecutarse.

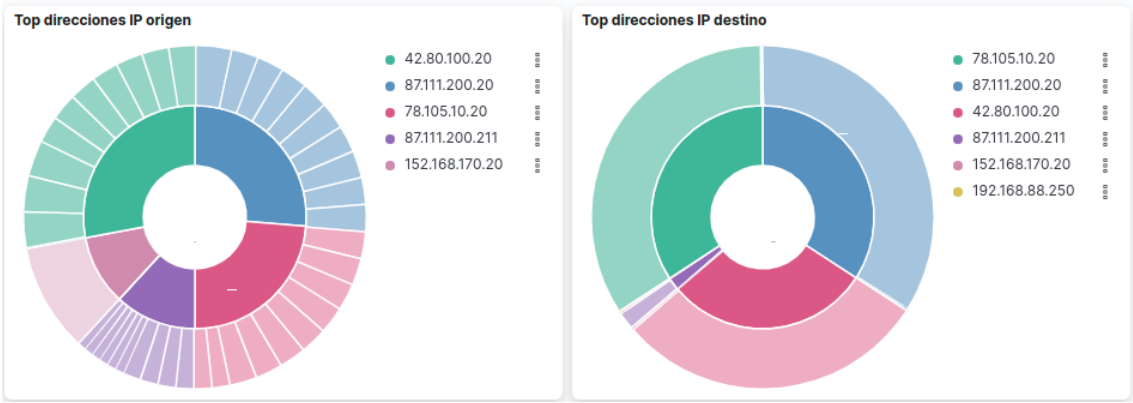
En el gráfico de la figura 5.18 se puede visualizar cómo la cantidad de tráfico medida en bytes se ha incrementado abruptamente a valores que van de los 3.000.000 bytes normales iniciales a alrededor de 250.000.000 bytes.

Figura 5.18: El tráfico observado en el tiempo se ve incrementado de manera abrupta cuando se inicia el ataque DDoS



En la figura 5.19 se observa que las direcciones IP que comienzan a predominar los *tops*, tanto de direcciones origen como de destino, son las direcciones IP de los nodos atacantes, **n24**, **n31** y **n43**.

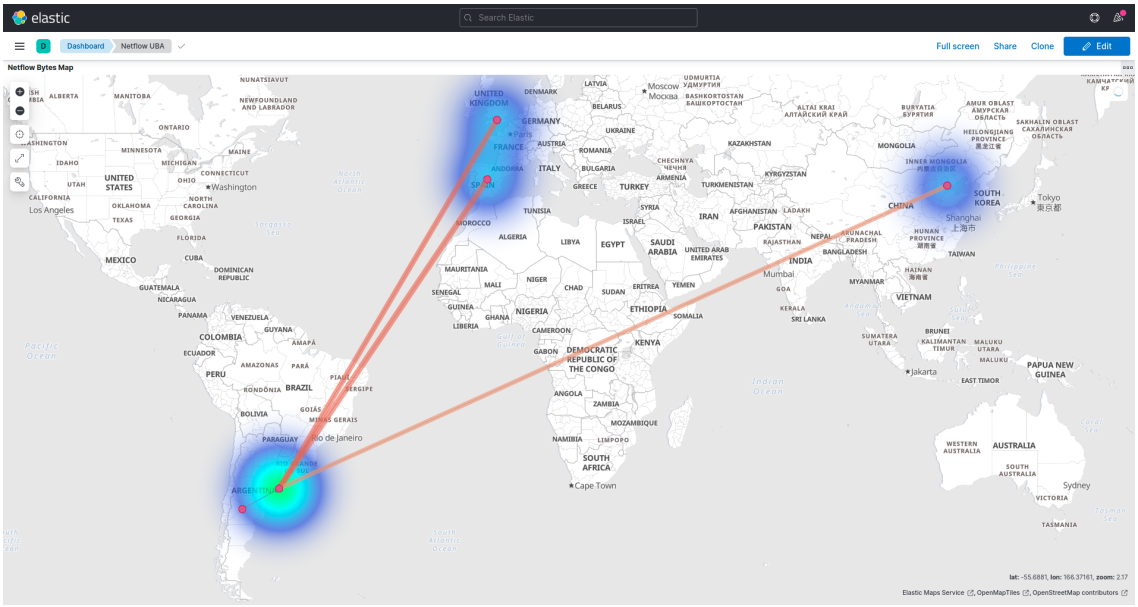
Figura 5.19: Las direcciones IP que predominan ambos *tops* corresponden a los tres atacantes



En el siguiente gráfico se observa el volumen de tráfico ubicados en el mapa por geolocalización. Estos puntos se encuentran localizados en España, Inglaterra y el este de China. Las líneas que unen las zonas representa la cantidad de flujo de tráfico que van desde un punto al otro. La mancha coloreada representa

también a la cantidad de tráfico procesado por los nodos en esas zonas, es por ello que la zona de Argentina, donde se encuentra el nodo afectado, es mucho más intensa ya que procesa todo el tráfico recibido por los tres atacantes.

Figura 5.20: Mapa donde se observa el gran volumen provocado por los atacantes



5.1 Funcionamiento de ScrubbingUNLP

Como se ha mostrado en la sección 5.0.5 se puede, a través de solo algunos equipos que entre todos sumen mayor tráfico que el que sea capaz de procesar el enlace de la víctima, lograr producir un ataque de denegación de servicio exitoso, por lo que a continuación se verá cómo mitigar el mismo utilizando ScrubbingUNLP y cómo se ve afectado el tráfico legítimo y el del ataque.

5.1.1 Mitigación del ataque

5.1.1.1 Detección

El usuario del sistema que recibe el ataque podrá ser advertido del mismo a través de diferentes métodos, estos pueden ser desde una herramienta de monitoreo y alarmas de tráfico y sobrecarga en dispositivos de red, por alarmas de sobrecarga en los mismos servidores, o hasta por un aviso personal de un usuario del servicio que no puede acceder. Los primeros mecanismos de advertencia son ejemplificados a través del procesamiento del tráfico utilizando NetFlow ubicado en el *router* de borde de la UBA en la topología, y luego enviado los resultados a Elasticsearch y graficados a través del Kibana, como se mostró en las secciones previas.

Cualquiera de estos métodos de detección serán suficientes para advertir de un problema y optar entre diferentes alternativas para solucionarlo. Es en este momento que, a partir de que el usuario bajo ataque comprende la situación de que el mismo proviene del exterior de su Sistema Autónomo, que atraviesa su ISP, que es un ataque de tipo volumétrico y que el ataque proviene desde múltiples locaciones de

Internet, será necesario mitigar el ataque hacia la organización a través de la limpieza de tráfico malicioso utilizando ScrubbingUNLP.

5.1.1.2 Elección de filtros

El cliente deberá analizar el tipo de ataque y analizar las características que lo identifican, es decir, determinar el puerto o puertos origen y destino, el protocolo o los protocolos y la dirección IP o las direcciones IPs origen y destino. A partir de este conjunto de datos se podrá determinar con más o menos precisión el ataque. En un primer caso, si se eligen identificadores de tráfico que agrupen a más objetivos se verán afectados usuarios legítimos con imposibilidad de acceder. En un segundo caso, contrario al anterior, el cliente podría elegir identificadores de tráfico que agrupen pocos flujos y esto podría provocar que el ataque no se logre mitigar por completo.

Para el primer caso, dejar usuarios legítimos fuera del alcance de los servicios tal vez sea la única opción. Por ejemplo, si estos usuarios se encuentran dispersos y mezclados en las mismas redes que producen el ataque, lamentablemente para ellos no habrá una buena respuesta del sistema pero será necesario dejarlos afuera para que aquellos usuarios legítimos fuera del rango de los atacantes puedan acceder. A lo sumo, aquí la regla de filtrado podría combinarse con una política de acción de tipo *rate limit*, para que eventualmente algunos de los usuarios que compartan el mismo bloque de red con los atacantes puedan llegar al mismo.

Para el segundo caso, filtrar el tráfico por debajo del flujo total del ataque puede que, a pesar de todo, sí logre mitigar la denegación de servicio. Esto se lograría gracias a que, si el volumen de tráfico bloqueado es igual o mayor a la diferencia entre el tráfico capaz de procesar por el destino y el legítimo de los usuarios, la mitigación tendrá éxito.

En la topología para lograr bloquear el ataque se elegirá tres conjuntos de identificadores diferentes ya que los ataques provienen de tres bloques de direcciones IP distintos. Estos identificadores dispondrán del protocolo TCP, ya que el ataque se efectúa contra un servicio HTTP, por el mismo motivo se elige el puerto 80 como destino, combinados a un bloque IP origen distinto para cada identificador.

Con cada uno de estos identificadores de tráfico se creará una regla FlowSpec, todo en el mismo formulario web, que definirá la acción a tomar, en este caso una acción de *deny* o denegar.

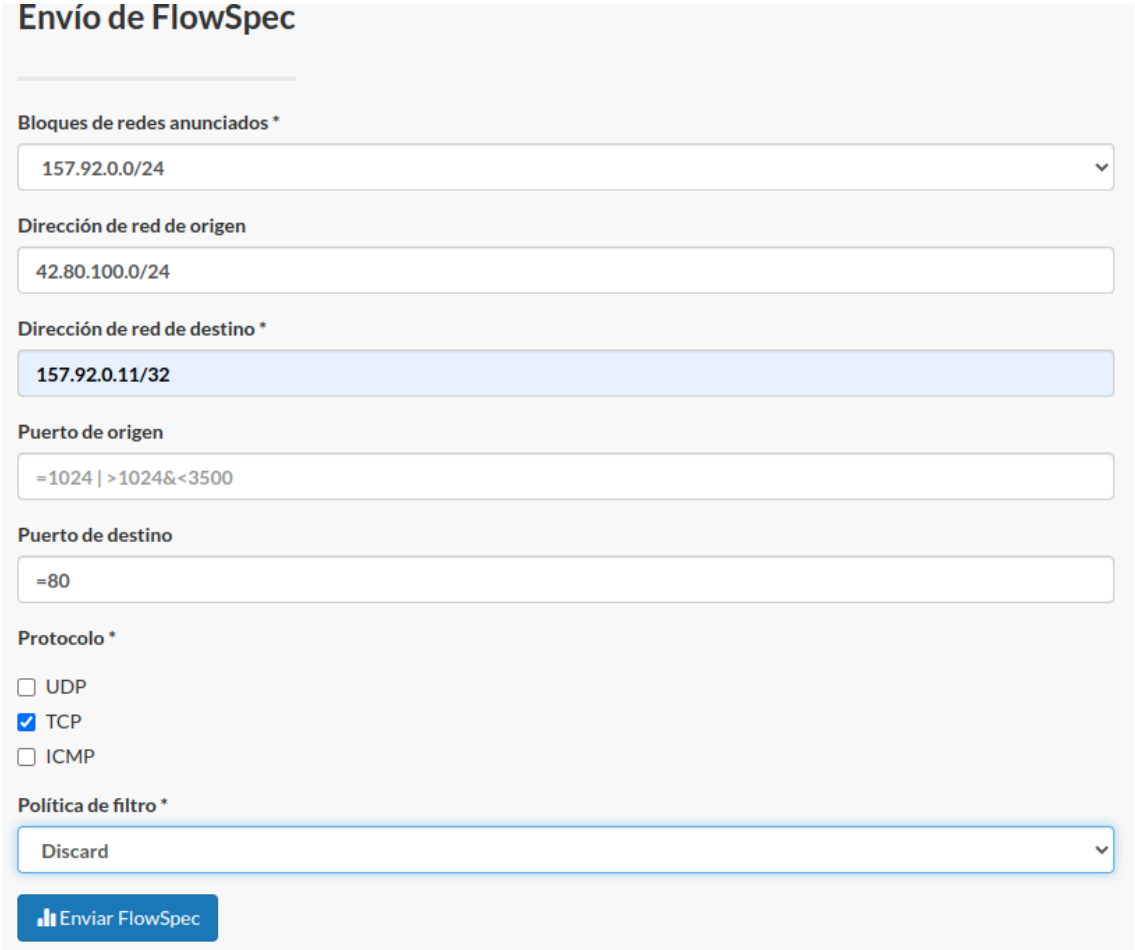
5.1.1.3 Publicar AS y aplicar filtros de limpieza

En la figura 5.21, a través de la interfaz de WebScrub, el cliente anunciará un bloque de red en el cual se encuentra el host afectado para su número de AS por BGP a través de los *scrubbing centers* para que los mismos reciban todo el tráfico cercano a ellos y sea retransmitido por cada uno de los *scrubbing centers* a través de los túneles GRE definidos en el mismo proceso de publicación, encaminando el tráfico por el mismo hacia la organización, evitando el *ruteo* de Internet.

Figura 5.21: Formulario de creación del anuncio BGP para el bloque de red atacado

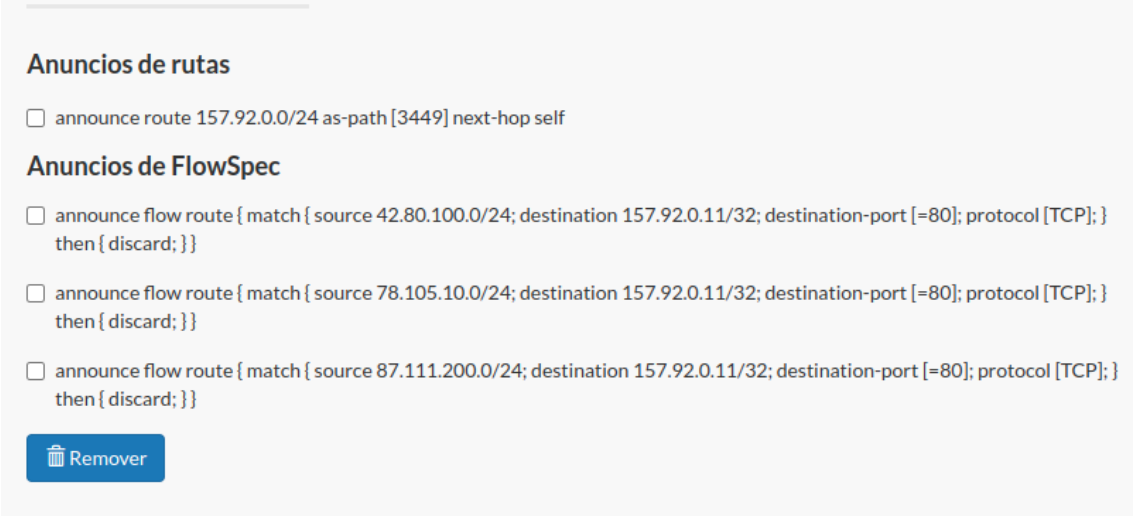
Como muestra la figura 5.22, a través del panel web de creación de FlowSpec, el cliente aplica los filtros mencionados en 5.1.1.2.

Figura 5.22: Formulario de ejemplo de creación de uno de los tres FlowSpec usados para bloquear los ataques. En la imagen se muestra la creación para el bloque IP origen 42.80.100.0/24 (China)



En la figura 5.23 se puede observar el listado de anuncios de rutas del WebScrub donde se visualizan todas las publicaciones realizadas para detener el ataque. Allí se pueden observar tanto la publicación del bloque de red afectado por BGP como las reglas de filtrado de tráfico mediante flowSpec.

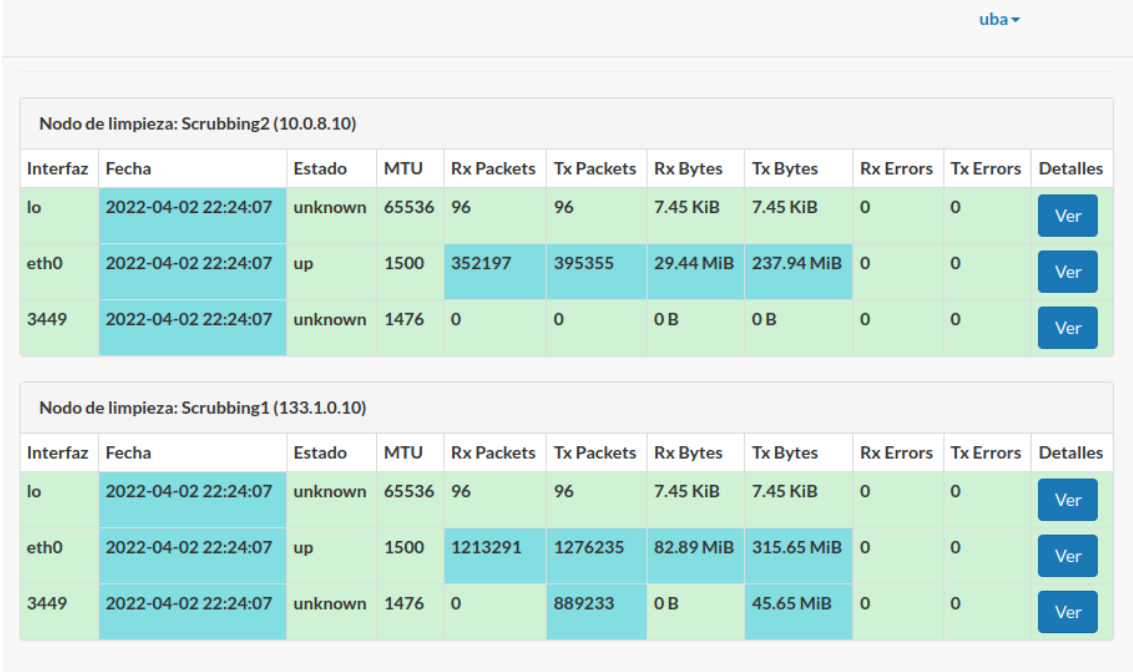
Figura 5.23: Listado de anuncios BGP y FlowSpec realizados en WebScrub



5.1.1.4 Visualización de la mitigación

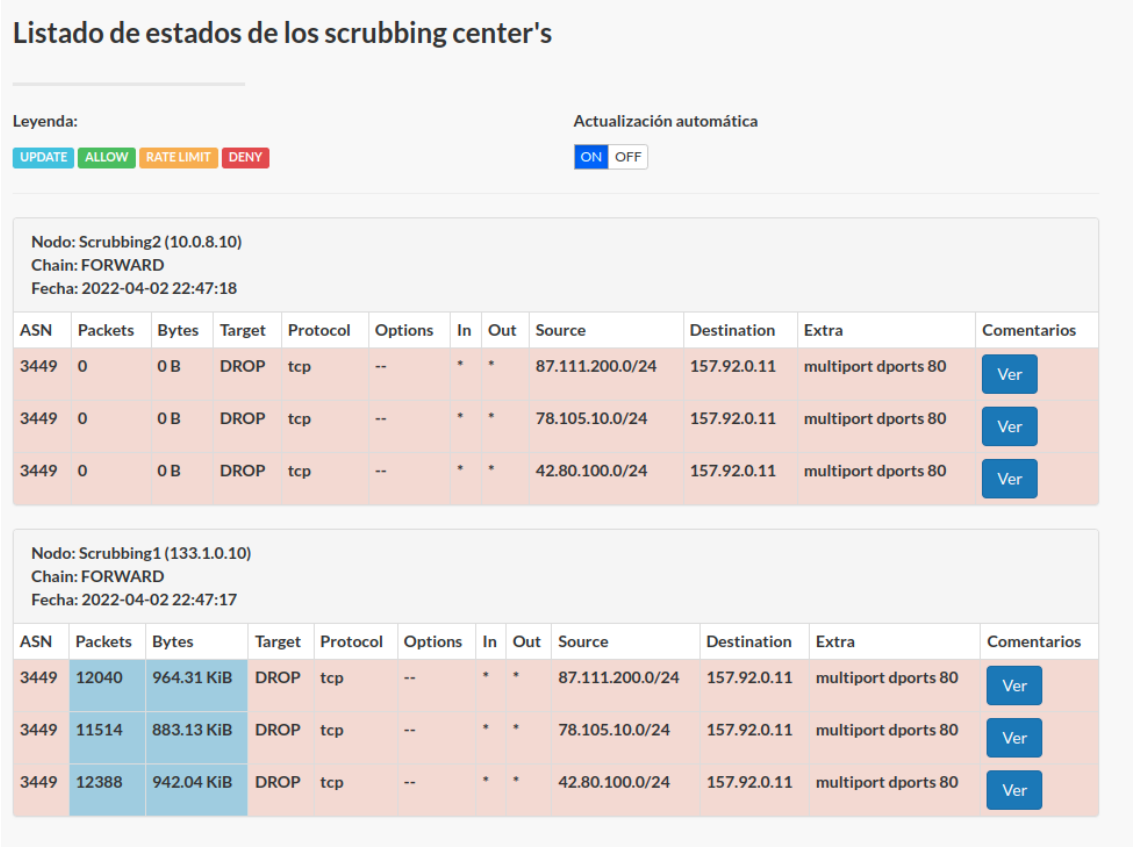
En esta instancia se ha logrado mitigar el ataque. Para ello se dispone de diferentes mecanismos. Por un lado se tienen las herramientas que proporciona el mismo ScrubbingUNLP, que es la interfaz web con el tráfico que atraviesa el túnel de los nodos de limpieza. Si se observa en el panel de **Interfaces en Nodos** se podrá observar que al pasar el tiempo las interfaces generales del *scrubbing center* que recibe tráfico, por ejemplo **eth0** incrementará, mientras que la interfaz del túnel, con nombre igual al ASN de la víctima, permanece igual o incrementa a menor velocidad, por ejemplo como muestra la figura 5.24. Esto indica que está ingresando demasiado tráfico al *scrubbing center* pero el mismo es procesado y filtrado si es malicioso. Es por esta razón que en la interfaz del túnel el incremento de bytes transmitidos es mucho menor (este es el tráfico limpio).

Figura 5.24: Interfaces de los *scrubbing centers* que aumentan por el tráfico que los atraviesa



A su vez, desde el panel de **Filtros Corriendo**, que posee las reglas de filtrado, se podrá observar que aquellas que coinciden con las reglas de FlowSpec definidas por el cliente incrementarán en el tiempo su número en Bytes, lo cual indica que están funcionando como se espera. Esto se muestra en la figura 5.25.

Figura 5.25: Filtros corriendo necesarios para detener el ataque, con las columnas coincidentes de filtrado incrementando su número de bytes señalado por el color azul



Listado de estados de los scrubbing center's

Leyenda: UPDATE ALLOW RATE LIMIT DENY

Actualización automática: ON OFF

Nodo: Scrubbing2 (10.0.8.10)
Chain: FORWARD
Fecha: 2022-04-02 22:47:18

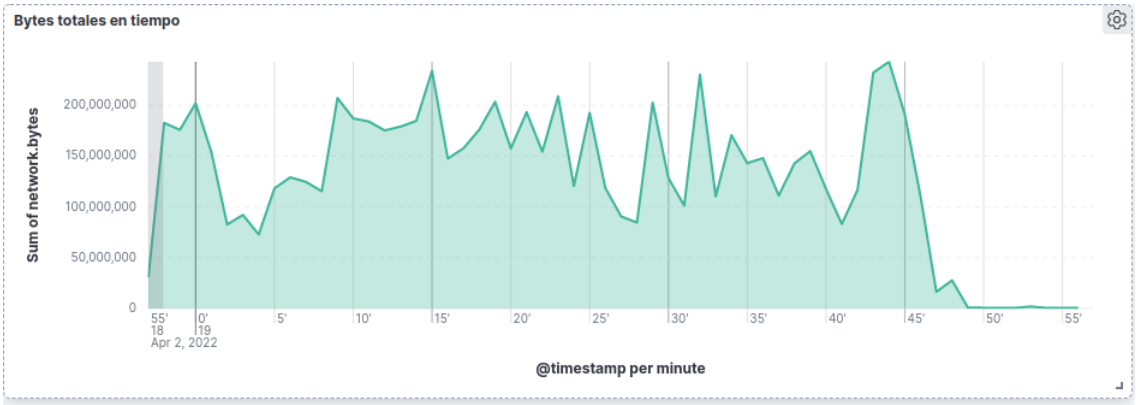
ASN	Packets	Bytes	Target	Protocol	Options	In	Out	Source	Destination	Extra	Comentarios
3449	0	0 B	DROP	tcp	--	*	*	87.111.200.0/24	157.92.0.11	multiport dports 80	Ver
3449	0	0 B	DROP	tcp	--	*	*	78.105.10.0/24	157.92.0.11	multiport dports 80	Ver
3449	0	0 B	DROP	tcp	--	*	*	42.80.100.0/24	157.92.0.11	multiport dports 80	Ver

Nodo: Scrubbing1 (133.1.0.10)
Chain: FORWARD
Fecha: 2022-04-02 22:47:17

ASN	Packets	Bytes	Target	Protocol	Options	In	Out	Source	Destination	Extra	Comentarios
3449	12040	964.31 KiB	DROP	tcp	--	*	*	87.111.200.0/24	157.92.0.11	multiport dports 80	Ver
3449	11514	883.13 KiB	DROP	tcp	--	*	*	78.105.10.0/24	157.92.0.11	multiport dports 80	Ver
3449	12388	942.04 KiB	DROP	tcp	--	*	*	42.80.100.0/24	157.92.0.11	multiport dports 80	Ver

Por otro lado, a través de los paneles de visualización que el cliente podría enriquecer a partir de la captura de NetFlow en su organización, se observa una caída considerable del tráfico analizado que se corresponde al filtrado por los *scrubbing centers*, como se muestra en la figura 5.26.

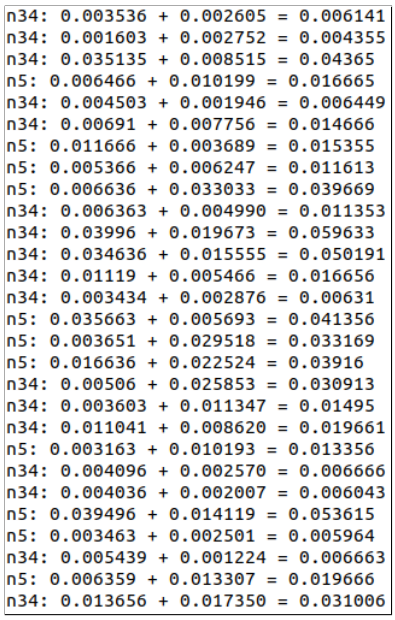
Figura 5.26: Tráfico entrante en la UBA, muestra la caída a partir del inicio del filtrado en la última parte



Para confirmar que el ScrubbingUNLP está funcionando como es esperado se puede ingresar al sitio como un cliente y observar el tiempo de respuesta, el cual, como se espera funciona perfectamente con

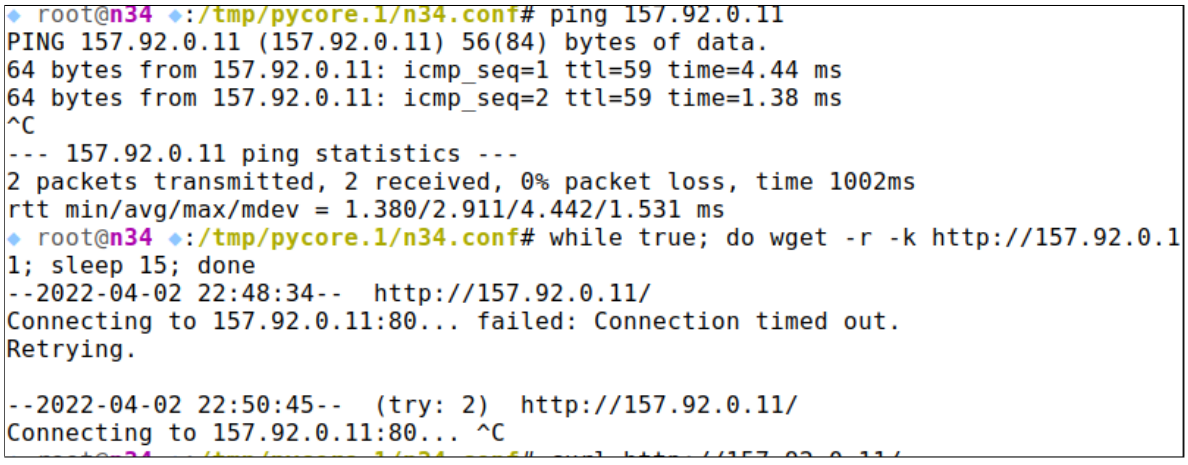
menos de 1 segundo de tiempo de respuesta, y esto se puede ver en la figura 5.27, la cual representa para los nodos cliente n34 (España) y n5 (Argentina) los tiempos de carga del sitio web.

Figura 5.27: Tiempo de respuesta en segundos para los nodos clientes n34 (España) y n5 (Argentina)



En la imagen 5.28 se observa que el cliente legítimo **n34** no puede realizar una petición HTTP al servidor web afectado. Se aprecia allí que recibe un *timeout* porque no puede llegar al servidor. Este cliente es legítimo pero está dentro de la red que contiene un atacante y, por ello, es una de las redes filtradas en el listado de FlowSpec. Este es uno de los problemas que se mencionó previamente sobre dejar a algunos clientes legítimos sin acceso al servicio, pero solo son aquellos que están en la misma red que los atacantes y, por ende, estos también podrían llegar a ser atacantes dada la cercanía entre ellos.

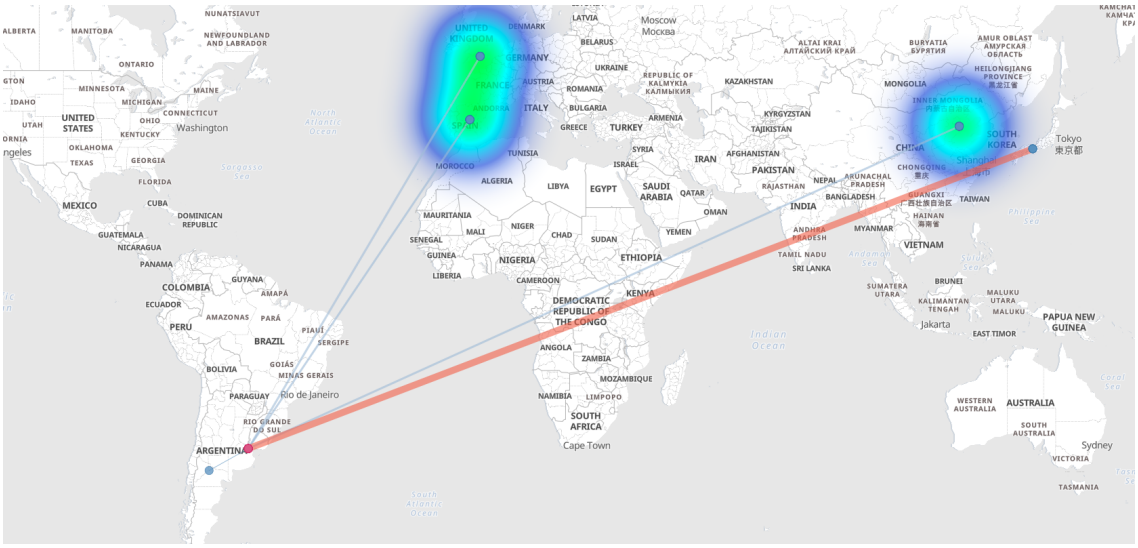
Figura 5.28: Cliente que no puede acceder al servicio web dado que su red es filtrada en FlowSpec debido a que de ella proviene uno de los ataques



Por otro lado, a través del sistema de monitoreo de flujos instalado en los *scrubbing centers* se podrá

observar que cambia el tamaño de los flujos respecto a su geolocalización, ya que, dependiendo de dónde se este monitoreando, el usuario puede que observe concentraciones de tráfico proviniendo desde los *scrubbing centers* en vez de los usuarios y atacantes previamente mostrados. La figura 5.29 muestra cómo se encuentran los flujos de tráfico durante el filtrado. Se nota que la línea que conecta la zona de Japón, lugar donde se encuentra instalado el *scrubbing center*, es mucho más gruesa que las tres que corresponden a los atacantes. Esto informa que la cantidad de tráfico que se está dando allí es mayor que las tres zonas previas.

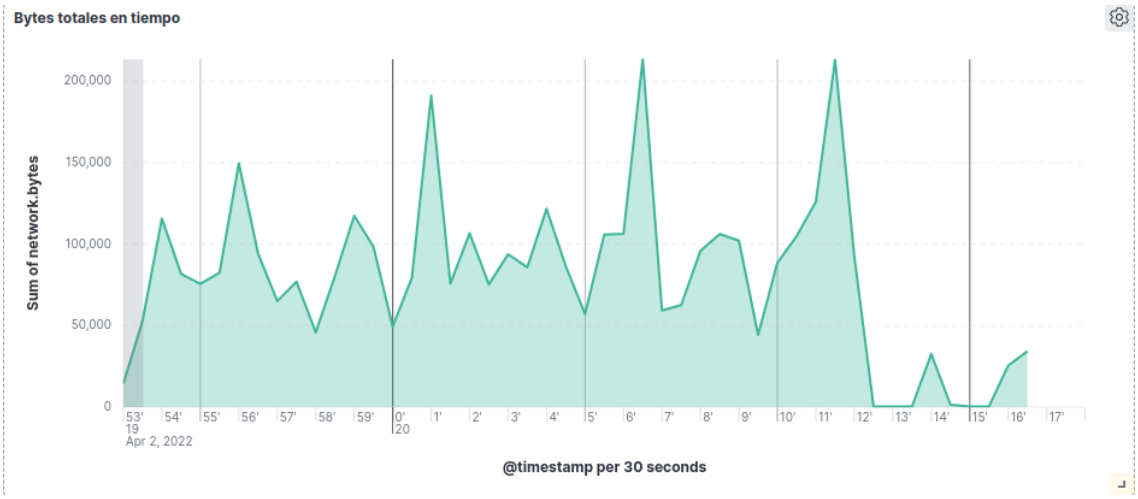
Figura 5.29: Los flujos de tráfico incrementan desde los *scrubbing centers* ya que retransmiten el tráfico legítimo de los clientes; en este caso, el *scrubbing center* ubicado en Japón es el que comienza a incrementar su flujo de tráfico



5.1.1.5 Fin del ataque

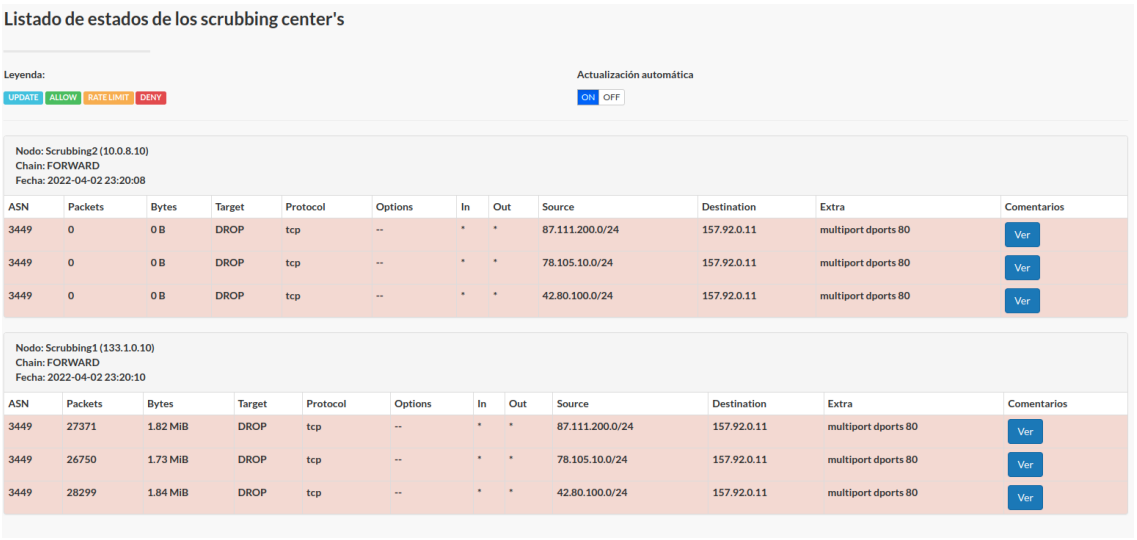
Con el o los ataques mitigados, los sistemas funcionarán como es debido y dependiendo de las características de los filtros puestos como se ha mencionado. Pero los mismos pueden costar recursos en Internet para los servicios contratados, por lo que eventualmente una vez que se detecta que el mismo cesa completamente se podrá detener el ScrubbingUNLP. Es posible observar el cese del ataque a través de los cambios del analizador del Kibana, que analiza los NetFlows procesados en los *scrubbing centers*. La figura 5.30 muestra la caída del volumen de datos que atraviesa un *scrubbing center*, con destino al AS víctima, debido a que deja de recibir los ataques. Se observa allí que los últimos picos son mucho más bajos que los que habían previamente.

Figura 5.30: Caída del volumen de datos, con destino al AS víctima, que recibe el *scrubbing center* visualizado a través del NetFlow instalado



Se observa cuando se finaliza el ataque que la cantidad de bytes filtrados (ver imagen 5.31) en la sección de **Listado de estados de *scrubbing centers*** se mantiene constante o varía apenas cada grandes intervalos. Cuando se detecte que no hay más tráfico filtrado, se puede deducir que ya no hay ataque ocurriendo y se puede dar por finalizado el mismo pasando al siguiente paso que implica detener finalmente el servicio de limpieza para este ataque.

Figura 5.31: Cuando el ataque DDoS finaliza, los bytes y paquetes observados en las tablas permanecen constante



5.1.1.6 Detención del sistema de limpieza

Cuando se considera que el ataque finalmente se ha acabado, se puede proceder al eliminado de las reglas de filtrado de FlowSpec si se lo desea ubicados en la sección de **Listado de anuncios**. Cuando el ataque de denegación de servicio ha finalizado, se procede a eliminar las reglas de filtrado de FlowSpec seguido de

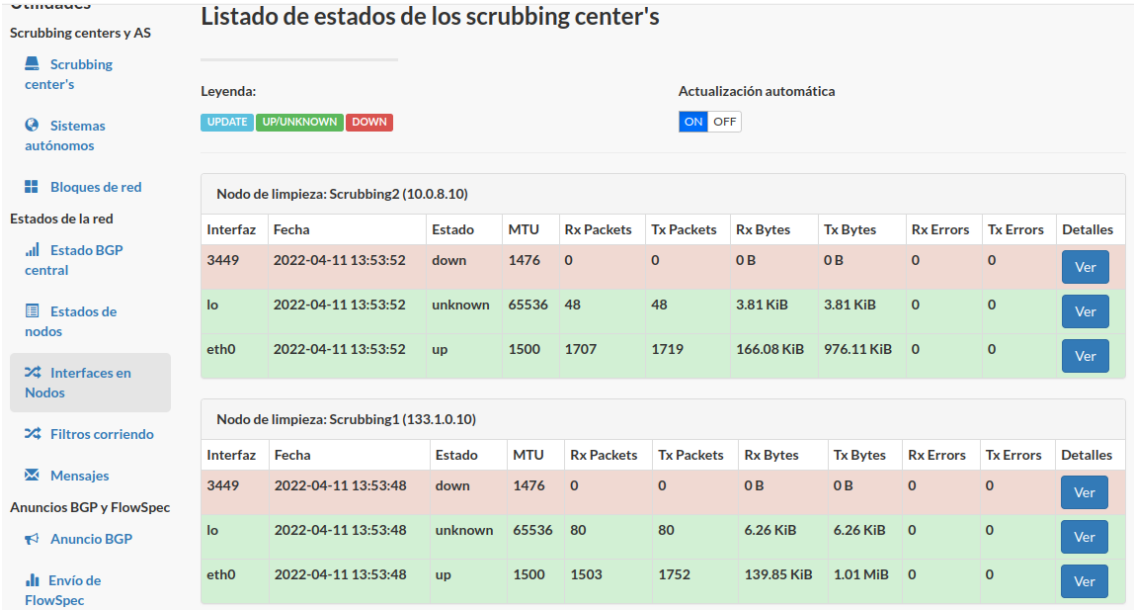
la publicación del bloque de red por BGP. De esta manera, se evita que el tráfico sea enviado al *scrubbing center* y este lo procese. También, si se elimina la publicación del bloque de red, se eliminarán también todas las reglas de filtrado de FlowSpec asociadas a la publicación.

Si esta operación fue ejecutada correctamente, debería observarse un mensaje y no debería verse listada la publicación en cuestión. En la vista de **Filtros Corriendo** se removerán automáticamente los filtros asociados a los FlowSpec que se eliminaron. En la figura 5.32 no hay ningún filtro porque no hay publicación de filtrado activa.

Figura 5.32: Cuando se eliminan los anuncios FlowSpec, los filtros se eliminan de los *scrubbing centers* automáticamente

Por otro lado, en la sección **Interfaces en Nodos** no se elimina la entrada asociada al túnel GRE del AS que había publicado un anuncio, esta permanece en las tablas de los *scrubbing centers* pero su estado pasa a estar *down* si no hay ningún anuncio asociado a esta interfaz activo (ver imagen 5.33).

Figura 5.33: Las entradas agregadas para el AS que realizó las publicaciones no se eliminan de las tablas, se cambia su estado a *down* señalado en color rojo



Esto provocará que el bloque antes afectado ya no sea enviado por BGP al resto de los *routers* de Internet para que sea enrutado a través de los *scrubbing centers* cercanos a los clientes y, por ende, el tráfico hacia la organización vuelva a continuar con su enrutamiento de red normal previo a los ataques.

CAPÍTULO 6

CONCLUSIONES Y TRABAJO FUTURO

6.1 Conclusiones generales

En relación al funcionamiento esperado, se puede decir que cumple las expectativas ya que la plataforma desarrollada permite filtrar el tráfico no deseado hasta el punto de mitigar el ataque como se debe, únicamente dependiendo de la correcta definición manual de reglas por parte de los usuarios.

Dada la ejecución del sistema en un entorno de simulación y en el escenario hipotético planteado, se observa que los tiempos de ejecución de los protocolos de enrutamiento son óptimos, así como también que los tiempos de propagación de reglas y tiempos de respuesta del sistema ante un ataque de DDoS son adecuados.

A partir del análisis de las soluciones existentes y el trabajo realizado durante esta tesina, se concluye que es necesario que los nodos de limpieza estén ubicados en lugares estratégicos. Sería interesante a partir de la publicación de esta tesina compartirla con otras universidades para poder promover el proyecto y así, entre todas las participantes, levantar el sistema de limpieza como un mecanismo de pruebas y opcional sistema de limpieza disponible para las mismas.

Por otro lado, se logró la integración de diversas herramientas que permiten resolver un problema en particular, en este caso un ataque de DDoS, evitando interacciones con ISPs y terceros. En particular, el módulo WebScrub simplifica las acciones del usuario evitándole los inconvenientes que conlleva ejecutar instrucciones desde la CLI (Interfaz de Línea de Comandos) y le presenta al usuario una interfaz gráfica con la información necesaria para que este pueda actuar en consecuencia cuando lo considere oportuno.

Para finalizar, se puede concluir que los objetivos iniciales de la tesina han sido alcanzados. Sin embargo, resta mucho trabajo por delante, tanto en aspectos relacionados a los ataques de DDoS como en añadir funcionalidad en la solución propuesta en el marco de esta tesina.

6.2 Trabajo futuro

Uno de los principales trabajos a realizar, y a su vez más interesantes, es la puesta en producción de la plataforma propuesta en una red pública como Internet. Para llevar a cabo esta tarea, será necesario contar con recursos suficientes para poder desplegar los componentes necesarios, aportes que podrían ser realizados por organizaciones que quieran colaborar en el proyecto.

Sin embargo, en una primera etapa sería deseable implementar el sistema en un entorno real con el objetivo de poder realizar pruebas y sacar conclusiones de las métricas obtenidas a partir del monitoreo global de los componentes de la solución.

Dentro de estas pruebas será necesario evaluar las capacidades de filtrado de diferentes herramientas, incluyendo el análisis con *firewalls* implementados en una máquina virtual, *firewalls* implementados en hardware dedicado y, por último, *firewalls* comerciales.

Sobre estos diferentes tipos de *firewalls*, se deberá ampliar la funcionalidad que brinda ScrubbingUNLP, implementando nuevos tipos de reglas de filtrado como filtros por protocolos de capa de aplicación. Además, en relación a esto, será necesario realizar modificaciones en algunos componentes de la solución con el fin de poder interactuar con estos tipos de *firewalls*.

Con respecto al componente web de la solución, se propone la internacionalización de la misma con el objetivo de poder ser utilizada por una comunidad más amplia. También se propone agregar más funcionalidades a la herramienta como validación de los recursos de red del usuario (ASN y bloques de direcciones IP) utilizando mecanismos automáticos de consultas a servicios como Peeringdb [63].

En relación al costo de la herramienta, al comienzo del trabajo se planteó que la plataforma de mitigación de ataques de DDoS desarrollada surge como alternativa a soluciones comerciales inalcanzables para ciertas organizaciones. A partir del *accounting*, la herramienta deberá poder determinar el uso del servicio de cada uno de los usuarios de la plataforma y definir el costo de utilización en base a estas métricas, más allá de los costos fijos existentes.

También desplegar la plataforma en un entorno colaborativo entre miembros de redes de investigación de diferentes países para que cada miembro pueda beneficiarse tanto del filtrado de ataques de DDoS así como de ayudar a mitigar los ataques sufridos por otros.

Para finalizar, a partir del análisis del estado actual y las tendencias relacionadas a los ataques de DDoS, determinar posibles ubicaciones de los *scrubbing centers* para una efectiva mitigación de ataques a nivel local y en diferentes zonas a nivel global.

Lista de Figuras

Lista de Tablas

Tabla **Página**

[1] Free Software Foundation, "¿Qué es el Software Libre?." https://www.gnu.org/philosophy/free-sw.es.html.

[2] Hawkinson, J. and Bates, T., "IGuidelines for creation, selection, and registration of an Autonomous System (AS)." https://datatracker.ietf.org/doc/html/rfc1930, marzo 1996.

[3] Mike Jensen, "Promoting the Use of Internet Exchange Points: A Guide to Policy, Management, and Technical Issues." https://www.isoc.org/wp-content/uploads/2012/12/promote-ixp-guide.pdf.

[4] Cámara Argentina de Internet, "Puntos de Intercambio de Tráfico – IXPs." https://www.cabase.org.ar/que-es-un-nap-3/.

[5] Le Faucheur, F. and Bitar, N., "Content Distribution Network Interconnection (CDNI) Problem Statement." https://datatracker.ietf.org/doc/html/rfc6707, septiembre 2012.

[6] Red Hat Enterprise Linux, "Protocolo SSH." https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/ch-ssh.html.

[7] Gopal Dommety, "Key and Sequence Number Extensions to GRE." https://datatracker.ietf.org/doc/html/rfc2890, 2000.

[8] University of Southern California, "TRANSMISSION CONTROL PROTOCOL." https://datatracker.ietf.org/doc/html/rfc793, 1981.

[9] D.L. Mills, "Exterior Gateway Protocol Formal Specification." https://datatracker.ietf.org/doc/html/rfc904, 1984.

[10] Kirk Lougheed, Jacob Rekhter, "A Border Gateway Protocol (BGP)." https://datatracker.ietf.org/doc/html/rfc1105, 1989.

[11] Y. Rekhter, T. Li, S. Hares, "A Border Gateway Protocol 4 (BGP-4)." https://datatracker.ietf.org/doc/html/rfc4271, 2006.

[12] C. Loibl, S. Hares, R. Raszuk, D. McPherson, M. Bacher, "Dissemination of Flow Specification Rules." https://datatracker.ietf.org/doc/html/rfc8955, 2020.

[13] J. Kurose and K. Ross, *Redes de computadoras: Un enfoque descendente. 7ma Edición.*, p. 47. PEARSON, S.A., 2017.

[14] J. Kurose and K. Ross, *Redes de computadoras: Un enfoque descendente. 7ma. Edición.*, p. 48. PEARSON, S.A., 2017.

[15] Cybersecurity & Infraestructure Security Agency , "UDP-Based Amplification Attacks." https://www.cisa.gov/uscert/ncas/alerts/TA14-017A, 2019.

[16] Ferguson, P. and Senie, D., "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing." https://www.rfc-editor.org/info/bcp38, mayo 2000.

[17] Baker, F. and Savola, P., "Ingress Filtering for Multihomed Networks." https://www.rfc-editor.org/rfc/rfc3704, marzo 2004.

[18] Oficina de Seguridad del Internauta, INCIBE, "Botnet: Mirai." https://www.osi.es/es/servicio-antibotnet/info/mirai, 2004.

[19] MANRS, "MANRS About." https://www.manrs.org/about/, 2022.

[20] INCIBE, "Medidas de prevención contra ataques de denegación de servicio." https://www.incibe.es/protege-tu-empresa/blog/medidas-prevencion-ataques-denegacion-servicio, 2019.

[21] Warren Kumari, Danny McPherson, "Remote Triggered Black Hole Filtering with Unicast Reverse Path Forwarding (uRPF)." https://datatracker.ietf.org/doc/html/rfc5635, 2009.

[22] CloudFlare, "What is blackhole routing?." https://www.cloudflare.com/learning/ddos/glossary/ddos-blackhole-routing/, 2022.

[23] Team Cymru, "Who We Are." https://team-cymru.com/company/, 2022.

[24] Team Cymru, "Unwanted Traffic Removal Service 2.0." https://team-cymru.com/community-services/utrs/, 2022.

[25] j-shank, "Unwanted Traffic Removal Service (UTRS)." https://github.com/team-cymru/network-security-templates/blob/master/UTRS-Peering-Guide/README.md, 2021.

[26] James Shank, "UTRS Roadmap." https://www.sanog.org/resources/sanog37/SANOG37_Conference-Unwanted_Traffic_Removal_Service-James_Shank-Team_Cymru.pdf, 2021.

[27] MadBoxPC, "Level 3 inaugura el Primer Scrubbing Center de DDoS de Latinoamérica en São Paulo." https://www.madboxpc.com/level-3-inaugura-el-primer-scrubbing-center-de-ddos-de-latinoamerica-en-sao-paulo/, 2016.

[28] Cisco, "Cisco Secure DDoS Protection: Global Scrubbing Centers Data Sheet." https://www.cisco.com/c/en/us/products/collateral/security/secure-ddos-protect-scrubbing-center-ds.html, 2021.

[29] Lumen, "DDoS Hyper." https://www.lumen.com/en-us/security/ddos-hyper.html, 2022.

[30] nScrub, "nScrub documentation." https://www.ntop.org/guides/nscrub/index.html, 2018.

[31] nScrub, "nScrub - DDoS Mitigation System." https://www.ntop.org/products/ddos-mitigation/nscrub/, 2018.

[32] ntop, "Multi-10 Gbit RX/TX Packet Processing from Hosts and Virtual Machines." https://www.ntop.org/products/packet-capture/pf_ring/pf_ring-zc-zero-copy/, 2022.

[33] Info-Stor, "ntop nScrub DDoS Mitigation Hardware Appliance." https://www.info-stor.co.uk/product/ntop-nscrub-ddos-mitigation-hardware-appliance/.

[34] Telecom Argentina, "Mitigación Ataques DDoS para Empresas." https://www.telecom.com.ar/grandes-empresas/productos/soluciones-digitales/mitigacion-ataques-ddos, 2022.

[35] CloudFlare, "Protección contra DDoS para empresas." https://www.cloudflare.com/es-es/lp/ppc/ddos-x/?gclsrc=aw.ds, 2022.

[36] Imperva, "BGP-based DDoS mitigation." https://www.imperva.com/products/infrastructure-ddos-protection-services/, 2022.

[37] Free Software Foundation, "Licenses." https://www.gnu.org/licenses/licenses.html, 2022.

[38] Debian Project, "Reasons to use Debian." https://www.debian.org/intro/why_debian.

[39] Free Software Foundation, "GNU Bash." https://www.gnu.org/software/bash/.

[40] Python Software Foundation, "General Python FAQ." https://docs.python.org/3/faq/general.html, 2021.

[41] Django Software Foundation, "Django makes it easier to build better web apps more quickly and with less code.." https://www.djangoproject.com/, 2022.

[42] Exa-Networks, "ExaBGP repository." https://github.com/Exa-Networks/exabgp#introduction, 2022.

[43] ArchLinux, "iptables (Español)." https://wiki.archlinux.org/title/Iptables_(Espa%C3%B1ol).

[44] sla, "fprobe." https://sourceforge.net/projects/fprobe/, 2002.

[45] Free Software Foundation, "Libcap - Free Software Foundation." https://directory.fsf.org/wiki/Libcap#tab=Overview.

[46] Ubuntu Manpage Repository, "fprobe - a NetFlow probe." https://manpages.ubuntu.com/manpages/bionic/man8/fprobe.8.html.

[47] Elasticsearch B.V., "¿Qué es Elasticsearch?." https://www.elastic.co/es/what-is/elasticsearch.

[48] Elasticsearch B.V., "¿Qué es Kibana?." https://www.elastic.co/es/what-is/kibana.

[49] Elasticsearch B.V., "Filebeat overview." https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-overview.html#filebeat-overview.

[50] MDN contributors, "MVC." https://developer.mozilla.org/es/docs/Glossary/MVC, 2020.

[51] A. Hunt and D. Thomas, *The Pragmatic Programmer : From Journeyman to Master*, p. 320. Addison-Wesley Professional, 1999.

[52] SQLite, "What Is SQLite?." https://www.sqlite.org/index.html.

[53] Oracle Corporation, "Why MySQL?." https://www.mysql.com/why-mysql/.

[54] The PostgreSQL Global Development Group, "What is PostgreSQL?." https://www.postgresql.org/about/.

[55] dormando, "Memcached protocol." https://github.com/memcached/memcached/blob/master/doc/protocol.txt, 2022.

[56] J. Mitchell, "Autonomous System (AS) Reservation for Private Use." https://datatracker.ietf.org/doc/html/rfc6996, 2013.

[57] U.S. Naval Research Laboratory, "Common Open Research Emulator (CORE)." https://www.nrl.navy.mil/Our-Work/Areas-of-Research/Information-Technology/NCS/CORE/.

[58] OffSec Services Limited, "hping3 Usage Example." https://www.kali.org/tools/hping3/.

[59] FRRouting Project, "FRRouting Project." https://frrouting.org/.

[60] Docker Inc., "Docker overview." https://docs.docker.com/get-started/overview/.

[61] HashiCorp, "Introduction to Vagrant." https://www.vagrantup.com/intro.

[62] Asociación Redes de Interconexión Universitaria, "Red de Interconexión Universitaria." https://riu.edu.ar/institucional/la-red/#page-content, 2022.

[63] PeeringDB, "The Interconnection Database." https://www.peeringdb.com/.