ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

ANÁLISIS Y DETECCIÓN DE VULNERABILIDADES TIPO BOTNETS, EN LA ESCUELA POLITÉCNICA NACIONAL, UTILIZANDO HERRAMIENTAS DE SOFTWARE LIBRE

PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN

JOSÉ ANDRÉS TINAJERO GUAJÁN
jose.tinajero@epn.edu.ec

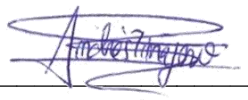
Director: PhD. JENNY GABRIELA TORRES OLMEDO
jenny.torres@epn.edu.ec

Quito, julio 2024

DECLARACIÓN

Yo, José Andrés Tinajero Guaján, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

La Escuela Politécnica Nacional puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.



José Andrés Tinajero Guaján

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por José Andrés Tinajero Guaján, bajo mi supervisión.

Firmado electrónicamente por:
JENNY GABRIELA
TORRES OLMEDO

_____

PhD. Jenny Gabriela Torres Olmedo

DIRECTOR

AGRADECIMIENTOS

Agradezco en primer lugar a Dios, por darme la capacidad, perseverancia, y fuerza para llegar hasta este punto de mi vida.

A mis padres y mis hermanos, porque gracias a ellos soy he podido seguir adelante.

A mi esposa Wendy por su amor y apoyo incondicional en cada momento importante de nuestras vidas.

A mis amigos, especialmente aquellos que siempre estuvieron ahí con sus palabras de aliento y cariño para que siga adelante, son muy importantes para mí. Y muchos de ellos los considero como hermanos.

A mi tutora, por su paciencia, comprensión y apoyo en los últimos semestres de la carrera y sobre todo durante el desarrollo de todo este trabajo.

Al personal del CSIRT-EPN y la DGIP por brindarme toda la información y el apoyo para llevar a cabo este proyecto.

A todas y cada una de las personas que me apoyaron de una u otra forma durante todo este periodo, de verdad se los agradezco y que Dios los bendiga considerablemente.

DEDICATORIA

Para mis padres que me han apoyado incondicionalmente, mi hermana que con su ejemplo ha demostrado que es posible cumplir lo que nos propongamos en la vida, mi hermano que siempre ha estado conmigo, mi esposa que me ha amado y motivado a seguir adelante, mi familia, y todos mis amigos que han orado por mí para que hoy pueda llegar a este punto, los quiero mucho. Finalmente, a Dios que me ha permitido prepararme día a día para poder ser un instrumento útil para la vida de los demás.

ÍNDICE DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

Las botnets representan una de las principales amenazas que afectan a las organizaciones, empresas, instituciones educativas y a los usuarios. A través de ellas los cibercriminales consiguen comprometer dispositivos sin el consentimiento de sus dueños, con el fin de usarlos para el lanzamiento de distintos tipos de ataques, como denegación de servicio, envió de spam, robo de identidad, entre otros.

El presente proyecto de titulación tiene como propósito realizar el análisis y detección de vulnerabilidades tipo botnet. Así como el estudio de las botnets y sus características, las técnicas de detección existentes y la búsqueda de herramientas de software libre para la detección de este tipo de vulnerabilidad.

Finalmente se procede a la creación de un ambiente de pruebas aplicado a un caso de estudio en una institución educativa del Ecuador donde se analizan los eventos ocurridos y la determinación de la vulnerabilidad.

Palabras clave: Bot, Detección de botnets, Incidente de seguridad, NIDS, software libre, Snort-IDS, Stack ELK

ABSTRACT

Botnets represent one of the main threats affecting organizations, businesses, educational institutions, and users. Through them, cybercriminals manage to compromise devices without their owners' consent, in order to use them for launching various types of attacks such as denial of service, spamming, identity theft, among others.

The purpose of this graduation project is to conduct the analysis and detection of botnet-type vulnerabilities. This includes studying botnets and their characteristics, existing detection techniques, and the search for open-source tools for detecting this type of vulnerability.

Finally, the creation of a testing environment is carried out, applied to a case study in an educational institution in Ecuador where the events are analyzed, and vulnerability assessment is conducted.

Keywords: Bot, Botnets Detection, Security Incident, NIDS, Free Software, Snort-IDS, Stack ELK

1. INTRODUCCIÓN

La presente propuesta de proyecto ha sido elaborada debido a la existencia de una colección o grupo de computadores conectados a Internet que permiten a un operador controlar remotamente ejecutando tareas automatizadas denominadas botnets. Una botnet realiza tareas ilícitas sin el conocimiento de los propietarios y son una amenaza constante para empresas, instituciones y usuarios finales. Los computadores que son parte de las botnets se los denomina “zombies o “drones”. La mayoría de ellos interactúan para realizar escaneo de direcciones IP en busca de vulnerabilidades [1] [2].

Las botnets han sido responsables de la propagación de correo no deseado(spam), instalación de malware, robo de datos personales, ataques de denegación de servicio (DoS), captura de actividad por teclado, fraude, entre otros [3] [4]. Las botnets son controladas remotamente por un botmaster. Un bot es un pequeño tipo de virus informático, los usuarios no pueden notar que la existencia de un bot en el equipo.

Los dispositivos que son infectados suelen pueden llegar a ser parte de una botnet al ser atacados mediante ingeniería social y archivos de dudosa procedencia, o mediante ataques directos en redes locales. Además de los sistemas que la misma botnet ataca para obtener algún beneficio, pero no para hacerlos parte de la botnet.

Las botnet son el problema de seguridad más importante y extendida en Internet, siendo responsables de casi el 90% del SPAM mundial [5]. Si bien múltiples esfuerzos se han hecho para detectarlas y detenerlas, todavía no hay métodos eficaces que puedan garantizar un alto grado de detección. Este hecho es lo que motiva al estudio en el área y a profundizar en la temática. Las técnicas para detectar botnets han sido variadas. Se han propuesto técnicas basadas en detección de anomalías mediante el estudio del tráfico DNS (Domain Name Service), detección basada en firmas, detección basada en algoritmos genéticos, detección basada en análisis de datos, entre otras. Las mismas han conseguido resultados variables y algunas de ellos han sido implementadas en producción. Sin embargo, todavía no son llegan a ser 100% eficientes y se necesitan otras técnicas de detección.

Actualmente diferentes instituciones educativas se ven afectadas como lo muestra el informe de Kaspersky Lab del segundo trimestre del 2018. En este informe se presenta un crecimiento del 94.47% de ataques a servidores Linux en comparación con el informe del primer trimestre del 2018 [6]. Así como varias instituciones educativas de Asia que fueron víctimas de la botnet XOR-DDoS, utilizando ataques DDoS, DNS flood y de fuerza bruta

contra la cuenta root de sistemas Linux, infectando varios servidores [7]. La preferencia de los atacantes a este tipo de instituciones se debe a la rapidez del canal de internet en comparación con los dispositivos domésticos y al crecimiento que aporta infectar servidores adicionales.

Es por esto que se plantea realizar un análisis de las zonas institucionales en donde existe la sospecha de la existencia de la vulnerabilidad tipo botnet e identificar los equipos afectados, describir y clasificar el tráfico malicioso, para finalmente, emitir recomendaciones para remediar los equipos infectados. Se utilizará la técnica de análisis de datos y un sensor con el IDS Snort integrado al administrador de eventos Stack ELK.

El capítulo 2 presenta el marco teórico específico. El capítulo 3 presenta un estudio comparativo y la selección de la técnica de detección de botnets apropiada para nuestro caso de estudio. El capítulo 4 presenta la elección de las herramientas de software libre. El capítulo 5 presenta el caso de estudio, la aplicación de la técnica y el software a utilizar; también la instalación correcta de las herramientas de software libre (IDS Snort con Stack ELK). El capítulo 6 presenta las estrategias de contingencia aplicadas a nuestro caso de estudio.

2. MARCO TEÓRICO

2.1. Aspectos Generales de la Seguridad de la Información

El objetivo de la seguridad de la información según la norma ISO 27001 es “preservación de la confidencialidad, la integridad y la disponibilidad de la información, además también pueden estar implicados otras propiedades como la autenticidad, la responsabilidad, el no repudio, y la fiabilidad” [8]. Las definiciones de las características de la seguridad de la información son [9] [10] [11]:

- Disponibilidad: acceso a la información cuando se requiere, teniendo en cuenta la privacidad. Es decir, evitar “caídas” del sistema que permitan accesos ilegítimos, e impidan el acceso a los diferentes servicios de la organización.
- Confidencialidad: información accesible solo para personal autorizado. La información no debe llegar a personas o entidades que no estén autorizados.
- Integridad: información correcta sin modificaciones no autorizadas ni errores. Se protege frente a vulnerabilidades externas o posibles errores humanos.
- Autenticación: información procedente de un usuario que es quien dice ser. Se verifica y se debe garantizar que el origen de los datos es correcto.

Conocer y aplicar los cuatro pilares de la seguridad de la información es la base para un correcto análisis, administración de vulnerabilidades y riesgos; de esta manera, superar los impactos que pueden ser ocasionados a la organización. A continuación, se va a definir el sustento teórico relacionado con la seguridad de la información.

2.1.1. Activo

Son los recursos de los sistemas de información, son necesarios para que una institución funcione adecuadamente y que aquellos objetivos de la organización sean alcanzados [12].

2.1.2. Activo de red

Hace referencia a dispositivos físicos que permiten conexión entre una o varias redes que pueden ser atacados intencionalmente o de forma accidental, generando problemas para el funcionamiento de la red haciendo que esta no trabaje de manera óptima si estos activos son afectados [13].

2.1.3. Amenaza

Es una acción o evento que puede violar la seguridad de un entorno de un sistema de Información el cual tiene tres componentes [14]:

- Objetivos: El aspecto de la seguridad que puede ser atacado;
- Agentes: Las personas u organizaciones que originan la amenaza;
- Eventos: El tipo de acción que origina la amenaza.

2.1.4. Vulnerabilidad

Se define como la susceptibilidad de algo para absorber negativamente incidencias externas; es una vía de ataque potencial.

2.1.5. Riesgo

Probabilidad de que la amenaza actúe sobre el activo. Se utiliza para cuantificar el daño (probable) que puede causar la amenaza [14].

2.1.6. Impacto

El impacto hace referencia a la magnitud del daño que puede ser causado por una amenaza que explota una potencial vulnerabilidad. El nivel de impacto se rige por las amenazas potenciales, y a su vez produce un valor para los activos de Tecnología de la información (TI) y para los recursos afectados [15].

2.1.7. Incidente de Seguridad

Puede ser un único evento o una serie de eventos de seguridad de la información no deseados o inesperados que tienen una probabilidad significativa de comprometer las operaciones comerciales y amenazar la seguridad de la información [16].

2.2. ¿Qué es una Botnet?

Una botnet es una red virtual de ordenadores zombies. Las botnets pueden aprovecharse para organizar ataques concertados contra otros recursos informáticos, por ejemplo, ataques distribuidos de denegación de servicio (DDoS) contra redes específicas [17]. Los principales objetivos de una botnet, recaen en tres categorías:

- Dispersión de la información: consiste en envió de spam, creando ataques de denegación de servicio DoS.
- Recolección de la información: involucra el almacenamiento de información de identidad, financiera, contraseñas, social como direcciones de correo o cualquier tipo de datos localizados en el host.
- Procesamiento de información: es el beneficio económico; los datos recolectados pueden ser vendidos.

Un host está comprometido cuando un software automatizado es instalado en él sin la autorización del administrador. El operador de la botnet pasa a tomar control total del equipo, e incorpora esta estructura. Una vez creado un nuevo nodo, puertas traseras (puertos de Internet abiertos) son abiertas para recibir instrucciones; worms (gusanos) o virus troyanos.

2.2.1. Bot

Los programas Bot son códigos que operan automáticamente como agentes para un usuario u otro programa [18]. El primer bot fue probablemente Eggdrop, creado por Jeff Fisher, que se originó como una característica útil del chat de retransmisión de internet (IRC) a principios de los 90. Los primeros bot se diseñaron para permitir a los operadores de IRC crear respuestas automatizadas a las actividades de IRC. Las instrucciones o comandos de ataque van desde iniciar un gusano, envió de spam a través de Internet hasta interrumpir una solicitud legítima del usuario. Los bots son computadoras con software malicioso instalado en ellos, e interactúan con la máquina de un individuo sin que el usuario lo note o incluso sin que intervenga.

2.2.2. BotMaster

Término utilizado para describir un individuo o sistema responsable del mantenimiento de bots. A menudo los bots responden quiénes son sus botmasters cuando se les pregunta y ejecutan las actividades para las cuales fueron diseñados.

2.3. Tipos de ataques realizados por botnets

Existen diferentes tipos de ataques realizado por botnets de acuerdo a la información o beneficio que desea obtener. A continuación, se definen los tipos de ataques más frecuentes:

2.3.1. Malware

El malware (abreviatura de software malicioso), generalmente se considera como un software que tiene como objetivo interrumpir las operaciones ordinarias de un sistema computarizado mediante la recopilación de información confidencial o el acceso no autorizado a los sistemas informáticos y principalmente acosar a los usuarios [19]. La identificación de malware es crítica, debido al hecho de que hoy en día, están creciendo cada vez más, causando mucho daño. El malware se puede dividir en varias categorías, como virus, gusanos, troyanos, spywares y adware, Rootkits, etc.

2.3.2. Denegación de Servicio Distribuida (DDoS)

Consiste en direccionar una gran cantidad de tráfico a un servidor u ordenador desde muchos equipos a la vez, de tal manera que los recursos del servidor acaben no siendo suficientes y no pueda responder peticiones legítimas. Una de las formas más comunes, es lanzar un ataque breve para extorsionar protección a cambio de no lanzar otro a un nivel mayor [20].

2.3.3. Spamming

Es una de las actividades presentes desde el comienzo de las bots. Las máquinas infectadas actúan como relevadores de correos electrónicos para el botmaster, y pueda enviar exorbitantes cantidades de correos electrónicos no deseados [21]. Muchas organizaciones tienen políticas anti-spam.

2.3.4. Fraudes financieros

Además de poder expandir el alcance de sus operaciones a fraudes bancarios, las bots, mantienen la habilidad de añadir malware adicional en un equipo comprometido, que ayuda a la recolección de información valiosa; seguridad social, números de tarjetas de crédito [22].

2.3.5. Fraude de clics

Los programas de bot y botnet de pay-per-click (pago por clic) publicitarios brindan oportunidades a los criminales para el lavado de dinero por medio de este tipo de ataque [23].

2.3.6. Search Engine Optimization (SEO)

Los propietarios del bot aumentan posiciones en motores de búsquedas, de manera artificial para llevar a los buscadores de sitios web e inyectar malware en un equipo [24].

2.3.7. Espionaje Industrial y Corporativo

Existen bots usados en combinación con emails específicos contra corporaciones y gobiernos en el intento de conseguir propiedad intelectual y secretos de estados, entre otros [25].

2.3.8. Bitcoin Mining

Bitcoin es una moneda virtual que puede ser intercambiada anónimamente en línea por productos y servicios. Funciona instalando un programa en la computadora del usuario, y realiza complejos cálculos. Al instalar bitcoin en un equipo, el atacante puede aprovechar el poder de procesamiento para minar algunas monedas y venderlas en sitios que operan por debajo de la ley [26].

2.4. Arquitecturas de Botnets

La fortaleza de las botnets radica en la flexibilidad de equipos que están conectados y controlados de forma remota. Por lo tanto, se utilizan diferentes enfoques para hacer frente a los problemas de comunicación entre las entidades de la botnet [27]. Existen dos tipos de arquitecturas principales: centralizadas e hibridas. En la arquitectura centralizada se detallan botnets IRC y HTTP; mientras que, en la arquitectura descentralizada, se describen las botnet P2P.

2.4.1. Botnets IRC

Esta arquitectura se asemeja al modelo cliente-servidor. Utiliza un servidor central de gran ancho de banda llamado servidor C&C (Control and Command) para reenviar mensajes entre varios bots. El servidor de C&C en una botnet es una computadora comprometida que ejecuta ciertos servicios de red como IRC, HTTP, etc. y que reúne los comandos emitidos por el botmaster a cada host en la botnet que se une al canal del servidor de C&C, tal como se muestra en la figura 2.1. Las botnets usan varios mecanismos para proteger sus comunicaciones, que incluyen el uso de contraseñas establecidas por los botmasters Las botnets basadas en IRC tienen una persistencia en la actualidad debido a la simplicidad y flexibilidad por parte del protocolo en texto basado en IRC. Las botnets contemporáneas utilizan mensajes ofuscados para evadir detección basada en firmas.

Las comunicaciones C&C son flujos dúplex bien estructuradas, similar a un protocolo de comando y respuesta. El bot debe responder cuando recibe un comando predefinido en un tiempo razonable. El nivel de red de respuesta de un bot ante un comando ofuscado, puede ser entre el mensaje de respuesta o la actividad de respuesta o ambos. Las ventajas de esta arquitectura son:

- Fácil implementación, no requiere ningún hardware especializado.
- Comunicación directa del botmaster con los bots.
- Oportunidad de enviar actualizaciones desde el botmaster.
- Escalabilidad.
- La principal desventaja de esta arquitectura tiene que ver con la facilidad de desactivar los bots cuando el servidor es detectado.

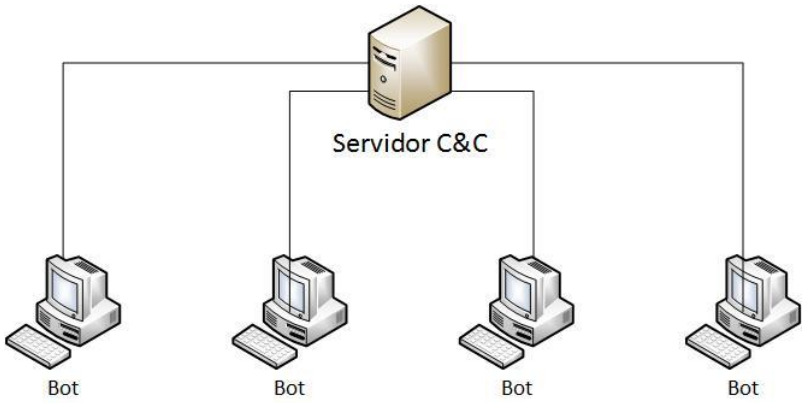


Figura 2.1 Botnet C&C

2.4.2. Botnets HTTP

Se utiliza en una arquitectura centralizada donde el botmaster publica las órdenes en un servidor web y los clientes, regularmente, se conectan al servidor y comprueban que tarea tienen que llevar a cabo. La comunicación se suele llevar a cabo encriptada. El protocolo HTTP permite la navegación web, por lo que los routers lo suelen permitir. En este caso, el servidor web se convierte en un único punto de fallo [28]. Bloqueando su acceso, se impediría el funcionamiento de la red y el uso de varios servidores permite eliminar el servidor C&C como un único punto de fallo

Este tipo de arquitectura comenzó con los avances de exploits (programa o técnica que aprovecha una vulnerabilidad) desarrollados por cibercriminales rusos. Estos pueden instalar software en máquinas remotas y controlarlos mediante un sitio web. Permite que sus nodos infectados utilicen una URL específica o dirección IP definida por su operador, de esta forma, establecen una conexión a un servidor web específico que juega el papel del C&C.

La diferencia que existe con la arquitectura IRC, son los elementos que conforman a la red de cómputo maliciosa, no permanecen conectados, luego de haber manifestado una comunicación con el servidor C&C por primera vez. De esta manera, cuando la comunicación se establezca, el bot revisa de manera periódica los comandos publicados en el servidor [29]. En la figura 2.2 se muestra el funcionamiento de una botnet HTTP.

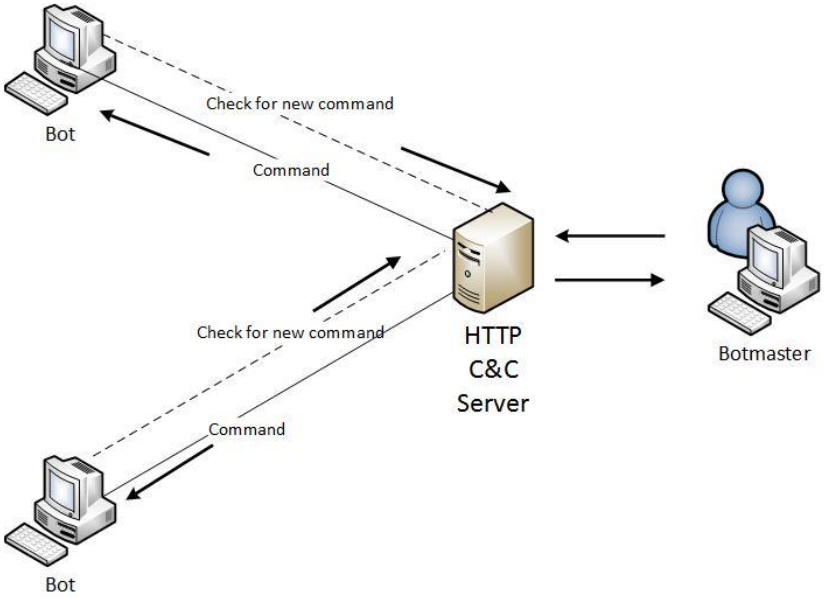


Figura 2.2 Botnet HTTP

2.4.3. Botnets P2P

En la arquitectura descentralizada se tienen a las botnets P2P. Botnets C&C de igual a igual utiliza la comunicación P2P sin un servidor central real para reenviar mensajes entre botnets, lo que lo hace más resistente a fallas en la red. A diferencia de la técnica centralizada de C&C, la técnica P2P C&C es mucho más difícil de descubrir y destruir; incluso si uno o más bots son neutralizados, el botnet aún continúa operando. Además, se puede usar una técnica P2P anónima para que sea aún más difícil de detectar. Sin embargo, el tamaño de botnet soportado por los sistemas P2P es generalmente muy bajo en comparación con los sistemas centralizados, lo que hace que los botmasters orientados a los beneficios eviten usar la técnica P2P. Además, la falta de propagación y la entrega de mensajes garantizados no existen en los sistemas P2P.

Un peer juega el papel de cliente y servidor al mismo tiempo; hacer peticiones a otros peers en la red y al mismo tiempo, poder responder peticiones entrantes. En una red P2P, los peers pueden organizarse en grupos mientras se comunican, colaboran, y comparten ancho de banda para completar tareas. Cada uno puede subir y descargar datos al mismo tiempo y en el proceso, nuevos peers pueden unirse mientras que otros pueden irse en cualquier momento. Cabe destacar que este proceso es transparente para los usuarios finales.

Dentro de las características de esta red se encuentra la capacidad de tolerancia a fallas; cuando un equipo se desconecta de la red, la aplicación P2P continuará con los clientes restantes. Si un peer descubre que no hay respuesta de otro nodo, este busca por otro peer que brinde el archivo a partir de la interrupción de la descarga.

Cada participante en una red P2P debe ser capaz de realizar ciertas operaciones para saber cómo detectar la existencia de otro cliente y las diversas partes de un archivo que otros peers puedan poseer [30]. Para superar estos inconvenientes es necesario poder descubrir otros clientes, poder conectarse con otros clientes y poder comunicarse con otros clientes, en la figura 2.3 se puede visualizar la organización de una botnet P2P.

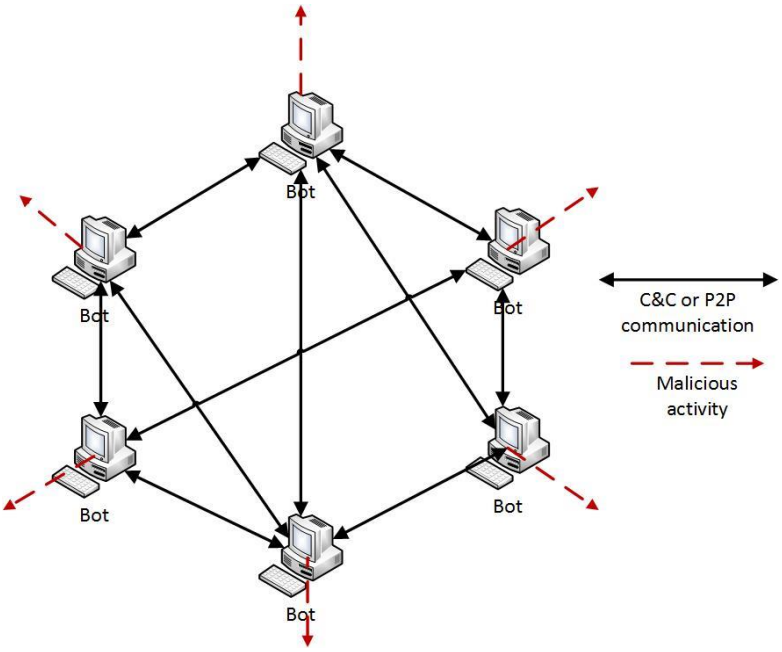


Figura 2.3 Botnet P2P

Se dice que un grupo de peers está bien conectado si al menos una de las siguientes declaraciones se cumple:

1. Exista una conexión entre dos peers, de tal manera que cada uno pueda conectarse a cualquier otro elemento requerido.
2. Que haya relativamente una pequeña cantidad de conexiones que atraviesan entre un par de peers. Cuando sea removido un peer, no se evitará que los peers se conecten entre ellos.
3. Esto no significa que un solo equipo deba conectarse a todos los elementos de la red, de hecho, basta una pequeña cantidad para conocer los peers disponibles.

Este tipo de arquitectura ha sido adoptada por los atacantes debido que las botnets tienen mayor flexibilidad para adquirir un mayor número de bots y lograr el máximo beneficio. La detección de este tipo es difícil por los siguientes motivos:

- Si se quiere eliminar una botnet por completo, descubrir todas las bots asociadas. Teniendo que destruirla completamente.
- Es difícil diagnosticar toda el área afectada por una botnet.
- Se complica la desactivación de la botnet debido a la interdependencia entre bots.

La ventaja que tiene este tipo de arquitectura, radica en que se evita un único punto de fallo y los atacantes pueden utilizar puertos HTTPS para ocultar códigos maliciosos cifrados de firewalls externos o filtros. Por otro lado, las botnets P2P son lentas en la convergencia y la respuesta, difíciles de manejar y no escalable.

2.5. Ciclo de vida de las botnets

El ciclo de vida de las botnets empieza cuando el botmaster configura los parámetros de la bot. Esta etapa se la denomina inicialización. A continuación, se asigna una dirección IP al botmaster, a esta etapa se le denomina registro. En la etapa de infección preliminar se utiliza la propagación de virus, descargas no autorizadas, descargas de archivos inusuales y ejecución de archivos maliciosos, lo cual, infecta los dispositivos electrónicos vulnerables. Con este paso se da inicio a la siguiente etapa, la cual consiste en la construcción de la red de bots más conocida como botnet debido a la instalación de malware en cada host, este proceso de descarga ocurre a través de protocolos HTTP, FTP o P2P.

Esta etapa se llama, fase de conexión. Se establece la comunicación entre bots y sus C&C, y cada vez que se reinicia un bot se asegura su establecimiento de la conexión entre el botmaster y los bots, para que sean partícipes de la botnet y de esta manera recibir comandos para tomar acciones [27]. La siguiente etapa se conoce como iniciación de ataque. En esta etapa los bots esperan los comandos de sus C&C e inician las actividades maliciosas, ataques DDoS, captura de tráfico de red, spamming, propagación de malware, captura de actividad de teclados y robo de credenciales de identidad.

Finalmente, la etapa actualización y mantenimiento es de gran importancia para el botmaster ya que mantiene sus bots conectados para futuros ataques y debe asegurarse que los cambios de código o de servidores C&C se realicen lo más pronto posible para así evitar que se detecte la vulnerabilidad.
En la figura 2.4 se visualiza el ciclo de vida de una botnet.

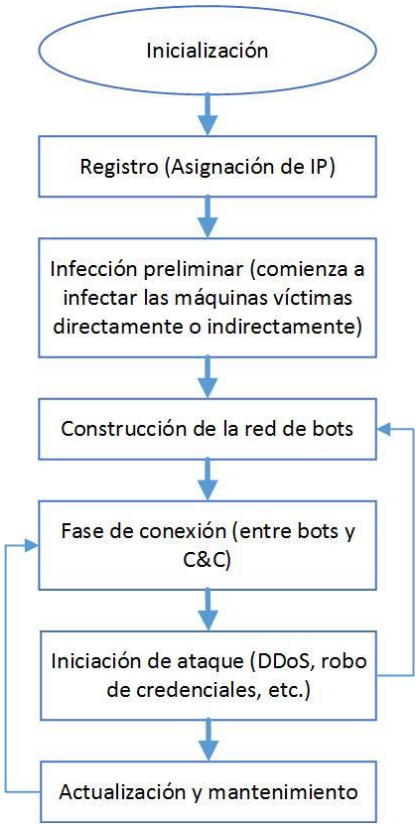


Figura 2.4 Ciclo de vida de una botnet

3. ESTUDIO COMPARATIVO DE LAS TÉCNICAS DE DETECCIÓN DE BOTNETS

Este capítulo aborda un análisis comparativo de las técnicas de detección de vulnerabilidades tipo botnet. Se empieza con un listado de las técnicas existentes y una descripción de las características de cada una de ellas. Después de haberlas enumerado, se procede con la selección de una de ellas la cual permite realizar el análisis de la vulnerabilidad tipo botnet.

Actualmente la detección de botnets es un reto constante para los investigadores y organizaciones. Se deben considerar todos los aspectos incluyendo la detección, mitigación y respuesta. Estos aspectos son cambiantes en el tiempo, por lo tanto, ninguna técnica de mitigación o detección ofrece una solución permanente. Del mismo modo, diferentes tipos de actores, por ejemplo, empresas, gobiernos, redes y proveedores de servicios Internet (ISP), tienen diferentes formas y objetivos al abordar el tema de redes de bot. Por esa razón se realizará un análisis para seleccionar la técnica más adecuada para nuestro caso de estudio.

Entre las técnicas existentes para la detección de vulnerabilidades tipo botnet se tienen las siguientes: honeypot, monitoreo de tráfico de red pasiva y otras técnicas, las cuales se muestran en la figura 3.1

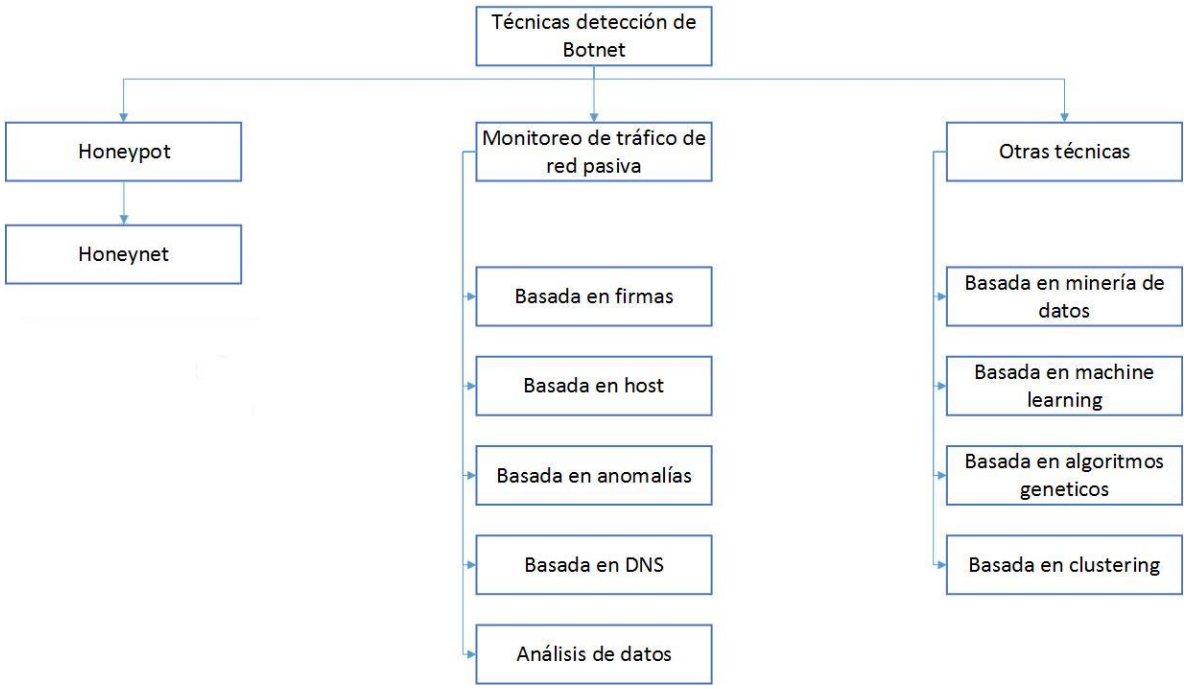


Figura 3.1 Resumen de las técnicas detección de Botnet

3.1. Honeypot

Un honeypot es un recurso de red que se configura para ser probado, atacado e inclusive comprometido. Cuando un honeypot es comprometido este puede capturar la actividad del atacante de acuerdo a direcciones IP, puertos usados, protocolos usados, entre otros. Esta información es muy valiosa para conocer patrones o estrategias al momento de intentar vulnerar la seguridad de un sistema en la red para realizar actividades maliciosas [31]. Existen varias formas de clasificar a los honeypots de acuerdo a su uso y a la forma de implementación. Así se tiene, en primer lugar, dos clasificaciones importantes: honeypots de producción y honeypots de investigación. Existe una última clasificación de acuerdo al nivel de interacción del honeypot con el atacante: honeypots de alta o baja interacción.

3.1.1. Honeypots de Producción

Estos honeypots se ubican en la red de producción de una organización, proporcionando servicios similares a dicha red. Como objetivo, el honeypot busca desviar el riesgo de un ataque de la red y ayudar a asegurarla de acuerdo a las actividades que los intrusos realicen en el honeypot. Si el intruso obtiene de alguna forma el control del honeypot de producción, este puede bloquear las conexiones salientes, limitando el accionar del atacante solo al interior del mismo. El honeypot de producción captura y defiende.

3.1.2. Honeypots de Investigación

Su propósito es ser atacado, pero con el objetivo de ser también una herramienta didáctica que permita aprender de patrones y estrategias de los atacantes y si se requiere, para formular estrategias de defensa de los sistemas contra amenazas nuevas o existentes. Es usado en centros académicos, sectores gubernamentales, etc. Este honeypot solo captura información para ser analizada [32]. Suele tener su propia conexión hacia Internet, totalmente aislado, evitando así la necesidad de defender la red de producción de posibles ataques que se produzcan desde el honeypot y permitiendo solo la captura información.

3.1.3. Honeypots de alta y baja interacción

Los honeypots tanto de producción como de investigación pueden ser de alta o baja interacción, de acuerdo al grado de interacción con los atacantes y el riesgo que eso implica, tal como se describe en la tabla 3.1

Honeypots de Alta Interacción	**Honeypots de Baja Interacción**
Implementación de servicios en SO reales	Simulaciones de servicios y SO
Alto riesgo	Bajo riesgo
Su implementación es más difícil	Fáciles de implementar y mantener
Mayor cantidad de datos recolectados	Menor cantidad de datos recolectados

Tabla 3.1 Características de honeypots de alta y baja interacción

3.2. Honeynet

Son un tipo de Honeypots, que actúa sobre una red entera. En este caso se usan equipos, sistemas operativos reales y por ende aplicaciones reales. Se utiliza para comprobar los ataques, la forma en que atacan y las nuevas técnicas que afectan a la red. Normalmente está compuesto por un solo sistema en el que emula servicios conocidos o vulnerabilidades, o crea entornos cerrados. Una honeynet se utiliza para capturar datos en la formación de las botnets para su posterior análisis, analizar la tecnología utilizada, sus características y la intensidad del ataque. Por otra parte, la información recopilada de los bots se utiliza para descubrir los sistemas C&C, susceptibilidades desconocidas, técnicas y herramientas utilizadas por el atacante.

Con las honeynets se atendería dos aspectos importantes:

- El Control de datos: define el grado de vulnerabilidad de los sistemas expuestos y sea apetecible al atacante.
- Captura de datos: permite establecer que herramientas y métodos se utilizan para el monitoreo de las actividades realizadas en el honeynet, donde se almacenarán las actividades de la red y establecer los perfiles de los atacantes [33].

Existen varias técnicas para capturar bots en una honeynet, sin embrago los atacantes han desarrollado métodos para evitar la caer en la trampa. Por esa razón se hace uso de un componente llamado honeywall que se utiliza para separar los honeypots del resto del mundo, este componente actúa como una pasarela del tráfico de red, se puede observar en la figura 3.2

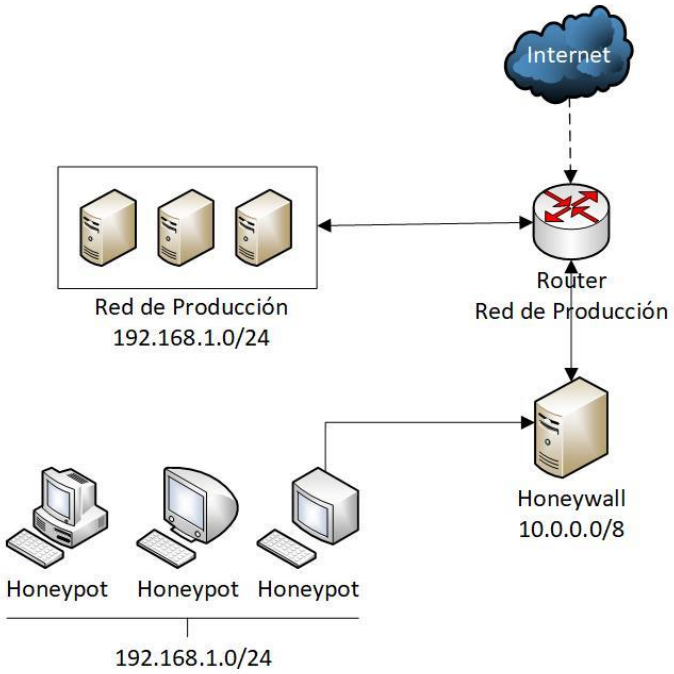


Figura 3.2 Arquitectura honeynet

Sin embargo, los aspectos críticos de una honeynet incluyen:

- Escalabilidad limitada, ya que requiere equipos de hardware intensivo (enrutadores de puerta de enlace y el sistema de la honeynet implementado).
- Honeypots no puede prever ataques de Internet y los sistemas solo pueden rastrear actividades maliciosas al interactuar con él.
- Descubrir los sistemas infectados, como una trampa también es un desafío.
- Los atacantes podrían controlar la honeypot para dañar otros sistemas fuera de la honeynet.

3.2.1. Honeynet de primera generación

Las honeynet de primera generación están conformadas por los honeypots, por un firewall que separa a la honeynet de Internet, de la red de producción y de un segmento de administración o monitoreo, un IDS y un router. Todo paquete que va hacia o desde la honeynet debe pasar por el firewall y el router. En la figura 3.3 se muestra la arquitectura de una honeynet de primera generación.

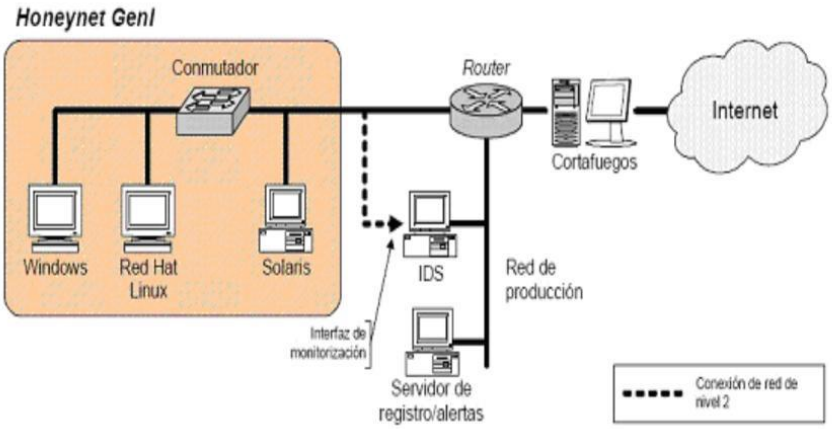


Figura 3.3 Honeynet de primera generación **[34]**

3.2.2. Honeynet de segunda generación

Lo que se busca en las honeynets de segunda generación es hacer un sistema de fácil despliegue y que sea cada vez más difícil de detectar por parte de los atacantes o intrusos, además estos requerimientos deben ser combinados en un solo dispositivo. La parte principal de la arquitectura de una honeynet Gen II es el honeywall, que es un gateway que separarla del resto de la red de producción. Todo el tráfico que se dirige hacia los honeypots debe pasar por el honeywall, siendo este último, además un dispositivo totalmente transparente para quienes interactúen con los mismos. La arquitectura de la honeynet de segunda generación se muestra en la figura 3.4

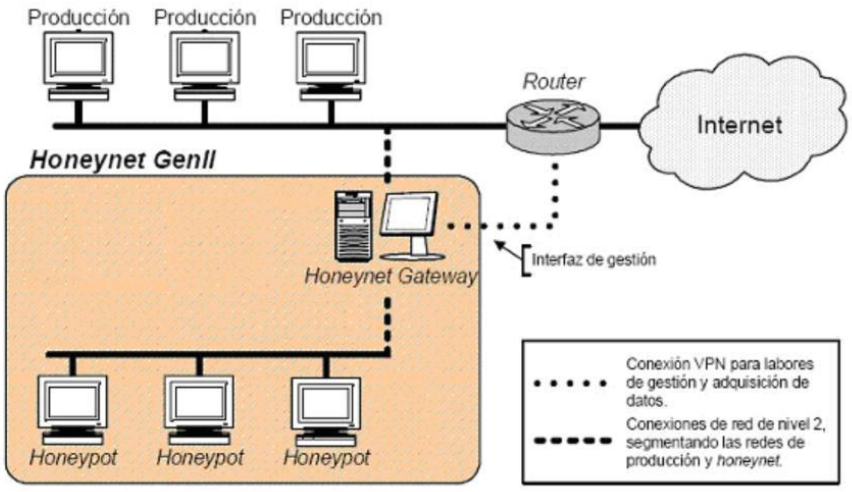


Figura 3.4 Honeynet de segunda generación **[34]**

3.2.3. Honeynet de tercera generación

La arquitectura de una honeynet de tercera generación es similar a la de segunda generación. El esquema de control de datos es el mismo, pero en la captura de datos se implementa una mejora. Se agregan herramientas que ayudan a una mejor identificación de los eventos que suceden dentro de la honeynet. En la segunda generación, se generaban varios formatos de archivos que indicaban las actividades en la honeynet. Se tenía, por un lado, los archivos pcap generados en el honeywall y por otro los logs del firewall y de los honeypots. Estos archivos debían ser relacionados unos con otros para determinar el camino que tomaron los paquetes dentro de la honeynet. Es decir, se hacía el análisis por separado, lo cual conllevaba más tiempo y esfuerzo. Es válido decir que estas honeynets pueden ser virtuales, tal como lo muestra la figura 3.5

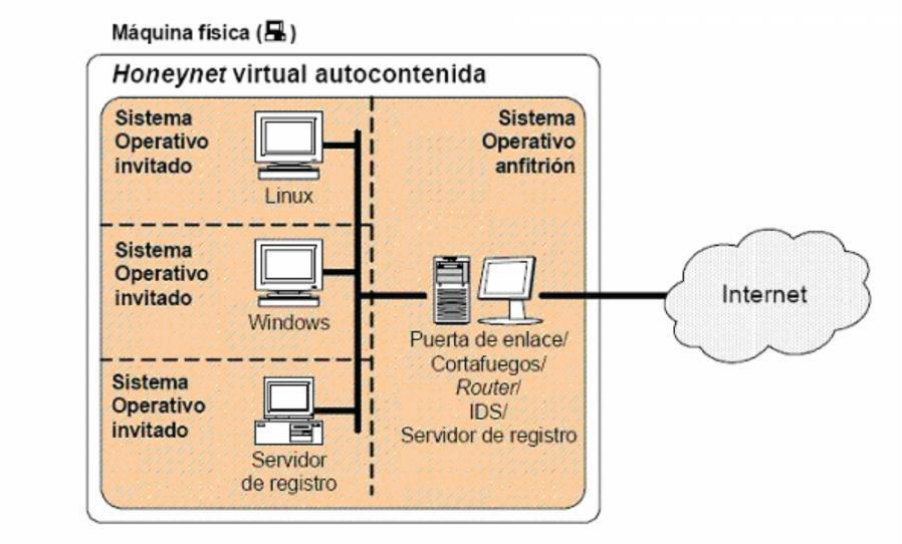


Figura 3.5 Honeynet virtual **[34]**

3.3. Monitoreo de tráfico de red

El monitoreo de tráfico de red pasiva puede clasificarse en basada en firmas, basada en host, basada en anomalías, basada en DNS, y análisis de datos [35].

3.3.1. Técnica basada en firmas

Es útil para las botnets conocidas ya que permite identificarlas mediante sus características o su firma, pero no es funcional para botnets desconocidas. Para detectar el malware existente en un host, se realiza un análisis interno del comportamiento y recursos del host. La manera de detectar la presencia de malware en un host es obteniendo el hash de los archivos del sistema, lo que permite conocer si intentaron o si fue vulnerada su integridad.

Se puede aplicar un sistema de detección de intrusos (IDS). Un IDS es una aplicación de software para monitorear los servicios del sistema en busca de actividades malintencionadas o infracciones de políticas de seguridad e informar estas violaciones a la administración. Requiere actualizaciones frecuentes del repositorio de la base de conocimientos de firmas para detectar botnets recién activados. Sin embargo, la frecuencia de actualización de firmas de IDS es pequeña por la razón básica de que las anomalías aumentan rápidamente. Los diagnósticos se realizan lentamente.

En contraste, las firmas para tales ataques maliciosos no se crean al mismo ritmo. Por lo tanto, es posible que los ataques de la red de bots no se detecten y que no funcione con redes de bots desconocidas. La principal desventaja de este tipo de técnica es que el IDS requiere actualizaciones frecuentes del repositorio de firmas. Por otra parte, la técnica basada en firmas ignora botnets idénticas con firmas ligeramente diferentes. En la tabla 3.2 se muestran un resumen de la técnica basada en firmas.

Modelo Propuesto	Metodología	Breve resumen
Un framework para caracterizar el comportamiento en una red de protocolo Ethernet	Conformidad con la firma y detección de cambios de comportamiento, detección del comportamiento periódico y sincrónico basado en la aproximación de las K-means (un tipo de regla)	La contención de que las botnets pueden evadir la detección al alejarse de la arquitectura C&C basada en IRC y la base de conocimientos, correlación automática del flujo, huellas dactilares de comportamiento.
Detectores de intrusos como: Snort, Suricata, Bro	Implementa un lenguaje de creación de reglas flexibles, potentes y sencillas. Durante su instalación, provee de cientos de reglas para backdoor, DDoS, finger, FTP, ataques web, CGI, Nmap, entre otros. Así como la detección de intrusión en tiempo real (IDS), prevención de intrusión en línea (IPS), monitoreo de seguridad de red (NSM) y procesamiento de pcap fuera de línea.	Ofrece la capacidad de almacenamiento de bitácoras en archivos de texto y en bases de datos abiertas. Implementa un motor de detección de ataques y escaneo de puertos que permite registrar, alertar y responder ante cualquier anomalía previamente definida.

Tabla 3.2 Resumen técnicas basadas en firmas

3.3.2. Técnica basada en host

Se utiliza una técnica basada en host para comprobar si la máquina individual es infectada por el bot o no. Cada bot afecta la máquina individual cambiando su estructura de registro, llamadas de sistema y archivos de sistema. Una de las ventajas de utilizar esta técnica es que fácilmente puede evitar descargar ataques y especialmente los de arranque de sistema. La protección a nivel de host puede proporcionarse escaneando máquinas individuales en una organización; sin embargo, esto se considera una tarea costosa y que consume mucho tiempo. En la tabla 3.3 enseña un resumen de las técnicas de detección botnets existentes basadas en host:

Modelo Propuesto	Metodología	Breve resumen
Modelos de Markov no estacionarios **[36]**	Detección basada en host mediante monitoreo de llamadas al sistema	Correlaciona las instancias de gusanos en el destino.
Comportamiento del control remoto de los bots **[37]**	Contaminación basada en contenido y basada en subcadenas	Exhibe raramente el comportamiento de control externo.
BotSwat **[37]**	Supervisa la biblioteca de Win32	No considera el protocolo de comunicación C&C o la estructura específica de la red de bots.
Minería de múltiples archivos de registro	Detección basada en flujo a través de minería de datos	Cuestiones de privacidad y seguridad
BotTracer **[38]**	Modelo trifásico / basado en flujo	No se pueden detectar máquinas virtuales
Sistema de detección de bot multiagente (MABDS) **[39]**	Modelo híbrido (host IDS + analizador de registro de eventos OS)	Nuevo problema de actualización de firma
HIDS **[40]**	Modelo híbrido (analizador de archivos de registro + red neuronal BP)	No escalable
DeWare	Hace cumplir las reglas sobre rutinas de SO	Las rutinas del sistema operativo de nivel de kernel podrían ser interceptadas
Sistema de detección de intrusión inteligente híbrido (HIIDS) **[41]**	Aplicación de lógica difusa a través de perfiles de red, modelo híbrido	Aprendizaje autónomo de las reglas de inferencia.
Kuppusamy, Técnicas de muestreo estratificado/aleatorio	Estabilidad, el equilibrio entre la precisión de detección y la sobrecarga de ancho de banda incurrida	Utilizado específicamente para MANET
Un enfoque semántico para la detección de intrusiones basada en host **[42]**	Basado en patrones de llamada de sistema contiguos/no contiguos	Se necesita investigar el proceso de transferencia.

Tabla 3.3 Resumen de técnicas de detección basada en host.

3.3.3. Técnica basada en anomalías

Trata de encontrar botnets basado en el comportamiento inusual de una red, por ejemplo: grandes volúmenes de tráfico de red, alta latencia de red, uso de puertos inusuales y comportamiento fuera de lo normal de equipos de la red. Este tipo es popular gracias a que los distribuidores de malware difunden actividades maliciosas y anómalas alrededor del mundo en un entorno de red controlada. Esta técnica no es escalable debido a que todos los equipos deben estar dotados de herramientas de supervisión efectivas, como antivirus y software de detección de spam; en la red, se inyectan paquetes a la red para medir el tiempo de respuesta, mientras el tráfico de la red para a través de dispositivos especializados de hardware y de este modo detectar actividades sospechosas.

Produce un bajo índice de falsos positivos, basándose en el servidor tratando de reducir las falsas alarmas positivas en la red. Se combinan dos perspectivas para la detección de anomalías, incluyendo detección de anomalías de nivel de host y la proliferación de la tasa de falsos positivos. En la tabla 3.4 se presenta un resumen de las técnicas basadas en anomalías.

Modelo Propuesto	Metodología	Breve resumen
BotSniffer	Detección de anomalías basada en la red, detección de servidores C&C y máquinas infectadas Correlación temporal espacial	No se puede escanear la comunicación cifrada y tiene soporte de más protocolos excepto IRC y HTTP.
BotHunter	Detección de infección de malware a través de la correlación de diálogos impulsada por IDS	Trabaja incorporando lógica de estado adicional, adaptándose a amenazas emergentes, adversarios y maximización de las historias de diálogo locales.
Monitoreo continuo de similitud	Marco de detección de botnet en línea en tiempo real enviando byte por paquete y cantidad de paquete.	Tiene la dificultad para calcular similitudes entre grandes flujos de anomalías que se actualizan continuamente

Tabla 3.4 Resumen técnicas basadas en anomalías

3.3.4. Técnica basada en DNS

Tiene la desventaja de no funcionar en todo tipo de botnets (conocidas o desconocidas) debido a que el botmaster puede cambiar de DNS constantemente. Los bots realizan consultas DNS para localizar un servidor C&C, por lo tanto, se puede identificar el tráfico botnet DNS al monitorear el tráfico. Existen algunas técnicas de detección basadas en DNS, las cuales se describen a continuación. Las técnicas de detección basadas en DNS utilizan la información del DNS que es compartida por el botnet y C&C. Es necesario localizar el servidor del botnet para comunicarse con sus bots.

Los bots emiten consultas DNS para localizar el servidor C&C. Durante esta etapa, se proporciona un mecanismo de detección para analizar el tráfico de DNS y detectar posibles inestabilidades y anomalías de comunicación DNS. Normalmente los bots se comunican con un único dominio administrativo y es fácil de medir la relación entre los bots y el mecanismo de C&C mediante el análisis de atributos de dominio diferente como la vida útil del dominio, tiempo de vida (TTL) de la consulta, page ranking de dominios.

Se deben trazar flujos de DNS para identificar a agentes incluyendo clientes y servidores DNS. Se utilizó un gráfico dirigido donde los nodos representan direcciones IP de las máquinas del servidor DNS y consultas originadas por los clientes. Una blacklist basada en DNS recoge direcciones IP publicadas en los servidores o redes, con la suposición de que están involucrados en actividades de maliciosas y spam. Es un intento de captar el botmaster e identificar su ubicación. Sin embargo, los aspectos críticos de este enfoque requieren una versión actualizada de la blacklist basada en DNS.

Un inconveniente de este enfoque es que, conociendo los mecanismos, el botmaster puede fácilmente evitar este esquema o incluso la suspensión de trabajo. El botmaster puede interrumpir este esquema mediante la aplicación de consultas DNS falsas masivas, que conduce a la creación de un número de falsas alarmas.

El detector de actividad de grupo de botnet o BotGAD, es un esquema de detección de anomalías basado en comportamiento del grupo de monitoreo mediante el uso de tráfico DNS. Ofrece características especiales para el tráfico DNS, permite mecanismos de detección en redes a gran escala, así como en entornos de tiempo real. Además, se emplea un mecanismo para la migración del servidor de C&C de botnet. De acuerdo a que una cabecera de IP es la fuente para obtener información de DNS, los botnets con canales de

comunicación encriptado son fácilmente trazados por recoger la información de la cabecera. La desventaja de esta técnica es que incurre en un gran tiempo de procesamiento en la vigilancia del tráfico de red. En la tabla 3.5 se presenta un resumen de las técnicas de detección basadas en DNS.

Modelo Propuesto	**Metodología**	**Breve resumen**
BotGAD	Consiste en cuatro partes. 1. Recolector de datos. 2. Clasificador grupal 3. Analizador de similitudes 4. Reportes	Permite detectar botnets desconocidos de redes a gran escala en tiempo real. Se implementa utilizando el tráfico DNS y se muestra la efectividad de los experimentos sobre la base del modelo de actividad de grupo y la métrica.
Modelado de la propagación de una botnet usando zonas horarias	Permite detectar botnets en tiempo de lanzamiento y enfoque regional, específicamente para HTTP y P2P	Puede predecir futuras botnet, pero no puede determinar botnets centralizadas de tipo Comand&Control.
Detección a través del análisis de tráfico DNS	Separa el tráfico DSN anómalo del tráfico DNS normal usando métricas como: TTL, CS_NS, DDNS_NS, CSAA	Si la botnet aumenta de tamaño, se vuelve ineficaz.
Detección basada en la observación de actividades grupales en DNS	Teniendo en cuenta el grupo se activa en el tráfico DNS a través de consultas DNS.	La detección a gran escala debe ser factible pero el tiempo de procesamiento es grande para una encuesta a gran escala.
Consulta activa de cachés de DNS	Detección de botnet al inferir el patrón de uso de las aplicaciones.	Sirve como soporte para otras aplicaciones, realiza extracción de registros. Pero no proporciona información de uso percibida de registros o rastreos de paquetes

Tabla 3.5 Resumen de técnicas de detección basada en DNS [43]

3.3.5. Técnica de análisis de datos

Es la técnica más abarcadora de todas. La actividad de los hosts en una red es monitoreada y registrada. Se examinan todos los archivos de registro disponibles, relacionando los registros con posibles eventos, como un incremento notorio en el tráfico de la red, para obtener más información. La ventaja de esta técnica sobre las demás es la capacidad de ser proactiva ya que permite la implementación de una combinación de enfoques de detección y la versatilidad en el software utilizado. Básicamente se pueden combinar varias técnicas para potencializar la detección del tipo de vulnerabilidad propuesto.

3.4. Otras técnicas

3.4.1. Técnica de minería de datos

Utiliza un sistema de puntuación y análisis ngram [35] para tráfico de servidores IRC. Esta técnica no puede detectar botnets IRC ni comunicaciones encriptadas, pero se pueden detectar mediante la extracción de múltiples archivos de registro correlacionada a registros de tráfico C&C. Existen herramientas que facilitan la recopilación, el análisis y el uso de los big data generados por una infraestructura tecnológica, en este caso los archivos de query-logs del servidor DNS, con las facilidades y plugins de este software se puede proporcionar una manera prometedora de lidiar con problemas de seguridad como la detección de botnets. La tarea de detección se completa con el desarrollo de un proceso que analiza los tiempos de conexión, números de conexiones muertas y consultas a bases de datos de botnets descubiertos.

La detección de botnets P2P basado en una técnica de minería de datos para analizar el comportamiento de la red a nivel de gateway. Un aspecto significativo de este enfoque es que puede monitorear el tráfico cifrado de red. Sin embargo, este esquema es para infraestructuras de red en pequeña escala (LANs) y no puede ser desplegado o validado en las redes a gran escala. Del mismo modo, utiliza el mecanismo de traducción de direcciones de red (NAT) para enrutar el tráfico de red; sin embargo, NAT no es eficiente en la detección de flujos de red de P2P. La tabla 3.6 presenta un resumen de las técnicas de detección basado en minería de datos.

Modelo Propuesto	Metodología	Breve resumen
BotMiner	Extensión de BotSniffer C-Plane: detección de C&C A-Plane: detección de actividades maliciosas.	Trabaja tanto en entornos centralizados como descentralizados. Tiene soporte SMTP y HTTP, pero No está diseñado para tipos especiales de protocolos.
Técnica basada en flujo para minar múltiples archivos de registro	Correlación temporal entre archivos de registro (tcpdump, exedump). Y máquinas de vectores, árboles de decisión, modelo bayesiano, árboles de decisión potenciados	Implementación en registros de nivel de sistema, experimento en tiempo real pero no rastrea los flujos de IRC.
Detección de botnet P2P utilizando un esquema de minería de datos	Un método de detección de botnet P2P que se basa en la supervisión del tráfico en la puerta de enlace y el uso de la tecnología de extracción de datos para analizar el comportamiento de la red. Posee paquetes encriptados Mecanismo de preaviso Detección sofisticada del flujo de botnets.	Funciona solo dentro de un entorno LAN; debe distribuirse a nivel de ISP para detectar redes de bots P2P en una red a gran escala. La existencia de la tecnología NAT dificulta la detección de flujos P2P y no se recomienda en el diseño de redes a gran escala para una mejor detección de botnets.

Tabla 3.6 Resumen de técnicas de detección basada en minería de datos

3.4.2. Técnica basada en machine learning

Permite obtener soluciones para toda la red. Debido a la naturaleza de ataques de botnets, ninguna defensa individual o mecanismo es suficiente para prevenirlos o detectarlos. Por esa razón se realiza un enfoque de defensa en profundidad. Una combinación de dos clases de contramedidas, a saber, honeypots y detectores basados en red. Si bien los honeypots se usan para detectar intentos de intrusión, los mecanismos de detección basados en la red pueden identificar, a través del análisis del comportamiento, bots que coexisten con máquinas benignas. Tanto los honeypots como los detectores basados en la red se pueden tratar como recursos disponibles para el defensor, pero un defensor generalmente tiene limitaciones en la cantidad de recursos disponibles. Para desplegar estos mecanismos de

manera óptima y dinámica, se emplea un modelo de aprendizaje de máquina con el objetivo de reducir la vida útil de botnets sigilosas en un entorno de recursos limitados [44].

El aprendizaje de máquina es un método algorítmico para resolver problemas secuenciales de toma de decisiones en el que un agente (o persona que toma decisiones) interactúa con el entorno para aprender a responder en diferentes condiciones. Formalmente, el agente busca descubrir una política que mapee el estado del sistema a una acción óptima. De forma particular, el agente monitorea el comportamiento de los hosts con respecto a las sesiones de escaneo y salientes dentro de diferentes subredes para guiar la ubicación de los mecanismos de defensa. En la tabla 3.7 se detalla un resumen de las técnicas de detección botnets existentes basadas en machine learning.

Modelo Propuesto	**Metodología**	**Breve resumen**
Aprendizaje incremental de LS-SVM	Detección de comunicación de botnet encriptada utilizando las direcciones IP del servidor	Se utiliza para diversos experimentos de aplicación centrándose en el algoritmo de aprendizaje en línea
Bloqueo de spam al separar las máquinas de los usuarios finales con las máquinas legítimas del servidor	Máquina de vectores de soporte (SVM). Características de la máquina: OS, nombre de host léxico	Debe basarse en conjuntos de datos diversos y grandes, pero tiene problema con el pequeño conjunto de datos, indeseable para servidores de correo
Uso de aprendizaje automático para detectar botnet	Un enfoque de dos etapas para la detección de botnets IRC. Flujo de protocolo IP, indicadores TCP, paquetes enviados, duración, rol, etc.	Se requiere un etiquetado preciso para obtener mejores resultados, pero el criterio de etiquetado no es exacto.
Detección basada en C&C estrecho	Un enfoque proactivo para detectar botnets IRC usando el ancho de banda, duración y tiempo de paquetes	Funciona correlacionando el tráfico periódicamente.

Tabla 3.7 Resumen de técnicas de detección basada en machine learning

3.4.3. Técnica basada en algoritmos genéticos

Estas técnicas analizan cierta información, como por ejemplo las características del tráfico de red, para inferir los patrones de comportamientos únicos de las botnets. Se debe realizar captura de tráfico normal y tráfico de botnets, y separar los paquetes en ventanas temporales y definir un juego de características a obtener en cada ventana temporal [45]. Se utilizan los algoritmos genéticos para encontrar los mejores umbrales de estas características. Una vez definidos los umbrales, se realizan comprobaciones con capturas normales para conocer cuán bien se comportaban. Este tipo de técnica abre las puertas a un tratamiento más claro del comportamiento de las botnets.

La ventaja de la utilización de algoritmos genéticos es que puede generar automáticamente reglas que clasificarán las conexiones de red en anómalas o normales. El sistema al ser planteado no toma decisiones directamente, sino que se utiliza como apoyo a la toma de decisiones. Las conexiones de red son capturadas por un IDS y clasificadas por el algoritmo en base a ciertas características, para seleccionar el juego de características que mejor reduce el error causado por decisiones basadas en la experiencia. En la tabla 3.8 se presenta un resumen de las técnicas de detección botnets basadas en algoritmos genéticos.

Modelo Propuesto	Metodología	Breve resumen
TreeMap y técnica de representación gráfica.	Análisis de NetFlow usando TreeMap y representación gráfica.	Recopilación y análisis del uso del servicio, detección de ataques distribuidos e investigación.
Aprendizaje incremental de LS-SVM	Detección de comunicación de botnet encriptada. Direcciones IP del servidor	Se centra en los algoritmos de aprendizaje en línea y en la experimentación diversa de aplicaciones.
BotCop	Características temporales frecuentes de los flujos de red basados en n-gramas.	Etiquetar los clusters es una tarea desafiante y mucho más en entornos P2P.
Un marco para la detección basada en DNS	Detección basada en entradas de DNS maliciosas utilizando el clasificador de árbol de decisión C5.0 y medidas estadísticas	Problema de actualización de la blacklist mediante algoritmos. Implementación de clasificadores de aprendizaje.

Tabla 3.8 Resumen de técnicas de detección basada en algoritmos genéticos

3.4.4. Técnica basada en clustering

En la detección de bots se examina características como IP, puerto, tiempos de eventos de paquetes y bytes por paquete para evidenciar la actividad de botnets. Primero se recopila los flujos y los registros basados en el filtrado básico, la whitelist y la blacklist. Los registros restantes producen un clúster y este se refina en función de patrones, políticas y otro clúster que se generó en función de los eventos informados, las alertas y las actividades de los sensores de seguridad de la red.

Se aplica clustering jerárquico que permite construir un dendrograma, es decir, un árbol como gráfico que codifica las relaciones entre los bots. El clúster fusionado se modifica en función de las reglas y se combina con otra información sobre los nodos infectados detectados para reducir los falsos positivos. En la tabla 3.9 se muestra un resumen de las técnicas de detección botnets existentes basadas en clustering:

Modelo Propuesto	Metodología	Breve resumen
Rastreos DNS recursivos (RDNS) **[46]**	Técnica pasiva para detectar redes de flujo malicioso yd ominios de flujo rápido	Asegura el enrutador perimetral del acceso no autorizado a recursos internos, pero degrada el rendimiento del mismo.
Detección automatizada en una red a gran escala	Un marco jerárquico para descubrir automáticamente las botnet aplicando las firmas de carga útil, algoritmo de agrupamiento asociativo cruzado, características de flujos temporales frecuentes.	Teniendo en cuenta P2P, SMTP y la comunidad de correo electrónico, pero solo funciona para IRC y la comunidad web.
Clúster de comportamiento **[47]**	Caracterización del comportamiento del nodo considerando conjuntamente correlaciones espaciales, temporales y una tasa de falsos positivos	Caracterización del tráfico, agrupación de variantes de tiempo, más trazas de tráfico. Pero baja detección y perfil de comportamiento en diferentes momentos.
SBotMiner **[48]**	Detección de botnet a partir de los registros del motor de búsqueda través de archivos de registro, direcciones IP, consultas	Trabajando para tráfico de búsqueda anormal pero no se pueden detectar diversas solicitudes de búsqueda.

Tabla 3.9 Resumen de técnicas de detección basada en Clustering

3.5. Análisis comparativo y selección de la técnica de detección de botnets

Con la selección de las técnicas de detección de botnets, necesario para el análisis comparativo. Se procederá a detallar las características principales, indicando las ventajas y desventajas para así hacer uso de una técnica con la cual se trabajará más adelante. En base a la comparación realizada, se ha elegido utilizar la técnica de “Análisis de datos” para este proyecto. Es la más completa de todas ya que se pueden combinar varias técnicas de detección y por la versatilidad del software que puede ser utilizado. Se detalla en la tabla 3.10

Técnica / Característica	Honeynet	Basada en firmas	Basada en host	Basada en anomalías	Basada en DNS	Análisis de datos	Basada en minería de datos	Basado en machine learning	Basada en algoritmos genéticos	Basada en clustering
Fortalezas	-Fácil de mantener -Poderosa herramienta para encontrar puntos débiles en la red	-Se puede conocer los intentos de vulnerar su integridad. -Envió de alertas cuando se intenta vulnerar su integridad	-Evita ataques en el arranque del sistema. -Las rutinas de nivel de kernel podrían ser interceptadas.	-Analiza el tráfico de red por un determinado periodo de tiempo y crea una línea base de comparación	-Se proporciona un mecanismo de detección para analizar el tráfico de DNS y detectar posibles inestabilidades y anomalías de comunicación DNS.	-Se examinan todos los registros con los eventos posibles. -Técnica proactiva -Se pueden combinar otras técnicas para detección de vulnerabilidades	-Facilitan la recopilación, el análisis y el uso de los big data generados por una infraestructura tecnológica	-Permite obtener soluciones para toda la red debido a que realiza un enfoque de defensa en profundidad. -Reducir la vida útil de botnets sigilosas en un entorno de recursos limitados gracias al aprendizaje de máquina.	-La ventaja de la utilización de algoritmos genéticos es que puede generar automáticamente reglas que clasificaran las conexiones de red en anómalas o normales	-Clustering jerárquico que permite construir un dendrograma, es decir, un árbol como gráfico que codifica las relaciones entre los bots

Debilidades	-Único punto de fallo-Recursos limitados-Se comprometa la seguridad de la red -Políticas de seguridad mínimas -Espera de un atacante (pasividad)	-Requiere una continua actualización de firmas. -Los diagnósticos se realizan lentamente.	-No puede detectar máquinas virtuales.-No escalable	-Genera muchos falsos positivos-Definición ambigua de patrones de tráfico-No escalable	-Es necesario localizar el servidor del botnet para comunicarse con sus bots, el botmaster puede cambiar constantemente el DNS-Requiere una versión actualizada de la blacklist basada en DNS		-No puede detectar botnets IRC ni comunicaciones encriptadas	-Pequeño conjunto de datos, indeseable para servidores de correo electrónico de pequeñas empresas-El criterio de etiquetado no es exacto.	-Los algoritmos genéticos pueden hacer uso de valores fuera del umbral definido para botnets, por lo tanto, detectaría otro tráfico distinto del que interesa.	-Degradación del rendimiento-No se pueden detectar diversas solicitudes de búsqueda-Trabajando solo para IRC y la comunidad web

Control de datos	Si	No	Si	Si	No	Si	Si	Si	Si	Si
Captura de datos	Si	Si	No	Si	No	Si	Si	Si	Si	SI
Análisis de datos	Si	No	No	Si	No	Si	Si	Si	Si	SI
Identificación de botnets desconocidas	Funcional	No funcional	No funcional	No funcional	No funcional	Funcional	Funcional	Funcional	Funcional	Funcional
Integración con IDS	Si	Si	Si	No	No	Si	No	Si	Si	No
Factibilidad Técnica	Difícil	Promedio	Promedio	Promedio	Promedio	Dependiendo de las técnicas combinadas	Promedio	Difícil	Difícil	Difícil
Factibilidad Operacional	Difícil	Promedio	Promedio	Promedio	Difícil	Promedio	Promedio	Promedio	Promedio	Promedio
Factibilidad Económica	Costos reducidos si se virtualiza la técnica	Costos reducidos	Altos costos, debido a que se debe scannear	Costos similares a la técnica basada en host	Costos altos, ya que incurre en un gran tiempo de	Dependiendo de las técnicas combinadas	Costos altos, debido a que esta técnica debe implementarse en	Costos intermedios	Costos reducidos	Costos intermedios

			cada host individual mente		procesam iento en la vigilancia del tráfico de red		sistemas en tiempo real.			
Portabilidad	Si	No	No	No	Si	Si	No	No	No	No
Criterio de selección	Técnica antigua, dificultad en instalación y configuraci ón	Poderosa al ser combinad a con otra técnica	La protecció n a nivel de host se considera una tarea costosa y que consume mucho tiempo.	Similar a la técnica basada en host	Tiempo de procesam iento alto para datos a gran escala	Permite la implementa ción de varias técnicas de detección y versatilidad del software utilizado.	La tarea de detección se completa con el desarrollo de un proceso que analiza los tiempos de conexión, números de conexiones muertas y consultas a bases de datos de botnets descubiertos.	El agente monitorea el comportamie nto de los hosts con respecto a las sesiones de escaneo y salientes dentro de diferentes subredes para guiar la ubicación de los mecanismos de defensa.	El tráfico de red es capturado por un IDS y clasificado por el algoritmo, seleccionand o el juego de característica s reduce el error causado por decisiones basadas en la experiencia.	Se examina las característica s de flujo tales como IP, puerto, tiempos de eventos de paquetes y bytes por paquete para evidenciar la actividad de botnets, se reducen los falsos positivos.

Tabla 3.10 Análisis comparativo de técnicas detección botnets

4. SELECCIÓN DE LA HERRAMIENTA DE SOFTWARE LIBRE PARA EL ANÁLISIS Y DETECCIÓN DE BOTNETS

Las soluciones tecnológicas pueden ofrecer una amplia variedad de beneficios tanto para usuarios individuales, así como para empresas e instituciones educativas, donde la principal diferencia es la cantidad real de recursos financieros que están en juego si la tecnología es aprovechada por un atacante. Integrando servicios en un modelo de negocio, una organización se puede aumentar la accesibilidad de alcance del negocio. Sin embargo, al mismo tiempo hace esa organización vulnerable a las amenazas cibernéticas, que les pueden causar un daño financiero. Así, la presencia del firewall correctamente configurado y actualizado con frecuencia anti-malware aún no garantiza la protección de la computadora o una red de computadoras y para ello es importante controlar el tráfico y detectar posibles casos de comportamientos sospechosos. A lo largo de este capítulo se exponen y justifican las elecciones tomadas respecto a las herramientas existentes, entre las que se encuentran: Botminer, ourmon, Nsgbot, bothunter, IDS basado en Red.

4.1. Botminer

Es un prototipo desarrollado por investigadores del College of Computing, Georgia Institute of Technology. Su funcionamiento empieza con la premisa de que, en una botnet, los bots se comunican con su servidor y sus pares, y ejecutan actividades maliciosas, siguiendo patrones de comunicación y de actividad maliciosa. BotMiner es una extensión de BotSniffer, que se utiliza para detectar botnets del mundo real incluyendo Nugache basado en IRC, basado en P2P, y Storm gusano con una baja tasa de falsos positivos [49].

Estos investigadores probaron un marco basado en flujo de red mediante la extracción de múltiples archivos de registro para detectar actividades bot en las máquinas de los usuarios mediante la correlación temporal de dos archivos de registro de usuarios (tcpdump, exedump). Estos archivos de registro se utilizaron para registrar todo el tráfico de red entrante y saliente, y para mantener el historial del tiempo de inicio de cada ejecución de la aplicación en el nivel de usuario de la máquina. Este enfoque debe implementarse en sistemas en tiempo real para validar la efectividad. En la figura 4.1, se puede ver la arquitectura de Botminer.

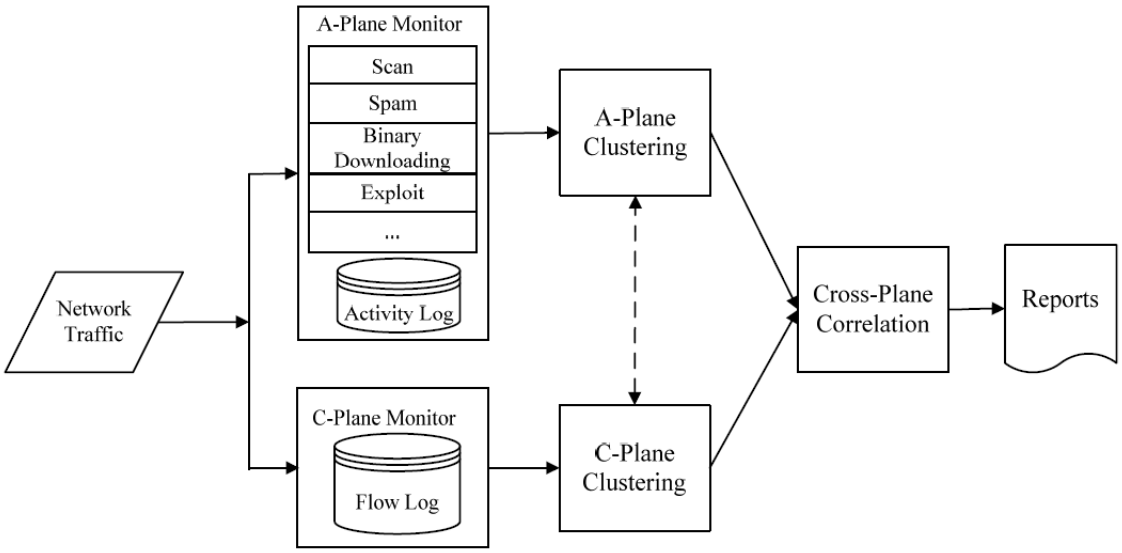


Figura 4.1 Arquitectura de Botminer **[49]**

4.2. Ourmon

Es una herramienta opensource que analiza el tráfico de red y detecta anomalías, orientada a la estadística. Las botnets se encuentran incluidas dentro de las anomalías que detecta. Ourmon se basa en la recopilación de paquetes en modo promiscuo en las interfaces Ethernet y generalmente utiliza la duplicación de puertos a través de un conmutador Ethernet. Un sensor recopila paquetes que se consideran importantes y envía tuplas definidas internamente a un sistema de visualización de gráficos que puede o no estar en el mismo host.

Ourmon analiza los datos utilizando las dos instancias múltiples del filtro de paquetes de Berkeley, y también varias listas hash y luego muestra los datos utilizando gráficos de RRDTOOL, histogramas e informes ASCII [50]. Los datos se producen casi en tiempo real cada treinta segundos. Los informes por hora también se producen para algunas de las listas y se resumen diariamente, dando aproximadamente una semana de informes de registro resumidos.

Ourmon incluye el control de flujo tradicional, de hecho, define sus propios formatos internos para "flujos", lo que permite que los flujos sean más eficientes y se centran solo en recopilar información de interés para la tupla de flujo en cuestión. Por ejemplo, ourmon incluye dos tuplas de "flujo" de IRC (aplicación de capa 7), una para canales de IRC y otra para hosts de IRC en un canal de IRC determinado. Existe una tupla de flujo para consultas de DNS, así

como otra tupla de flujo para la exploración de SSH [51]. Muchas tuplas de flujo se basan en la dirección de host IP y dan pistas importantes sobre lo que está haciendo un host. Estos son útiles para la detección de anomalías. Ourmon también proporciona información sobre ataques coordinados y varios tipos de gusanos. La arquitectura de Ourmon puede visualizarse en la figura 4.2

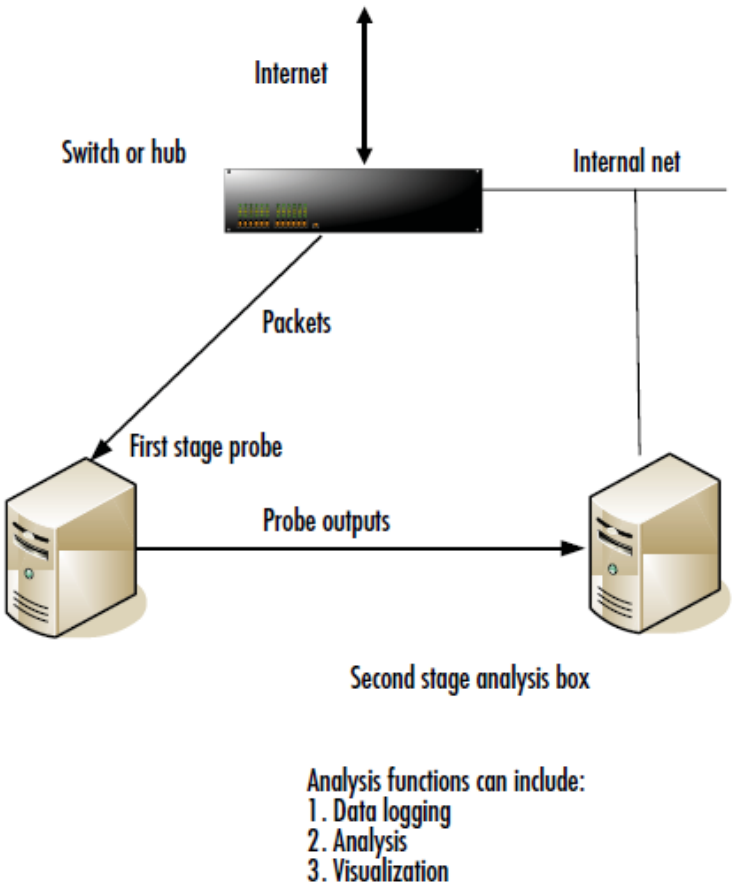


Figura 4.2 Arquitectura de Ourmon **[52]**

4.3. IDS basado en red

Los Network based IDS o IDS basados en red (NIDS), son un subgrupo dentro de los IDS que se caracterizan por analizar eventos del tráfico de la red en la que se instalan y clasifican adecuadamente dependiendo de su contenido [53]. Un NIDS realiza un examen exhaustivo de los paquetes y segmentos de la red que monitoriza, buscando coincidencias con firmas previamente definidas. Normalmente están formados por un conjunto de sensores, localizados estratégicamente en diferentes puntos de la red. Estos sensores se encuentran configurados para actuar de manera transparente a la comunicación, por lo que no requiere ningún tipo de modificación de la infraestructura a monitorizar.

La principal ventaja de este tipo de sistemas es que, gracias a los sensores, abarca toda la infraestructura de red y no dispositivos por separado. Esto permite detectar un mayor número de ataque y relacionar acciones entre sí. Esta transparencia también tiene implicaciones positivas en el ámbito de la seguridad, pues resulta difícil para un atacante detectar la presencia de un NIDS. Se muestra la arquitectura de un NIDS en la figura 4.3

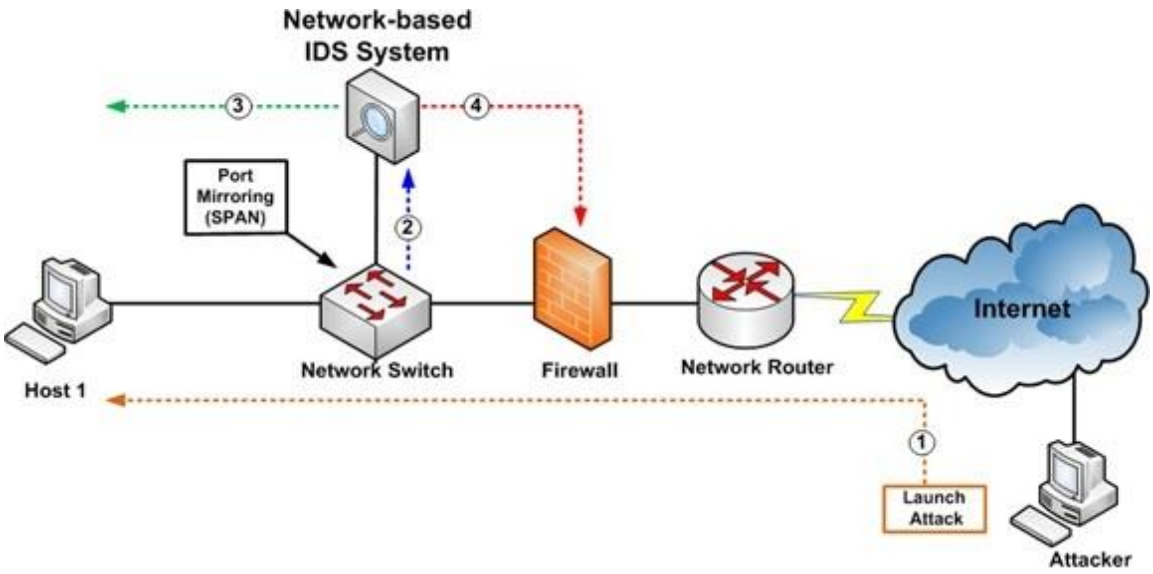


Figura 4.3 Arquitectura de un NIDS **[54]**

4.4. Selección de las herramientas de software libre

Se procederá a decidir la herramienta con la que se trabajará en nuestro caso de estudio. En base a la comparación realizada, se ha elegido utilizar un complemento de Snort con Stack ELK. Es la más completa de todas ya que se pueden permite la integración con el stack ELK, el cual permite una visualización detallada de los resultados.

Botminer, es una herramienta basada en la técnica de minería de datos. Por lo tanto, no será utilizada para el desarrollo de este proyecto. En la sección 3.5 se visualiza el análisis comparativo de técnicas de detección de vulnerabilidades tipo botnet.

Ourmon, es una herramienta antigua, su última actualización se dio el 10 de abril del 2013. Tiene poca documentación y no tiene soporte técnico de parte de sus desarrolladores. Por lo tanto, no será utilizada.

IDS basado en red, es una combinación de un sistema de detección de intrusos con un administrador/analizador de registros o logs de los eventos generados en múltiples servidores. La característica principal es el manejo de un gran número de eventos sin tener que preocuparse por su escalabilidad. Por lo tanto, será utilizada en nuestro proyecto.

En nuestro caso de estudio se va a usar de un IDS (Snort) integrado a un administrador de logs (Stack ELK), los cuales se detallan en el siguiente capítulo. La comparación y elección de la herramienta de software se muestra en la tabla 4.1

	Botminer	**Ourmon**	**Snort con Stack ELK**
Desarrollador	Texas A&M University	Jim Binkley, Portland State University	Cisco Systems Elastic
Disponibilidad	Desde 1998	Desde 2006	Desde 2009
Sistema Operativo	UNIX/Linux Open Source	UNIX/Linux Open Source	Cross-platform
Versión estable	31 de julio 2008	28 agosto 2006	2018
Resumen	El objetivo de BotMiner es detectar grupos de máquinas comprometidas dentro de una red monitoreada que forman parte de una red de bots. Se hace analizando pasivamente el tráfico de red en la red monitoreada. Genera un numero bajo de falsos negativos. Es independiente de los protocolos y estructura de comunicación con el	La idea básica es que ourmon detecta anomalías de red basadas en hosts que atacan a otros hosts mediante ataques de denegación de servicio (DoS) o mediante escaneo de red. Luego puede correlacionar esta información con los canales de IRC y decirle si un canal de IRC completo (conjunto de hosts de comunicación) es sospechoso.	El ELK Stack es una colección de tres productos de código abierto: Elasticsearch, Logstash y Kibana, todos desarrollados, administrados y mantenidos por Elastic. Elasticsearch es una base de datos NoSQL es una herramienta de canalización de registros que acepta entradas de varias fuentes, ejecuta diferentes transformaciones y

	botmaster.	Por lo tanto, es posible encontrar un conjunto completo de hosts infectados a la vez.	exporta los datos a varios destinos. Kibana es una capa de visualización que funciona sobre Elasticsearch.
Fuente de información que monitorea	Red	Red	Red
Dificultad de instalación y configuración	Alta	Alta	Alta
Detección en tiempo real	No	Si	Si
Detección automática de protocolos	Si	Si	Si
GeoIP	SI	Si	Si
Análisis Avanzado de HTTP	No	Si	Si
Criterio de selección	Mayor robustez, basado en correlación horizontal, evaluación de resultados. Software en desuso, falta de seguimiento de parte de sus desarrolladores. Documentación no disponible.	Basado en anomalías, generación de reportes TCP, UDP y mail. Eficiente para ataques DDoS. Software en desuso, falta de seguimiento y respaldo de parte de sus desarrolladores. Sin documentación.	Personalización de reglas, integración de elementos de hardware y software. Generación de reportes, soporte técnico por parte de los desarrolladores. Documentación disponible. Actualizaciones periódicas.

Tabla 4.1 Selección de herramientas de software

4.4.1. Sensores IDS y Stack ELK

En primer lugar, en todo sistema de detección de intrusos basado en red se tiene sensores. Los sensores son los responsables de la captura del tráfico de red en bruto. Estos sensores contienen el motor de correlación que, en base a unas reglas predefinidas y mediante una comparación optimizada, emite alertas indicando posibles comunicaciones sospechosas. Estas reglas han sido creadas en base a muestras de tráfico reportadas previamente, extrayendo sus cualidades comunes como: direcciones IP, dominios, puertos específicos, contenido dentro del mensaje y/o un conjunto de las anteriores. Estas reglas se obtienen de una actualización constante e incremental, lo cual significa que solo será necesario actualizar a la última firma existente.

En la red de datos de una organización se generan miles o millones de eventos diariamente, los cuales pueden y deben ser analizados para la identificación de posibles vulnerabilidades. Los eventos son almacenados en una base de datos conectada a un visualizador grafico web. Existen una gran variedad de datos que pueden ser representados como orígenes, destinos, fechas, horas, acciones o el contenido de la comunicación.

La última fase del flujo de datos corresponde a la interacción del analista con el sistema de detección de intrusos. La interfaz gráfica web es la parte del sistema informático que actúa como punto de comunicación entre este y el usuario, utilizando un conjunto de objetos gráficos para representar la información y acciones disponibles. La capacidad de un analista para relacionar un grupo de alertas, trazar una línea temporal y, por último, emitir un diagnóstico y solución dependerá de las opciones que dicha interfaz ofrezca. Por consiguiente, entre las necesidades de esta se encontrará la aplicación de múltiples filtros como: IP origen, IP destino, puerto origen, puerto destino, fecha, hora, tipo de alerta, protocolo y campos dentro de la comunicación (payload). En la tabla 4.2 se detalla los elementos que se va a analizar:

Sensor:		
Snort	Suricata	Bro
Base de datos:		
Elasticsearch		
Recolección de datos:		
Logstash		
Interfaz gráfica:		
Kibana		

Tabla 4.2 Sensores IDS y Stack ELK

4.4.2. IDS – Sistemas de detección de intrusos

Entre el amplio elenco de soluciones de detección de intrusos basados en red, cabe destacar tres productos sobre el resto: Snort, Suricata y Bro.

Snort ha sido el motor de IDS de facto durante años. Tiene una enorme comunidad de usuarios soportándolo y, un abanico aún mayor y en aumento de suscriptores a sus reglas. Por otro lado, aunque Suricata no tiene una vida tan larga en comparación con Snort, ha estado reclamando mercado como una respuesta evolucionada o alternativa a este, basando sus argumentos principalmente en sus capacidades de proceso multi-hilo. Bro busca amenazas específicas y alertas, almacenando los metadatos de red más eficientes de captura de paquetes, indexando y divulgando formas antes no registradas. La herramienta ha ganado mercado debido a sus capacidades y flexibilidad única.

4.4.2.1. Snort

Hay infinidad de opciones a la hora de elegir motores IDS disponibles para automatizar y simplificar el proceso de detección de intrusiones, y Snort es una de las mejores opciones. Snort se ha convertido en la tecnología de prevención y detección más ampliamente extendida y confiable del mundo [55]. El éxito de Snort se debe al hecho de que los usuarios de la comunidad de seguridad de código abierto de todo el mundo pueden detectar y responder ante incidentes de seguridad de forma más rápida y eficiente que con otros motores IDS. Usa un lenguaje de reglas flexible para detectar tráfico en tiempo real, así como un motor de detección basada en arquitectura modular.

Snort Está disponible bajo licencia GPL, gratuito y funciona bajo plataformas Windows y UNIX/Linux. Es uno de los más usados y dispone de una gran cantidad de filtros o patrones ya predefinidos, así como actualizaciones constantes ante casos de ataques, barridos o vulnerabilidades que vayan siendo detectadas a través de los distintos boletines de seguridad [56]. Además, hay una amplia variedad de guías de referencia disponibles para instalar, configurar, implementar y administrar sensores de Snort. Con casi 4 millones de descargas y cerca de 400.000 usuarios registrados [57], Snort se ha convertido en la tecnología de detección de intrusos más utilizada del mundo.

El principal trabajo de Snort es escuchar el tráfico de red y buscar firmas en el flujo de datos que puedan indicar un incidente de seguridad en la infraestructura a monitorizar. Las reglas están configuradas para realizar una acción que puede ser pasiva, alertando del incidente, o activa, aplicando contramedidas en tiempo real. Las organizaciones pueden aprovechar la aplicación de conjuntos de reglas existentes proporcionadas por la comunidad de Snort, así como escribir o modificar sus propias reglas de acuerdo con requisitos específicos de la red. Es posible escribir reglas complejas para identificar casi cualquier tipo de tráfico que atraviesa la red. Gracias a la comunidad, las reglas de IDS se encuentran bajo una continua revisión y mejora con el fin de detectar las últimas amenazas de seguridad conocidas.

4.4.2.2. Suricata

Suricata es un sistema de detección de intrusos (al igual que Snort) basado en código abierto [58]. El objetivo de la comunidad OISF (Open Infomation Security Foundation) es aportar nuevas ideas de seguridad e innovaciones tecnológicas a la industria de detección de intrusiones de IDS de código abierto. Con la ayuda financiera del Departamento de

Seguridad Interna de los Estados Unidos, se creó una alternativa multiproceso a Snort para ayudar a proteger las redes contra intrusiones de seguridad avanzadas. La arquitectura de múltiples hilos de Suricata es única, ya que puede soportar sistemas multi-core y multiprocesador de alto rendimiento. Los principales beneficios de un diseño de múltiples hilos es que ofrece mayor velocidad y eficiencia en el análisis de tráfico de red y también puede ayudar a dividir la carga de trabajo de IDS/IPS en función de las necesidades de procesamiento. Además de la aceleración de hardware (con limitaciones de hardware y de tarjeta de red), el motor está diseñado para utilizar la mayor potencia de procesamiento ofrecida por los últimos chips de CPU multi-núcleo.

Suricata se ha desarrollado con la idea de una fácil implementación, acompañado de documentación de inicio paso a paso y un potente manual de usuario. El motor también ha sido desarrollado en C, pensado desde los inicios para escalar. Adicionalmente, para facilitar la migración a este producto, Suricata emplea las mismas reglas y formatos de salida que Snort. Aunque Suricata sigue siendo un producto joven y menos extendido que su competencia, esta tecnología está ganando impulso entre los usuarios y empresas. El aumento del rendimiento, la compatibilidad con IPv6 nativa, modelos estadísticos de detección de anomalías y aceleración GPU son, entre otras, los principales puntos a favor de este motor.

4.4.2.3. Bro

Bro es un marco de software de código abierto para analizar el tráfico de red que se utiliza más comúnmente para detectar anomalías de comportamiento en una red con fines de ciberseguridad. Es principalmente un monitor de seguridad que inspecciona todo el tráfico en un enlace en profundidad para detectar signos de actividad sospechosa [59]. Sin embargo, de manera más general, Bro admite una amplia gama de tareas de análisis de tráfico incluso fuera del dominio de seguridad, incluidas las mediciones de rendimiento y la resolución de problemas. Bro se ejecuta en hardware básico y, por lo tanto, ofrece una alternativa de bajo costo a soluciones propietarias costosas. Sin embargo, a pesar de la etiqueta de precio, tiene más capacidades en comparación con otras herramientas de monitoreo de red, que normalmente se limitan a un pequeño conjunto de tareas de análisis codificadas.

El beneficio más inmediato que obtiene un sitio al implementar Bro es un conjunto extenso de archivos de registro que registran la actividad de una red en términos de alto nivel. Estos

registros incluyen no solo un registro completo de cada conexión vista en el cable, sino también transcripciones de nivel de aplicación como, por ejemplo, todas las sesiones HTTP con sus URI, encabezados clave, tipos MIME y respuestas del servidor solicitados; solicitudes de DNS con respuestas; certificados SSL; contenido clave de las sesiones SMTP; y mucho más. De forma predeterminada, Bro escribe toda esta información en archivos de registro separados por tabuladores y bien estructurados, adecuados para el post procesamiento con software externo. Sin embargo, los usuarios también pueden elegir entre un conjunto de formatos de salida y backends alternativos para interactuar directamente con, por ejemplo, bases de datos externas.

Además de los registros, Bro viene con una funcionalidad incorporada para una variedad de tareas de análisis y detección, incluida la extracción de archivos de las sesiones HTTP, la detección de malware mediante la interfaz con registros externos, el reporte de versiones vulnerables de software que se ven en la red, la identificación de sitios web populares, aplicaciones, detección de forzados brutos SSH, validación de cadenas de certificados SSL y mucho más.

Sin embargo, la clave para entender a Bro reside en darse cuenta de que, aunque el sistema viene con una funcionalidad potente, fundamentalmente representa una plataforma para análisis de tráfico que es completamente personalizable y extensible. Bro proporciona a los usuarios un dominio específico, Turing-Lenguaje de scripting completo para expresar tareas de análisis. Conceptualmente, se puede pensar en Bro como un "Python específico del dominio" (o Perl): al igual que Python, el sistema viene con un gran conjunto de funciones pre construidas, pero no está limitado a lo que el sistema incluye, pero puede hacer que Bro se utilice de manera novedosa al escribir su propio código. De hecho, todos los análisis predeterminados de Bro, incluido todo el registro, es el resultado de dichos scripts; no hay un análisis específico codificado en el núcleo del sistema.

Una buena manera de entender es compararlo con un IDS convencional basado en reglas, como Suricata o Snort. Un caso de uso clásico para esas herramientas es monitorear el tráfico en un puerto específico para un atributo específico, un cierto protocolo o patrón de bytes que existe en las cargas útiles de paquetes [60]. Cuando esas condiciones coinciden con la regla, el IDS dispara una alerta.

4.4.3. Análisis comparativo y selección del sensor IDS

Se procede a detallar las características principales para así hacer uso de uno de los sensores con el cual se trabajará más adelante. Cabe mencionar que el sensor a utilizar puede ser cualquiera que se crea conveniente y no se debe sentirse obligado al uso de la técnica que se ha obtenido. En base a la comparación realizada, se ha elegido utilizar el sensor Snort para este proyecto. Es la más completa de todas ya que se puede integrar con el Stack ELK. Se detalla la elección en la tabla 4.3

	Snort	**Bro**	**Suricata**
Desarrollador	Sourcefire, Inc.	International Computer Science Institute (ICSI)	Open Information Security Foundation (OISF)
Disponibilidad	Desde 1998	Desde 2010	Desde 2009
Lenguaje de codificación	C	script Lua	C
Sistema Operativo	Cross-platform	UNIX/Linux Open Source	Cross-platform
Versión estable	2.9.12 (11 octubre 2018)	2.5.5 (28 agosto 2018)	4.1 (6 noviembre 2018)
Hilos	Hilo único	Hilo único	Multi-hilo
Soporte IPv6	Si	Si	Si
Fuente de información que monitorea	Red	Red	Red
Soporta reglas de Snort (VRT)	Si	Si	Si
Personalización de reglas	Si	No	Si
Soporta reglas de Emerging Threats	Si	Si	Si
Compatible con Aanval	Si	No	Si
IP Reputation	No	Algo	Si
Dificultad de instalación y	Alta	Alta	Baja

configuración			
Detección en tiempo real	Si	Si	Si
Detección automática de protocolos	No	Si	Si
Aceleración de GPU	No	No	Si
Variables Globales/Flowbits	No	Si	Si
GeoIP	No	Si	Si
Análisis Avanzado de HTTP	No	Si	Si
HTTP Access Logging	No	Si	Si
SMB Access Logging	No	Si	Si
Criterio de selección	Documentación disponible, gestión de falsas alarmas, personalización de reglas, detección ataques externos e internos, almacenamiento de intrusiones detectadas.	Documentación disponible, personalización de reglas, detección ataques externos e internos, almacenamiento de intrusiones detectadas.	Documentación disponible, personalización de reglas, detección ataques externos e internos, almacenamiento de intrusiones detectadas, análisis de integridad de un equipo.

Tabla 4.3 Selección de IDS

4.4.4. Stack ELK

Es un paquete de tres herramientas de software libre de la empresa Elastic. Las herramientas son Elasticsearch, Logstash y Kibana. Estas tres herramientas son proyectos independientes y pueden ser usadas por separado, pero juntas forman un gran equipo. Entre ellas se van a encargar de leer y almacenar toda la información que se va a necesitar

para posteriormente consultarla y monitorizarla. Elasticsearch es una base de datos NoSQL, sirve para análisis y búsqueda de texto completo altamente escalable. Le permite almacenar, buscar y analizar grandes volúmenes de datos de forma rápida y casi en tiempo real [61]. Logstash es un motor de recopilación de datos de código abierto con capacidades de canalización en tiempo real, puede unificar dinámicamente los datos de diferentes fuentes y normalizar los datos en los destinos de su elección y exporta los datos a varios objetivos [62]. Kibana es un visualizador web de Elastic, posibilita la exploración visual y análisis en tiempo real de los datos en Elasticsearch. El stack también incluye una familia de transportistas de registro llamado Beats. Beats es una familia de agentes muy ligeros que recolectan datos en hosts o servidores y los envían a Elasticsearch. En nuestro caso de estudio se utilizará filebeats.

4.4.4.1. Arquitectura Stack ELK

Los distintos componentes ELK fueron diseñados para interactuar sin demasiada configuración. Ver la figura 4.4

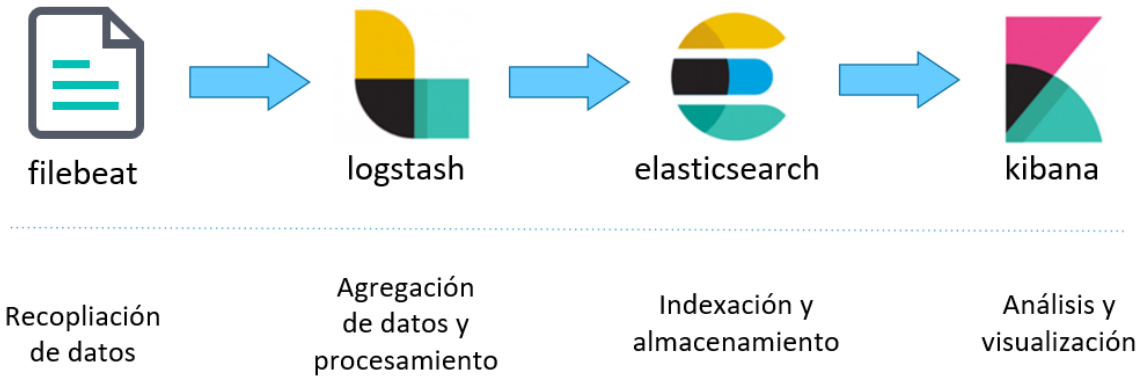


Figura 4.4 Arquitectura Stack ELK

4.4.4.2. Base de datos

Con respecto a la persistencia de datos y consultas parece no existe una mejor opción que: ElasticSearch.

4.4.4.2.1. ElasticSearch

ElasticSearch es una solución de base de datos. Este producto provee un motor de búsqueda distribuido y con capacidad multi-tenencia. El término multi-tenencia se refiere a una arquitectura de software en la que una única instancia de software se ejecuta en un servidor y sirve a varios inquilinos. Los inquilinos son un grupo de usuarios que comparten

un acceso común con privilegios específicos a la instancia de software. Con una arquitectura multi-tenencia, una aplicación de software está diseñada para proporcionar a cada inquilino una parte dedicada de la instancia, incluyendo sus datos, configuración, gestión de usuarios, funcionalidad individual y propiedades no funcionales [63]. multi-tenencia contrasta con arquitecturas multi-instancia, donde las instancias de software por separado operan en nombre de los diferentes inquilinos.

4.4.4.3. Recolección de alertas

Se analiza las diferentes opciones para enviar las alertas generadas por Snort desde nuestro sensor a la base de datos de Elasticsearch.

4.4.4.3.1. Logstash

Logstash es un motor de recolección de datos de código abierto desarrollado por Elastic. Este puede unificar dinámicamente datos de fuentes dispares y normalizar los datos en destinos de su elección [62]. Si bien Logstash originalmente impulsó la innovación en la recopilación de registros, sus capacidades se extienden mucho más allá de ese caso de uso. Cualquier tipo de evento puede ser enriquecido y transformado con una amplia gama de entradas, filtros y plugins de salida, con muchos códecs nativos simplificando aún más el proceso de la recolección. Esto permite modificar los registros para almacenar toda la información necesaria, en el formato que más nos interese y de la forma más óptima.

La aplicación se encuentra basada en jRuby y requiere de una “Java virtual Machine” para funcionar. Gracias a esto la herramienta es multiplataforma, pudiendo ser ejecutada en cualquier sistema operativo moderno como Linux, Mac OS X o Windows. Otra ventaja de Logstash respecto a sistemas basados en syslog es que ha sido creada por el mismo desarrollador que la base de datos. Esto asegura que no habrá incompatibilidad entre las diferentes piezas de la implementación.

4.4.4.4. Interfaz gráfica

Por último, queda por analizar la capa más externa de la arquitectura, la interfaz gráfica la cual estará en contacto con el usuario. La elección de esta influirá en gran medida en la agilidad y capacidad de interpretar los eventos de seguridad que se han detectado en fases anteriores. Una mayor claridad en la exposición de los datos y posibilidad a la hora de aplicar filtros jugarán a favor de la eficiencia a la hora de investigar incidentes de seguridad.

4.4.4.4.1. Kibana

Kibana es una potente herramienta pensada para el análisis masivo de datos almacenados en una base de datos Elasticsearch [64]. Ofrece una infinidad de nuevas posibles representaciones gráficas para los eventos, como mapas de calor, localizaciones, etc. Ofrece un servicio accesible vía navegador. El software servidor no tiene dependencias del sistema operativo en el que se encuentra dado que, al igual que el resto de software de elastic, se monta sobre la Máquina Virtual Java (JVM).

Otra de las principales ventajas de Kibana es su completa integración con el resto de productos elegidos. Al ser un stack pensado y recomendado por el propio fabricante, no existirán incompatibilidades entre versiones al actualizar el software. Kibana es utilizando por importantes organizaciones para el manejo de un gran volumen de datos, algunos ejemplos son: Linkedin, Stackoverflow, Netflix, HipChat. En la figura 4.5, se presentan una captura de la interfaz gráfica web Kibana.

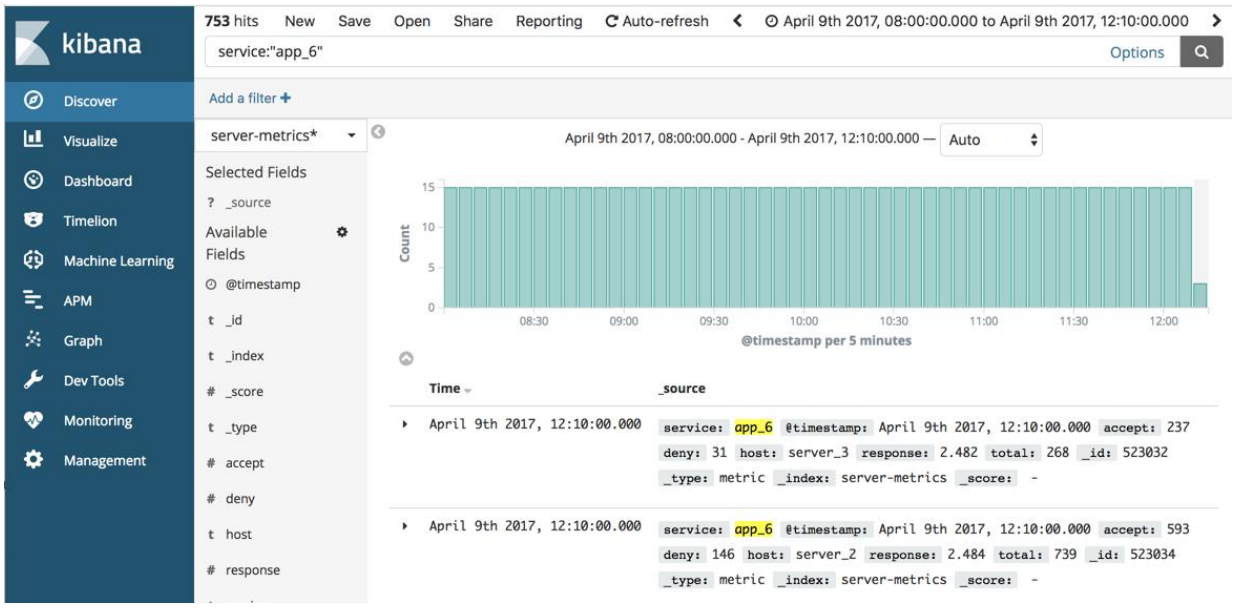


Figura 4.5 Visualización en Kibana

5. CASO DE ESTUDIO

Durante los cuatro capítulos anteriores, se ha abarcado todo lo concerniente al tema propuesto, con la cual se pretende realizar una efectiva detección de vulnerabilidades tipo botnet mediante la técnica de análisis de datos aplicado con las herramientas NIDS de Software libre; utilizando Snort con Stack ELK. En esta sección se comienza por describir los elementos de software de la herramienta seleccionada, seguidamente se hace una descripción de problema e identificación/valoración de los activos de red de la EPN. La instalación de las herramientas de software para la detección de vulnerabilidades tipo botnet, se detallarán en los siguientes capítulos, al igual que los resultados.

5.1. Presentación del caso de estudio

Inicialmente se quiere detallar los activos y la arquitectura utilizados en la *Red 4* de servidores de red de la EPN. Se ha elegido esta red debido a la sospecha de vulnerabilidades tipo botnet, muchos de los equipos infectados han sido difíciles de detectar debido a que son dispositivos portátiles y cambian su dirección IP cada vez que esta ha sido bloqueada por los administradores de la red. Cabe destacar que la detección de equipos portátiles no es parte de esta investigación, pero puede ser un tema a desarrollar en un futuro. En la figura 5.1 se muestra la arquitectura de la Red 4 de la EPN con conexión al servidor compuesto de Snort y Stack ELK. En la figura 5.2 se muestra la organización interna de nuestro servidor CentOS 7.

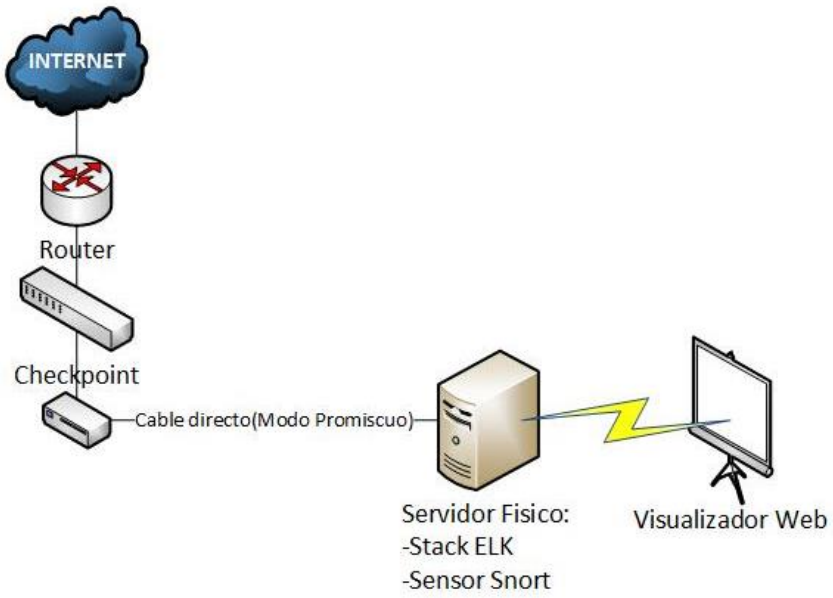


Figura 5.1 Arquitectura RED4 EPN

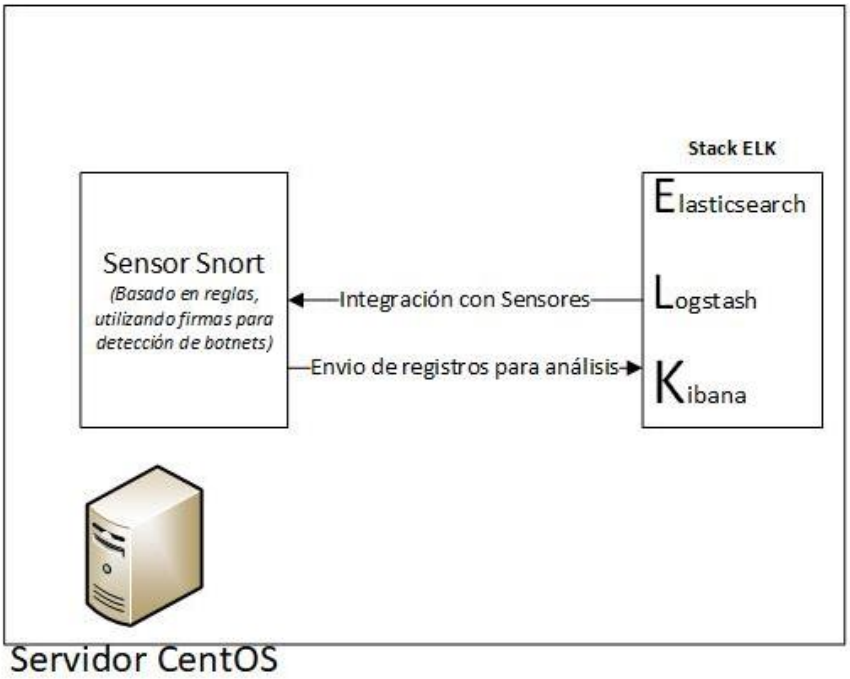


Figura 5.2 Detalle del servidor físico CentOS

El número de sensores de este tipo y su localización pueden variar dependiendo de las necesidades de la organización. Esto permite obtener infinidad de escenarios de monitorización, pero en nuestro proyecto se utiliza un solo sensor. En la red de datos de la EPN es posible capturar alrededor de 16 millones de eventos diariamente, es un número muy grande para ser analizado, por esa razón este proyecto se enfoca solamente en la Red 4, lo cual, proporcionará un número significativamente menor de eventos.

El sensor va a generar las alertas de detección de tráfico malicioso y serán procesadas por un usuario. Las aletas pueden ser generadas a cualquier hora, y a una velocidad vertiginosa, desbordando las capacidades humanas de los operarios. Por esta razón se ve necesario almacenarlas en una base de datos. Además del gran volumen de información, existe en una gran variedad de datos que pueden ser representados como orígenes, destinos, fechas, horas, acciones o el contenido de la comunicación.

5.2. Selección de la Metodología de Evaluación de Riesgos

Para la selección de la metodología de evaluación de riesgos, este proyecto está basado en el estudio presentado en [65]. De esta revisión de metodologías, se ha escogido a Octave y NIST gracias a la documentación disponible que presentan. En esta sección se realiza un análisis comparativo de estas dos metodologías. Se inicia con la descripción de cada una, con el objetivo de determinar la metodología más idónea.

5.2.1. OCTAVE

Octave Allegro es el método más recientemente desarrollado y apoyado activamente por el CERT (Community Emergency Response Team). Se centra en los activos de información. Los activos importantes de una organización se identifican y evalúan en base a los activos de información a los que se encuentran conectados. Se compone de 4 fases:

Fase 1: Desarrollar criterios de medición del riesgo en consonancia con la misión de la organización, las metas y objetivos del negocio, y los factores críticos de éxito.
Establecer los criterios para la medición de riesgos: establece los criterios de medición que se utilizarán para evaluar los efectos de un riesgo para la misión y objetivos de negocio de una empresa.

Fase 2: Crear un perfil de cada uno de los activos críticos de información, en donde se establecen claramente los límites para el activo, identifica sus necesidades de seguridad, e identifica todos sus contenedores (es decir en donde la información es almacenada, procesada y transportada).
Desarrollar un perfil de un activo de información: un perfil es una representación de un activo de información que describe sus características únicas, cualidades y valor. Es importante que los perfiles sean claros y consistentes, que no haya una descripción inequívoca de los límites del activo, y que los requisitos de seguridad del mismo estén definidos de forma adecuada.
Identificar los contenedores de los activos de información: Los contenedores son los lugares en donde los activos de información son almacenados, transportados y procesados, sean estos internos o externos. Es importante identificar los contenedores debido a que los riesgos asociados a los mismos son heredados por los activos de información.

Fase 3: Identificar las amenazas de cada activo de información en el contexto de cada uno de sus contenedores.

Identificar las áreas de preocupación: comienza con una lluvia de ideas sobre las posibles condiciones o situaciones que pueden poner en peligro la información de activos de información de una organización. Estos escenarios se los conoce como áreas de preocupación, y pueden representar amenazas y sus correspondientes resultados indeseables.

Identificar situaciones de amenaza: las áreas de preocupación identificadas en el paso anterior se expanden en escenarios de amenaza, que tienen en mayor detalle las propiedades. En este paso también se considera la probabilidad de ocurrencia de un escenario.

Fase 4: Identificar y analizar los riesgos para los activos de información y empezar a desarrollar los enfoques de mitigación.

Identificar riesgos: en base a las amenazas identificadas en el paso anterior, se identifican las consecuencias que tendría para una organización si una amenaza se presenta. Una amenaza puede tener múltiples impactos en la organización.

Analizar riesgos: se calcula el grado en que una amenaza afecta a la organización. Esta puntuación de riesgo relativo se obtiene teniendo en cuenta el grado del impacto del riesgo de la organización respecto de la importancia relativa de las distintas áreas de impacto, y, posiblemente, la probabilidad. Dependiendo de la organización, se dará prioridad a uno u otro criterio.

Seleccionar un enfoque de mitigación: las organizaciones determinan cuáles de los riesgos que se han identificado requieren de mitigación, y en base a esto se desarrolla una estrategia. Éstos son priorizados de acuerdo al riesgo relativo calculado en el paso anterior.

5.2.2. NIST SP 800-30

La metodología NIST SP 800-30 provee una guía para la evaluación de riesgos de sistemas de información federales y de organizaciones. De esta metodología existen 2 publicaciones, la que se encuentra vigente es la revisión 1 del 2012. De acuerdo a la publicación del 2012 de esta metodología, el proceso de evaluación de riesgos está conformado por cuatros fases que son:

Fase 1: Preparación para la evaluación de riesgos: este paso tiene como objetivo establecer el marco en el que se va a realizar la evaluación, es decir se debe definir el propósito, alcance, los supuestos y restricciones asociadas con la evaluación, las fuentes de información, los modelos de riesgos y los enfoques analíticos.

Fase 2: Realización de la evaluación de riesgos: que tiene como objetivo elaborar una lista de riesgos de seguridad priorizados por el nivel de riesgo. Para lo cual, en las organizaciones, se analiza las amenazas y vulnerabilidades, el impacto y la probabilidad de cada una, y la incertidumbre asociada con el proceso de evaluación de riesgos.

Fase 3: Comunicar y compartir los resultados de la evaluación de riesgos a toda la organización: el objetivo de este paso es asegurarse de que las personas que toman las decisiones dentro de la empresa cuenten con la información necesaria sobre los riesgos para orientarse en la toma de decisiones relacionadas con los riesgos.

Fase 3: Mantenimiento de la evaluación de riesgos: tiene como objetivo mantener actual el conocimiento sobre riesgos de las organizaciones, de manera que se pueda apoyar con esto el monitoreo de la efectividad de las respuestas al riesgo, que se han implementado. De acuerdo al cuadro comparativo a continuación, se ha escogido a Octave como la metodología para la evaluación de riesgos.

	OCTAVE	**NIST SP 800-30**	**MAGERIT V3.0**
Resumen	Se centra en el estudio de riesgos organizacionales, La evaluación inicia a partir de la identificación de los activos relacionados con la información. OCTAVE estudia la infraestructura de información. Se considera que, los empleados de todos los niveles necesitan entender qué activos relacionados con la información son importantes y cómo deben protegerlos.	Se centra en el aseguramiento de los sistemas de Información que almacenan, procesan y transmiten información. Realizando una adecuada gestión de Riesgos lo cual permitirá optimizar la administración de a partir del análisis de riesgos. Además de ayudar a la organización con la consecución de objetivos.	Considera la concientización de los usuarios de los sistemas de información de la existencia de riesgos y de la necesidad de detenerlos a tiempo. Ofrece un método sistemático para analizar riesgos, ayudando a descubrir y planificar las medidas oportunas para y a su vez preparar a la organización para procesos de evaluación, auditoría, certificación o acreditación.
Desarrollado por:	Universidad Carnegie Mellon	National Institute of Standard Technology	Ministerio de Hacienda y Administraciones Públicas

Idiomas disponibles y Precio	Se encuentra disponible en inglés. Es Gratuito	Se encuentra disponible en inglés. Disponible gratuitamente	Disponible en español, inglés e italiano (parcialmente) Disponible gratuitamente
Pasos	1. Establecer Criterios de Medición del Riesgo. 2. Desarrollar un Perfil para el Activo de Información 3. Identificar los Contenedores del Activo de la Información 4. Identificar Áreas de Interés. 5. Identificar Escenarios de Amenazas. 6. Identificar Riesgos 7. Analizar Riesgos 8. Seleccionar un Enfoque de Mitigación	1. Preparación para la evaluación del riesgo: 2. Conducción de la evaluación del riesgo: 3. Comunicar y compartir los resultados de la evaluación de riesgo: 4. Mantenimiento de la evaluación del riesgo:	1. Determinar activos 2. Amenazas (Determinación del impacto potencial, determinación del riesgo potencial) 3. Salvaguardas 4. Impacto residual 5. Estimación riesgo residual
Medición del riesgo	Establece los criterios de medición que se utilizarán para evaluar los efectos de un riesgo.	El proceso de medición del riesgo nos ayuda a determinar el alcance de la amenaza potencial y el riesgo asociado a un sistema.	Estimar el riesgo, definido como el impacto ponderado con la tasa de ocurrencia (o expectativa de materialización) de la amenaza
Identificación del riesgo	Se identifican las consecuencias que tendría para una organización si una amenaza se presenta	Implica identificar: propósito, alcance, supuestos y limitaciones, fuentes de información e insumos requeridos.	Determinar los activos, determinar amenazas en los activos y determinar qué salvaguardas hay dispuestas.
Análisis del riesgo	Se calcula el grado (probabilidad) en que una amenaza afecta a la organización.	Se debe analizar en conjunto con las vulnerabilidades y controles establecidos por el sistema de TI.	Se determina qué posee la organización a su disposición, y se trata de estimar la probabilidad qué

			posibles eventos podrían ocurrir.
Evaluación del riesgo	Se podría realizar una evaluación que se centrará en los activos más importantes para la organización,	Producir una lista de riesgos en la seguridad de la información que pueden ser priorizados por el nivel de riesgo y usados para informar decisiones en respuesta al riesgo.	Impacto y riesgo residual son una medida del estado presente, entre la inseguridad potencial (sin salvaguarda alguna) y las medidas adecuadas que reducen impacto y riesgo a valores aceptables.
Tratamiento del riesgo	Se clasifica cada uno de los riesgos según su puntaje para facilitar la posterior toma de decisiones en la cual se asigna un enfoque de mitigación para cada uno de los riesgos detectados mediante una estrategia elaborada.	Comunicar los resultados de la evaluación del riesgo al personal de toma de decisiones para saber dar respuesta, así como compartir información relacionada con el riesgo que ha sido generada durante el proceso de la evaluación.	Establece un plan de defensa contra incidentes que puedan poner en peligro, y a su vez se prepara para contrarrestar y continuar operando. El riesgo es reducido a niveles residuales que son asumidos por la organización.
Criterio de selección	Es uno de los métodos de Análisis y Gestión de Riesgos de los Sistemas de Información más popular y más utilizado. Presenta unas guías muy completas en comparación con Magerit y NIST SP 800-30.	Se siguen pasos de los análisis generales de riesgos: identificación de amenazas, identificación de vulnerabilidades y determinación de riesgos. Brinda una guía adecuada para la evaluación de riesgos.	Esta metodología ayuda a implementar un sistema de gestión y análisis de Riesgos de los sistemas de información de forma organizada ya que está estructurada en procesos bien definidos.

Tabla 5.1 Comparación de metodologías de evaluación de riesgos.

5.3. Identificación de activos de la Red 4

Los activos identificados en la Red 4 de la EPN, son aquellos que están detallados en la tabla 5.2

Componente/tipo de equipo (SWITCH, ROUTER, FIREWALL, SERVIDORES, etc)	MARCA/ MODELO/ FABRICANTE/ TIPO	DIRECCIÓN IP	NOMBRE DE MAQUINA	SISTEMA OPERATIVO/ VERSION DE SISTEMA OPERATIVO	Tipo de Log Soportado (Syslog/CEF/WMI, ASP, ETC)
Vlan 4 (Servidores producción: AD, DHCP, Srv_biblioteca, Srv_Antivirus, Srv_federacion, Quipux, Eduroam, Reloj Biométrico, Srv_Web_App_Moviles)	Ejemplo Srv_Antivirus: Kaspersky	172.31.4.5	antivirus.epn.edu.ec	Windows 2012 R2 Estándar	WMI
Balanceador de Carga / Servicio VPN	Citrix Netscaler	172.31.196.25	ns1.epn.edu.ec	NS11.1 57.11.nc	syslog
Firewall	Checkpoint R80.10	172.31.2.242	epncluster	Gaia Kernel Version: 2.6	opsec
Switches Distribución DGIP	WS-C4506-E	172.31.2.26	ddgip	c6880x-ipservicesk9-mz.SPA.151-2.SY3.bin	syslog, snmp
Controladora Wireless 1	AIR-WLC4404-100-K9	172.31.2.66	WLC-EPN1	7.0.240.0	syslog, snmp
Switch Core VSS netflow					
Gestor Identidades (ISE)	SNS-3415-K9	172.31.2.88	ise1-epn	2.0.1.130	syslog, snmp

Tabla 5.2 Activos de la Red 4

5.3.2. Valoración de activos

Una vez identificados los activos de la Red 4, se debe identificar que tan significativo es el aporte de cada uno de ellos, para esto, se ha analizado su valoración con respecto a las características de la seguridad informática.

Confidencialidad: se debe evaluar el impacto que tendría si el activo fuera accedido por personas, procesos o entidades no autorizadas. Para valorar la confidencialidad se debe considerar los criterios establecidos a continuación en la tabla 5.3:

Escala	Valor	Criterio
Muy Alto	5	El conocimiento o divulgación no autorizada de la información que gestiona el activo impacta negativamente a toda la organización.
Alto	4	El conocimiento o divulgación no autorizada de la información que gestiona el activo impacta negativamente de manera leve a la organización. Además, impacta al proceso evaluado y a los otros procesos de la organización.
Medio	3	El conocimiento o divulgación no autorizada de la información que gestiona el activo impacta negativamente al proceso evaluado.
Bajo	2	El conocimiento o divulgación no autorizada de la información que gestiona el activo impacta negativamente de manera leve al proceso evaluado.
Muy Bajo	1	El conocimiento o divulgación no autorizada de la información que gestiona el activo no impacta negativamente al proceso.

Tabla 5.3 Criterio de confidencialidad

Integridad: se debe evaluar el impacto que tendría si la precisión, calidad, veracidad, imparcialidad y completitud de la información fuera alterada. Para valorar la integridad se debe considerar los criterios establecidos a continuación en la tabla 5.4:

Escala	Valor	Criterio
Muy Alto	5	La pérdida de exactitud, precisión y completitud del activo impacta negativamente a la organización.
Alto	4	La pérdida de exactitud, precisión y completitud del activo impacta negativamente de manera leve a la organización. Además, impacta al proceso evaluado y a los otros procesos de la organización.
Medio	3	La pérdida de exactitud, precisión y completitud del activo impacta negativamente al proceso evaluado.
Bajo	2	La pérdida de exactitud, precisión y completitud del activo impacta negativamente de manera leve al proceso evaluado.
Muy Bajo	1	La pérdida de exactitud, precisión y completitud del activo no impacta negativamente al proceso.

Tabla 5.4 Criterio de integridad

Disponibilidad: se debe evaluar el impacto que tendría si los usuarios autorizados no tuvieran acceso a los activos de información en el momento que lo requieran. Para valorar la disponibilidad se debe considerar los criterios establecidos a continuación en la tabla 5.5:

Escala	Valor	Criterio
Muy Alto	5	La falta o no disponibilidad del activo impacta negativamente a la organización.
Alto	4	La falta o no disponibilidad del activo impacta negativamente de manera leve a la organización. Además, impacta al proceso evaluado y a los otros procesos de la organización.
Medio	3	La falta o no disponibilidad del activo impacta negativamente al proceso evaluado.
Bajo	2	La falta o no disponibilidad del activo impacta negativamente de manera leve al proceso evaluado.
Muy Bajo	1	La falta o no disponibilidad del activo no impacta negativamente al proceso.

Tabla 5.5 Criterio de disponibilidad

5.4. Evaluación de riesgos

Una vez definida la arquitectura del sistema de detección de intrusos basado en red en la sección anterior, se realizará la instalación de las herramientas necesarias para llevar a cabo este proyecto. Con estas herramientas se desea obtener un ambiente controlado del cual extraer valiosa información del funcionamiento del sistema más allá de la teoría. A lo largo de esta sección se describen los diferentes puntos de la instalación llevado a cabo.

Para empezar con la instalación en el servidor, es necesario una breve descripción de cada uno de los elementos, en la figura 5.3 se visualiza la arquitectura del Stack ELK. Se añade el sensor para la detección de tráfico e identificación de amenazas en la arquitectura mencionada en la descripción de las herramientas del Stack ELK. Es importante destacar que Filebeat no es necesario, ya que todos los elementos funcionan sobre el mismo equipo. Filebeat es útil cuando se tiene algún cliente desde donde se puede almacenar datos. Por lo tanto, los logs o registros de Snort entran directamente a Logstash.

5.4.1. Instalación de herramientas

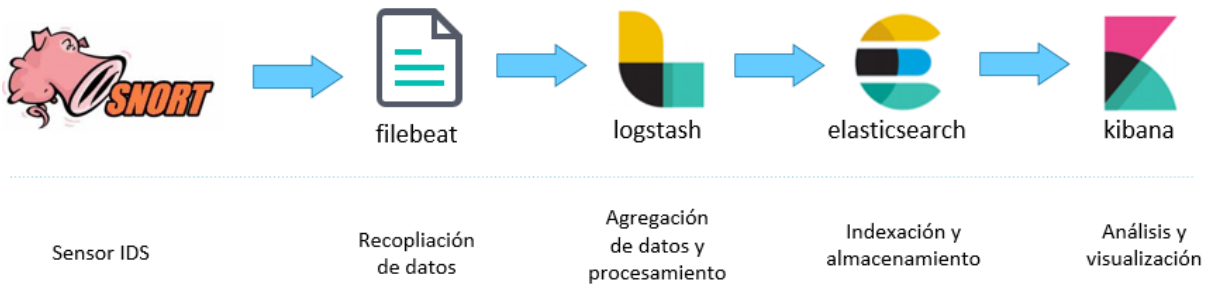


Figura 5.3 Implementación de Snort con Stack ELK

5.4.1.1. Instalación Snort

Snort es una opción popular para ejecutar un sistema de detección de intrusos de red o NIDS. Supervisa los datos del paquete enviados y recibidos a través de una interfaz de red específica. El NIDS puede detectar amenazas dirigidas a las vulnerabilidades de su sistema mediante la detección basada en firmas y las tecnologías de análisis de protocolo. El software NIDS, cuando se instala y configura adecuadamente, puede identificar los últimos ataques, infecciones de malware, sistemas comprometidos y violaciones de políticas de red [66].

Snort es uno de los IDS basados en red más utilizados. Es un código ligero, de código abierto, disponible en multitud de plataformas, y se puede instalar cómodamente incluso en las instancias más pequeñas de servidores en la nube. Aunque Snort es capaz de mucho más que el monitoreo de la red, se enseña cómo configurar y ejecutar Snort en modo NIDS con una configuración básica que luego puede expandir según sea necesario. Una configuración básica de Snort en CentOS es bastante simple, toma unos pocos pasos para completar. Primero deberá preparar su servidor en la nube para la instalación de Snort. Instale las bibliotecas requeridas con el siguiente comando.

```
# yum install gcc flex bison zlib libpcap pcre libdnet tcpdump
```

La última versión de Snort en este momento también requiere libnghttp2, que se puede descargar e instalar usando el comando que se encuentra debajo y también verificar que se pueda acceder a *libdnet* creando el siguiente enlace simbólico.

```
# yum install https://dl.fedoraproject.org/pub/epel/7/x86_64/Packages/l/libnghttp2-1.21.1-1.el7.x86_64.rpm
```

```
# ln -s /usr/lib64/libdnet.so.1.0.1 / a /lib64/libdnet.1
```

Ahora se confirma que se ha instalado correctamente

```
# java -version
```

Snort proporciona paquetes de rpm convenientes para CentOS 7, que se pueden instalar simplemente con los siguientes comandos. Snort utiliza algo llamado biblioteca de adquisición de datos (DAQ) para hacer llamadas abstractas a las bibliotecas de captura de paquetes.

```
# yum install https://www.snort.org/downloads/snort/daq-2.0.6-1.centos7.x86_64.rpm

# yum install https://www.snort.org/downloads/snort/snort-2.9.11.1-1.centos7.x86_64.rpm
```

A continuación, deberá configurar Snort para su sistema. Esto incluye editar algunos archivos de configuración, descargar las reglas que seguirá Snort y tomar Snort para una prueba de ejecución.

```
# ldconfig

# ln -s /usr/local/bin/snort /usr/sbin/snort
```

Crear la estructura de carpetas para albergar la configuración de Snort, copiar los comandos a continuación.

```
# mkdir -p /etc/snort/rules

# mkdir /var/log/snort

# mkdir /usr/local/lib/snort_dynamicrules
```

Establecer los permisos para los nuevos directorios en consecuencia.

```
# chmod -R 5775 /etc/snort

# sudo chmod -R 5775 /var/log/snort

# sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules

# sudo chown -R operador:operador /etc/snort

# sudo chown -R operador:operador /var/log/snort

# sudo chown -R operador:operador  /usr/local/lib/snort_dynamicrules
```

Cree nuevos archivos para las listas blancas y negras, así como las reglas locales.

```
# touch /etc/snort/rules/white_list.rules

# touch /etc/snort/rules/black_list.rules

# touch /etc/snort/rules/local.rules
```

Descargar las reglas de detección que Snort seguirá para identificar amenazas potenciales. Snort proporciona tres niveles de reglas, comunidad, registro y reglas de suscriptor. Extraer las reglas y cópielas en su carpeta de configuración

```
# wget https://www.snort.org/rules/community -O ~ / community.tar.gz

# tar -xvf ~/community.tar.gz -C ~/

# cp ~/community-rules/* /etc/snort/rules
```

Por defecto, Snort en CentOS espera encontrar una serie de archivos de reglas diferentes que no están incluidos en las reglas de la comunidad. Comente las líneas innecesarias usando el siguiente comando.

```
# sed -i 's/include \$RULE\_PATH/#include \$RULE\_PATH/' /etc/snort/rules/snort.conf
```

Con la configuración y los archivos de reglas en su lugar, edite el archivo *snort.conf* para modificar algunos parámetros. Abra el archivo de configuración para editarlo.

```
# vi /etc/snort/snort.conf
```

Encontrar estas secciones que se muestran a continuación en el archivo de configuración y cambie los parámetros.

```
# Setup the network addresses you are protecting

ipvar HOME_NET 10.254.13.100/24

-------------------------------------------------------

# Set up the external network addresses. Leave as "any" in most situations

ipvar EXTERNAL_NET !$HOME_NET

-------------------------------------------------------

# Path to your rules files (this can be a relative path)

var RULE_PATH /etc/snort/rules
```

```
var SO_RULE_PATH /etc/snort/so_rules

var PREPROC_RULE_PATH /etc/snort/preproc_rules

----------------------------------------------------------

# Set the absolute path appropriately

var WHITE_LIST_PATH /etc/snort/rules

var BLACK_LIST_PATH /etc/snort/rules
```

Desplazarse hacia abajo hasta la sección 6 y configurar la salida de *unified2* para registrar bajo el nombre de archivo de *snort.log* como se muestra a continuación.

```
# unified2

# Recommended for most installs

output unified2: filename snort.log, limit 128
```

Por último, desplazarse hacia la parte inferior del archivo para encontrar la lista de conjuntos de reglas incluidos. Deberá des comentar las reglas locales para permitir que Snort cargue las reglas personalizadas. Guardar los cambios y salir del editor.

```
# include $RULE_PATH/local.rules

# include $RULE_PATH/community.rules
```

Snort debería estar listo para correr. Pruebe la configuración usando el parámetro -T para habilitar el modo de prueba. Ver la figura 5.4

```
        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.12 GRE (Build 325)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2018 Cisco and/or its affiliates. All rights reser
ved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.5.3
           Using PCRE version: 8.32 2012-11-30
           Using ZLIB version: 1.2.7

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
```

Figura 5.4 Prueba de ejecución Snort

Si Snort está registrando las alertas como se espera, agregar una alerta de regla de detección personalizada en las conexiones ICMP entrantes al archivo *local.rules*. Abrir sus reglas locales en un editor de texto.

```
# vi /etc/snort/rules/local.rules

-- Agregar –

alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:10000001; rev:001;)
```

La regla consta de las siguientes partes:

- Acción para el tráfico que coincide con la regla, alerta en este caso.
- Protocolo de tráfico como TCP, UDP o ICMP como aquí
- La dirección de origen y el puerto, simplemente marcados como cualquiera para incluir todas Las direcciones y puertos
- La dirección y el puerto de destino, $ HOME_NET como se declara en la configuración y cualquier puerto
- Algunos bits adicionales
    - mensaje de registro
    - identificador de regla único (sid) que para las reglas locales debe ser 1000001 o superior
    - Número de versión de la regla.

Iniciar Snort con las opciones de una consola para imprimir las alertas a la salida estándar. Seleccionar la interfaz de red *ens224* correcta con la dirección IP

```
# snort -A consola -i ens224 -u snort -g snort -c /etc/snort/snort.conf

# ip addr
```

Snort registra las alertas en un registro en */var/log/snort/snort.log.<timestamp>*

```
# snort -r /var/log/snort/snort.log.
```

Snort en CentOS proporciona un script de inicio para ejecutar el servicio en segundo plano. Snort se puede ejecutar con la configuración que configuró usando el comando a continuación. Y también se ve el estado de nuestro sensor.

```
# systemctl start snortd

# systemctl status snortd
```

Snort tiene la capacidad de enviar una alerta de datos de tráfico en tiempo real. Se puede enviar una alerta de archivos separados o syslog, o ventanas emergentes. Snort está lógicamente dividido en varios componentes. Estos componentes, detectan ataques particulares y para generar la salida en formato requerido por el sistema de detección. Snort permite utilizar con la regla a los datos de tráfico de la red. Las reglas de Snort IDS tienen dos partes lógicas como la cabecera de la regla y la opción de regla como se muestra en la figura 5.5

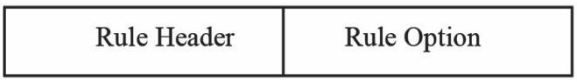


Figura 5.5 Estructura de reglas de Snort

Rule header: describe atributos de un paquete y Snort hace que el paquete que coincida con la regla como se muestra en la figura 5.6
Rule option: Seguirán a rule header y puede enviar un mensaje de alerta, sobre la parte del paquete que tiene la información.

En la figura 5.7 se detalla una regla de Snort. Los datos de tráfico de red del protocolo TCP por ejemplo dirección de origen están cualquier puerto origen 21, dirección de destino es 10.199.12.8, es puerto de destino es cualquiera, una generación que salidas el mensaje "Paquete TCP es detectado" con la firma id: 1000010.

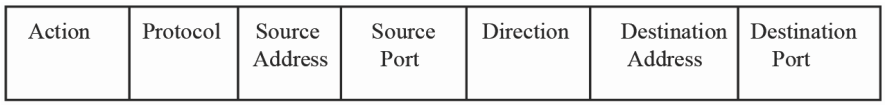


Figura 5.6 Estructura de Rule header

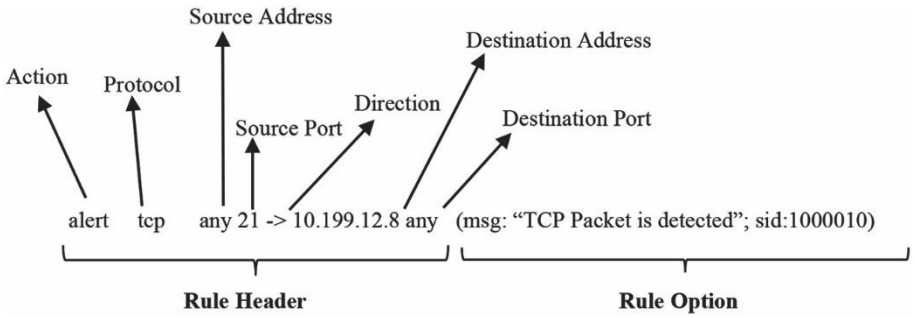


Figura 5.7 Regla de Snort

Cuando el Snort detecta paquetes de tráfico que coinciden con la regla de Snort, el sistema genera una alerta Es una parte importante del sistema ya que actúa como un medio para analizar y convertir los datos de tráfico de una red. Se puede analizar con las reglas

existentes en Snort o también con reglas personalizadas. A continuación, se muestra una regla personalizada, pueden incluirse otras.

	Reglas de botnet
1	Regla 1: alert tcp any -> any any (msg:"Possible NBSS Neris.exe SENT TO BROWSER"; flow:to server,established; content:"ACACACAC"; ttl:128; sid:2000341; rev:1) Regla 2: alert tcp any any -> any any (msg:"Neris IRC sent to HTTP"; flow:to server,established; content:"7& salt="; ttl:128; sid:2000366; rev:1)
2	Regla 1: alert tcp any any -> any any (msg:"TROJAN Possible IRCBot.DDOSCommon Commands"; flow:to server,established; content:"AAC6F603"; ttl:128; sid:1000255; rev:7) Regla 2: alert udp any any -> any any (msg:"IRC-bot sent spam BROWSER"; flow:to server,established; content:"FENEBEOC"; ttl:128; sid:1000199; rev:1)
3	Regla 1: alert tcp any any -> any any (msg:"Win32.Domsingx.A contact to C&C server attempt"; flow:to server,established; content:"ent-Type"; ttl:128; sid:4000309; rev:2) Regla 2: alert tcp any any -> any any (msg:"BLACKLIST DNS request for known malware/domain/sxzyong.com";flow:to server,established;content:"mail.com"; fast pattern:only;/metadata:impact flagred,service dns;reference:url,www.virustotal.com/filescan/report.html?id=a7f97ed5c064b038279dbd02554c7e555d97f67b601b94bfc556a50a41dae137-1304614426; classtype:trojan-activity; ttl:128; sid:3000230; rev:2)

Tabla 5.6 Reglas personalizadas de Snort

Algunos de los ejemplos de las reglas de Snort IDS se muestran en la tabla 5.6. El número 1 permite al sensor detectar un ataque en la conexión de red de Internet. El sensor le alerta cuando el atacante escanea el dispositivo en la web, que permite abrir cualquier puerto. En la número 2, se permite la conexión a internet cuando el atacante envía al sitio Web. Algo similar a un malware enviado por el atacante. Y esta regla también permite al sensor alertar cuando el atacante envía spam al navegador por los paquetes de protocolo UTP. En la número 3, se alerta por paquetes de protocolo TCP cuando el atacante enviar "Win32" al servidor C&C. Y también existe una "blacklist", cuando el atacante realiza una solicitud a un elemento de la blacklist.

5.4.1.2. Instalación Stack ELK

En instituciones educativas, empresa u organizaciones; existen un sin número de logs generados en los servidores y el monitoreo de forma manual de estos, es un trabajo complicado. En la red de la EPN se generan alrededor de 16 000 000 de paquetes de tráfico de red. Afortunadamente, la combinación de Elasticsearch, Logstash y Kibana en el lado del servidor, junto con Filebeat en el lado del cliente, hace que la tarea una vez difícil se vea como algo mucho más sencillo de analizar. Los primeros tres componentes forman lo que se llama el Stack ELK, cuyo propósito principal es recopilar registros de varios servidores al mismo tiempo (también conocido como registro centralizado). Una interfaz web basada en Java. Kibana incorporada le permite inspeccionar los registros rápidamente de un vistazo para facilitar la comparación y la resolución de problemas. Filebeat envía estos registros de clientes a un servidor central, que puede describirse como un agente de envío de registros [67] [68].

Para instalar Elasticsearch, Logstash y Kibana , se debe descargar las ultima versiones de estos tres componentes de la página oficial. *https://www.elastic.co/products*

5.4.1.2.1. Instalar Elasticsearch

Primero, se debe dirigir al directorio *Descargas*. A continuación, se procede a descargar e instalar la última versión de Elasticsearch [69].

```
# cd /home/operador/Descargas

# wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.5.2.rpm

# yum install elasticsearch-6.5.2.rpm
```

Cuando se complete la instalación, se le solicitará que inicie y habilite Elasticsearch.

```
# systemctl daemon-reload

# systemctl enable elasticsearch

# systemctl start elasticsearch
```

Permitir el tráfico a través del puerto TCP 9200 en su firewall.

```
# firewall-cmd --add-port=9200/tcp

# firewall-cmd --add-port=9200/tcp --permanent
```

Comprobar si Elasticsearch responde a solicitudes simples a través de HTTP. Se muestra la figura 5.8 a continuación.

```
# curl -X GET http://10.254.13.100:9200
```

```
[root@honeywall-tesis /]# curl -X GET http://10.254.13.100:9200
{
  "name" : "A80Ftr_",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "cXYd5L2pT2alCGU5jZTglg",
  "version" : {
    "number" : "6.5.2",
    "build_flavor" : "default",
    "build_type" : "rpm",
    "build_hash" : "9434bed",
    "build_date" : "2018-11-29T23:58:20.891072Z",
    "build_snapshot" : false,
    "lucene_version" : "7.5.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Figura 5.8 Elasticsearch responde a solicitudes HTTP

Se verifica que nuestro Elasticsearch se encuentre activo, a través de la terminal de comandos (figura 5.9) y del navegador web (figura 5.10).

```
# systemctl status elasticsearch
```

```
[root@honeywall-tesis /]# systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
   Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; enabled; vendor pres
et: disabled)
   Active: active (running) since mié 2018-12-05 23:47:59 -05; 1 day 14h ago
     Docs: http://www.elastic.co
 Main PID: 702 (java)
    Tasks: 142
   CGroup: /system.slice/elasticsearch.service
           ├─702 /bin/java -Xms1g -Xmx1g -XX:+UseConcMarkSweepGC -XX:CMSInitiatingO...
           └─892 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/b...

dic 05 23:47:59 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Started Elasticsearch.
dic 05 23:47:59 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Starting Elasticsearch...
Hint: Some lines were ellipsized, use -l to show in full.
```

Figura 5.9 Estado activo del servicio Elasticsearch (Terminal)

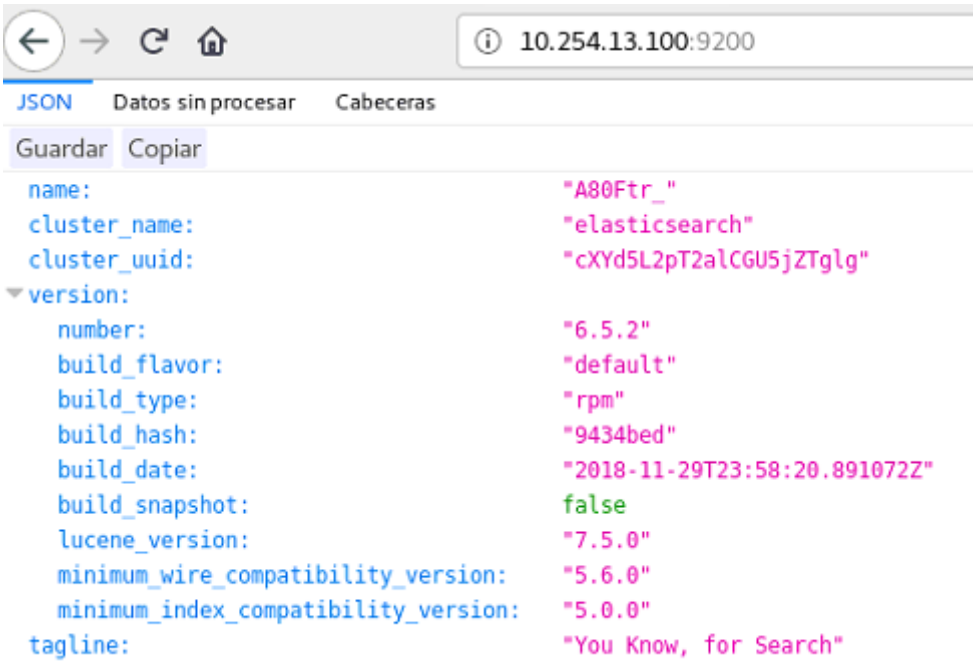


Figura 5.10 Estado activo del servicio Elasticsearch (Web)

5.4.1.2.2. Instalar Kibana

Primero, se debe dirigir al directorio *Descargas*. A continuación, se procede a descargar e instalar la última versión de Kibana [70].

```
# cd /home/operador/Descargas

# wget https://artifacts.elastic.co/downloads/kibana/kibana-6.5.2-x86_64.rpm

# yum install kibana-6.5.2-x86_64.rpm
```

Cuando se complete la instalación, se le solicitará que inicie y habilite Kibana.

```
# systemctl daemon-reload

# systemctl enable kibana

# systemctl start kibana
```

Asegúrese de que puede acceder a la interfaz web de Kibana desde otra computadora (permita el tráfico en el puerto TCP 5601)

```
# firewall-cmd --add-port=5601/tcp
```

```
# firewall-cmd --add-port=5601/tcp --permanent
```

Se verifica que nuestro Kibana se encuentre activo, ver la figura 5.11

```
# systemctl status kibana
```

```
[root@honeywall-tesis /]# systemctl status kibana
● kibana.service - Kibana
   Loaded: loaded (/etc/systemd/system/kibana.service; enabled; vendor preset: disable
d)
   Active: active (running) since jue 2018-12-06 00:24:12 -05; 1 day 13h ago
 Main PID: 20129 (node)
    Tasks: 20
   CGroup: /system.slice/kibana.service
           ├─ 3887 /usr/share/kibana/node/bin/node --no-warnings /usr/share/kibana/...
           └─20129 /usr/share/kibana/bin/../node/bin/node --no-warnings /usr/share/...

dic 06 00:24:12 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Started Kibana.
dic 06 00:24:12 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Starting Kibana...
dic 06 00:25:00 honeywall-tesis.csirt-epn.edu.ec kibana[20129]: [BABEL] Note: The c...
dic 06 01:23:52 honeywall-tesis.csirt-epn.edu.ec kibana[20129]: [BABEL] Note: The c...
dic 06 01:24:07 honeywall-tesis.csirt-epn.edu.ec kibana[20129]: [BABEL] Note: The c...
Hint: Some lines were ellipsized, use -l to show in full.
```

Figura 5.11 Estado activo del servicio Kibana

La configuración se encuentra en el fichero */etc/kibana/kibana.yml* en el cual se debe editar las siguientes variables para apuntar al servidor de Elasticsearch, que en este caso es el mismo en el que se hallará Kibana.

```
server.host: “10.254.13.100”

elasticseacrh.url: “http://10.254.13.100:9200”
```

Cada vez que se hace un cambio en algún archivo de configuración, es necesario reiniciar el servicio utilizando el siguiente comando y se comprueba el estado de Kibana con el comando mencionado unos pasos atrás.

```
# systemctl restart kibana
```

Se detallan el host y puerto donde escucha Kibana y el servidor de Elasticsearch donde realizará las consultas. El puerto indicado es el que se define por defecto. Ver la figura 5.12

```
# Specifies the address to which the Kibana server will bind. IP addresses and host n$
# The default is 'localhost', which usually means remote machines will not be able to$
# To allow connections from remote users, set this parameter to a non-loopback addres$
server.host: "10.254.13.100"
# The URL of the Elasticsearch instance to use for all your queries.
#elasticsearch.url: "http://localhost:9200"
elasticsearch.url: "http://10.254.13.100:9200"
```

Figura 5.12 Configuración Kibana

Una vez instalada la aplicación, es hora de cargar los “dashboards”, o presentaciones, que se encargaran de mostrar de una manera amigable los datos almacenados en la base de datos. Para ello, en primer lugar, se descarga un paquete prediseñado de Elastic:

```
# curl -L -O https://download.elastic.co/beats/dashboards/beats-dashboards-1.1.0.zip
```

A continuación, descomprimir el archivo y ejecutar el script para cargarlos que se encuentra en su interior.

```
# unzip beats-dashboards-*.zip

# cd beats-dashboards-*

# ./load.sh
```

Iniciar Kibana *http://10.254.13.100:5601* para verificar que puede acceder a la interfaz web, figura 5.13

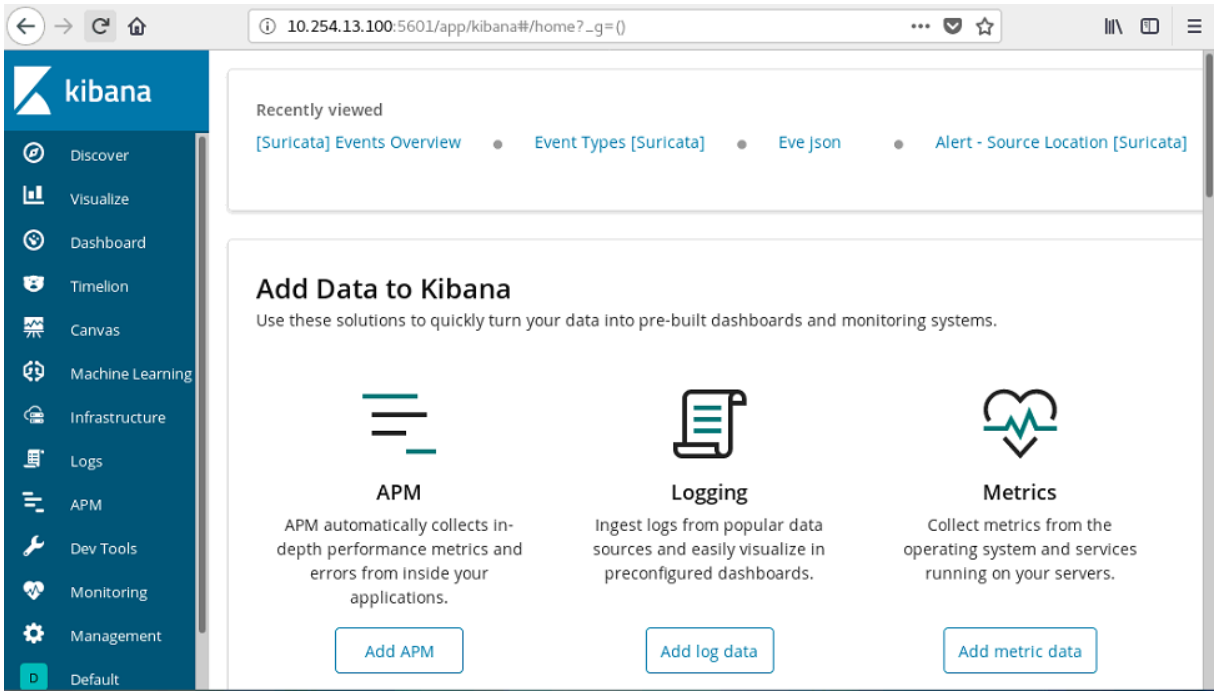


Figura 5.13 Interfaz gráfica web Kibana

5.4.1.2.3. Instalar Filebeat

Primero, hay que dirigirse al directorio *Descargas*. A continuación, se procede a descargar e instalar la última versión de Filebeat [71].

```
# cd /home/operador/Descargas

# wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.5.2-x86_64.rpm

# yum install filebeat-6.5.2-x86_64.rpm
```

Cuando se complete la instalación, se le solicitará que inicie y habilite Filebeat.

```
# systemctl daemon-reload

# systemctl enable filebeat

# systemctl start filebeat
```

Se verifica que nuestro Filebeat se encuentre activo. Ver la figura 5.14

```
# systemctl status filebeat
```

```
[root@honeywall-tesis Descargas]# systemctl status filebeat
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch
.
   Loaded: loaded (/usr/lib/systemd/system/filebeat.service; enabled; vendor preset: d
isabled)
   Active: active (running) since jue 2018-12-06 01:17:57 -05; 1 day 14h ago
     Docs: https://www.elastic.co/products/beats/filebeat
 Main PID: 29364 (filebeat)
    Tasks: 24
   CGroup: /system.slice/filebeat.service
           └─29364 /usr/share/filebeat/bin/filebeat -c /etc/filebeat/filebeat.yml -...

dic 06 01:17:57 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Started Filebeat sends...
dic 06 01:17:57 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Starting Filebeat send...
Hint: Some lines were ellipsized, use -l to show in full.
```

Figura 5.14 Estado activo del servicio Filebeat

Para configurar Filebeat, hay que editar el archivo de configuración. El archivo predeterminado se llama /etc/filebeat/*filebeat.yml*

```
# nano /etc/filebeat/filebeat.yml
```

Para configurar correctamente Filebeat, es necesario definir la ruta (o rutas) de los archivos de registro o logs que genera nuestro sensor Snort. En la sección sobre instalación de Snort, se tiene los archivos logs los cuales recolectan los eventos ocurridos en nuestro servidor.

Para completar la integración del dispositivo con la infraestructura, se debe indicar a Filebeat que reenvíe la información que genera Snort en */var/log/snort/snort.log*, si no se le cambia el nombre, a nuestro clúster de Elasticsearch. Para llevar a cabo esta tarea se debe editar el siguiente archivo de configuración, */etc/filebeat/filebeat.yml*.

```
- /var/log/snort/snort.log

  document_type:snort
```

Después se añade las siguientes líneas en la sección de Kibana. Ver la figura 5.15

```
host: "10.254.13.100:5601"
```

```
# Kibana Host
# Scheme and port can be left out and will be set to the default (http and 5601)
# In case you specify and additional path, the scheme is required: http://localhost$
# IPv6 addresses should always be defined as: https://[2001:db8::1]:5601
#host: "localhost:5601"
 host: "10.254.13.100:5601"
```

Figura 5.15 Configuración Kibana en Filebeat

Y finalmente se agrega las siguientes líneas en los outputs de Elasticsearch y de Logstash para su correcto funcionamiento.

```
hosts: ["10.254.13.100:9200"]

username: "operador"

password: "<LcTHNeT5678**>"

----------------------------

hosts: ["10.254.13.100:5044"]
```

En nuestro archivo de configuración se mostrará como en la figura 5.16:

```
# Configure what output to use when sending the data collected by the beat.

#----------------------- Elasticsearch output ------------------------------
output.elasticsearch:
  # Array of hosts to connect to.
   hosts: ["10.254.13.100:9200"]
   username: "operador"
   password: "<LcTHNeT5678**>"

  # Optional protocol and basic auth credentials.
  #protocol: "https"
  #username: "elastic"
  #password: "changeme"

#----------------------------- Logstash output --------------------------------
#output.logstash:
  # The Logstash hosts
  #hosts: ["localhost:5044"]
  #hosts: ["10.254.13.100:5044"]
```

Figura 5.16 Configuración outputs de Filebeat

5.4.1.2.4. Instalar Logstash

Primero, se debe dirigir al directorio *Descargas*. A continuación, se procede a descargar e instalar la última versión de Logstash [72].

```
# cd /home/operador/Descargas

# wget https://artifacts.elastic.co/downloads/logstash/logstash-6.5.2.rpm

# yum install logstash-6.5.2.rpm
```

Cuando se complete la instalación, se le solicitará que inicie y habilite Logstash.

```
# systemctl enable logstash

# systemctl start logstash
```

Se verifica que Logstash se encuentre activo. Ver la figura 5.17

```
# systemctl status logstash
```

```
[root@honeywall-tesis /]# systemctl status logstash
● logstash.service - logstash
   Loaded: loaded (/etc/systemd/system/logstash.service; disabled; vendor preset: disa
bled)
   Active: active (running) since vie 2018-12-07 14:40:45 -05; 35s ago
 Main PID: 15716 (java)
    Tasks: 24
   CGroup: /system.slice/logstash.service
           └─15716 /bin/java -Xms1g -Xmx1g -XX:+UseParNewGC -XX:+UseConcMarkSweepGC...

dic 07 14:40:45 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Started logstash.
dic 07 14:40:45 honeywall-tesis.csirt-epn.edu.ec systemd[1]: Starting logstash...
```

Figura 5.17 Estado activo del servicio Logstash

Es importante realizar la configuración de los archivos de entrada, salida y filtro de Logstash. Esto es necesario para que Logstash "aprenda" cómo procesar los datos provenientes de los Filebeat. Se debe dirigir a la siguiente ubicación y allí crear un archivo de configuración tesis.conf. Ver la figura 5.18

```
# cd /etc/logstash/conf.d/

# mkdir tesis.conf
```

```
[root@honeywall-tesis conf.d]# ll
total 12
-rw-r--r--. 1 root root 1072 dic  5 10:52 filter.conf
-rw-r--r--. 1 root root  160 dic  5 10:51 input.conf
-rw-r--r--. 1 root root  133 dic  5 10:55 output.conf
```

Figura 5.18 Configuración de input, filter y output de Logstash

Con un editor de texto, se agrega las siguientes líneas:

tesis.conf

```
 input {

# En estas líneas comentadas, se encuentra la configuración input en caso de utiliza el sensor Suricata

#              file {

#             path => [ "/var/log/suricata/eve.json" ]

#             type => "suricataLog"

#             tags => "suricataLog"

#             codec => "json"

#              }


#Input para el sensor Snort, se toma el archivo alerts.json
```

```
                file {
                path => ["/var/log/snort/alerts.json"]
              codec => json
              type => "snort-json"
  }
} #close input


filter {
# En estas líneas comentadas, se encuentra la configuración filter en caso de utiliza el sensor Suricata
#               if [path] == "/var/log/suricata/eve.json" {
#              json {
#                     source => "message"
#                     target => "parsed"
#              }
#              mutate {
#                     #add_field => { "newid" => "%{[parsed][_id]}" }
#                     add_field => { "[insertion_epoch_timestamp]" => "%{[par$
#              }
#              date {
#                     match => [ "insertion_epoch_timestamp", "UNIX" ]
#              }
#
#               } #close if


#Filter para el sensor Snort
            if [type] == "snort-json" {
                mutate {
                  add_field => {
                    "engine" => "snort"
          }#close add_file
```

```
        }#close mutate

        }#close if

        date {

       match => ["timestamp", "dd/MMM/yyyy:HH:mm:ss Z"]

      locale => "en"

    }

} # close filter


#Output para el sensor Snort

output {

        #stdout { codec => json }

        elasticsearch {

                hosts => ["10.254.13.100:9200"]

                index => "snort-%{+YYYY.MM.dd}" #nombre de salida de index para elasticsearch

        }

} #close output
```

Y para finalizar con la configuración se debe dirigir al directorio */var/log/snort/*
Allí se encuentran los archivos de registro de los datos capturados, ver Figura 5.19

```
[root@honeywall-tesis snort]# ll
total 140
-rwxrwxr-t. 1 snort snort     0 dic 10 23:29 alert
drwsrwxr-t. 2 snort snort     6 dic 11 12:08 archived_logs
-rw-------. 1 snort snort     5 dic 11 12:38 snort_ens224.pid
-rw-------. 1 snort snort     0 dic 11 12:38 snort_ens224.pid.lck
-rw-------. 1 snort snort 10128 dic 11 12:36 snort.u2.1544549756
-rw-------. 1 snort snort 25428 dic 11 12:39 snort.u2.1544549887
-rw-------. 1 snort snort 23466 dic 11 20:13 snort.u2.1544549974
```

Figura 5.19 Registro de datos capturados por Snort

Se puede notar que su extensión es .u2 y no .json como es necesario para el procesamiento de datos. Por lo tanto, es necesario realizar una transformación con la siguiente línea de código.

```
# idstools-u2json /var/log/snort/snort.u2.1544549756 ./* > alerts.json
```

Finalmente se reinicia el servicio Logstash y lo único que quedará por hacer es analizar los datos capturados.

```
# systemctl restart logstash
```

5.4.2. Identificación de Amenazas y Vulnerabilidades

En este capítulo se describe el uso de las herramientas de software libre seleccionadas y la técnica adecuada para dar una solución a nuestro caso de estudio. En primer lugar, se habla del sensor Snort integrado con el Stack ELK, su uso y ventajas. Y después de esto se detalla las vulnerabilidades más comunes en instituciones educativas, las vulnerabilidades encontradas en el análisis de datos. Finalmente se mencionan las complicaciones en el análisis de datos y propuesta de la posibilidad de utilizar otras herramientas.

5.4.2.1. Snort con Stack ELK

Snort produce registros en modo texto y en modo binario, guardando los paquetes que generaron las alertas. El modo texto se puede abrir con cualquier editor de texto, mientras que el binario requiere un visor de paquetes (lo cual se puede realizar con nuestro Stack ELK).

A continuación, en la figura 5.20, se muestra el Stack ELK, ingresar en Kibana tal cual se mostró en la sección anterior y buscar la pestaña *Management*.

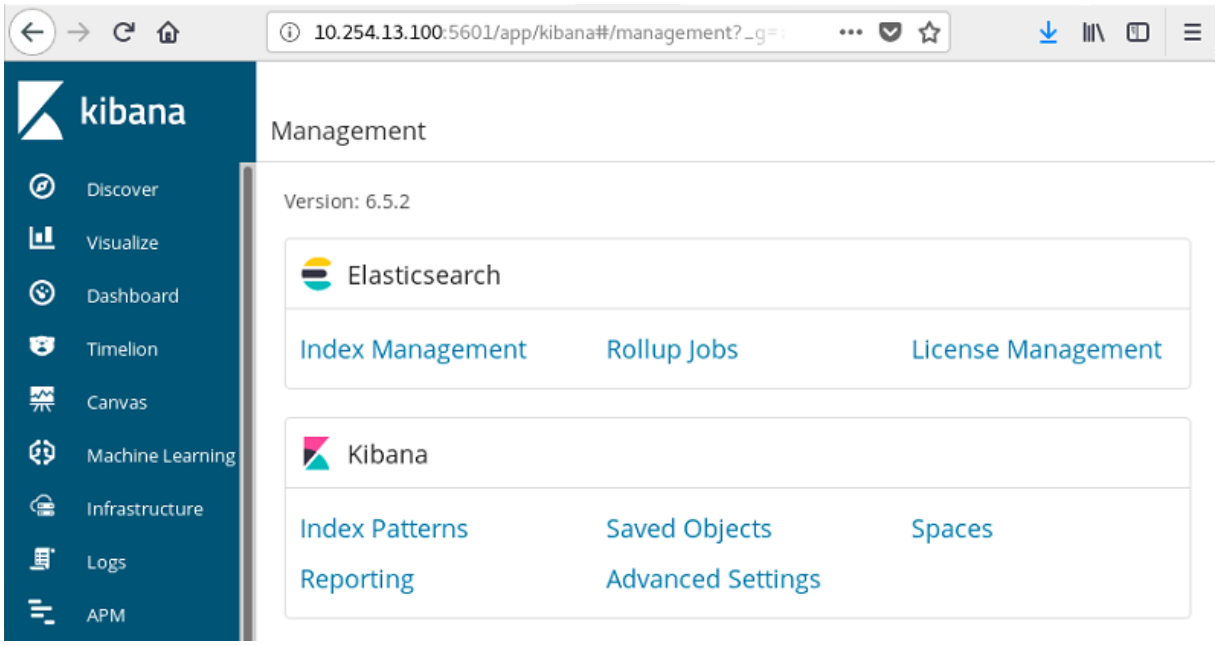


Figura 5.20 Management de Kibana

Ingresar en Index Managenent concerniente a Elasticsearch. Se tiene un índice llamado Filebeat, el cual permitirá la visualización de los eventos capturados por Snort. En Kibana, se crea un índice para luego utilizarlos en la pestaña de visualización, ver la figura 5.21

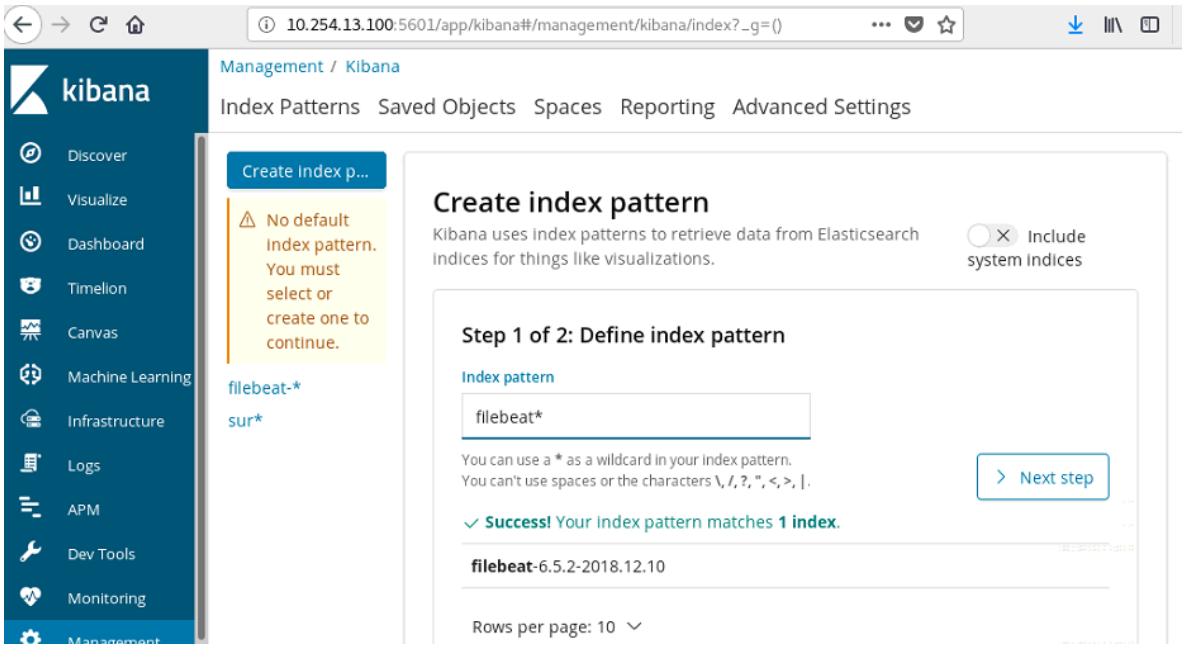


Figura 5.21 Creación Filebeat Index

Ahora aparecerá una nueva pantalla en la cual se presenta una lista de filtros, se selecciona @timestamp, y se da click en Create index pattern. Ver la figura 5.22

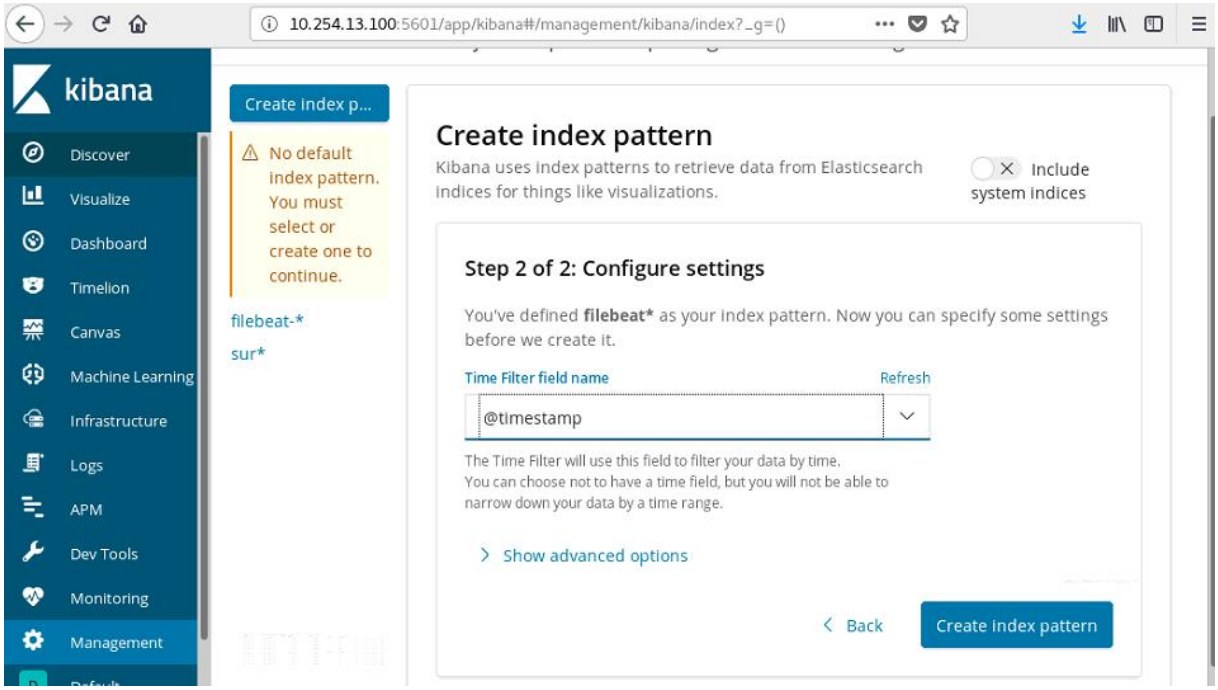


Figura 5.22 Configuración index Filebeat

A continuación, se muestra nuestro índice creado, ver figura 5.23

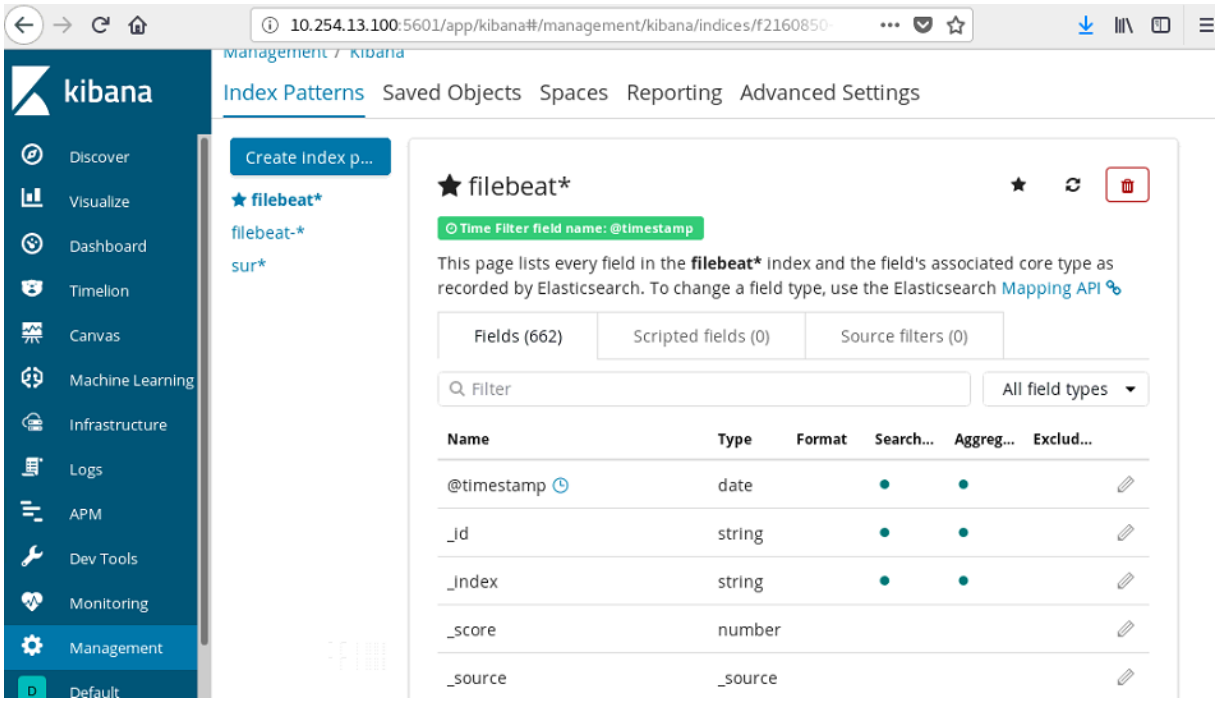


Figura 5.23 Índice Filebeat creado

Cabe destacar que es importante ingresar en la opción Manage de nuestro índice, lo cual permitirá visualizar de qué manera se encuentra mapeados los eventos capturados. Lo cual ayudará elegir los datos de interés en cada evento. Ver figura 5.24

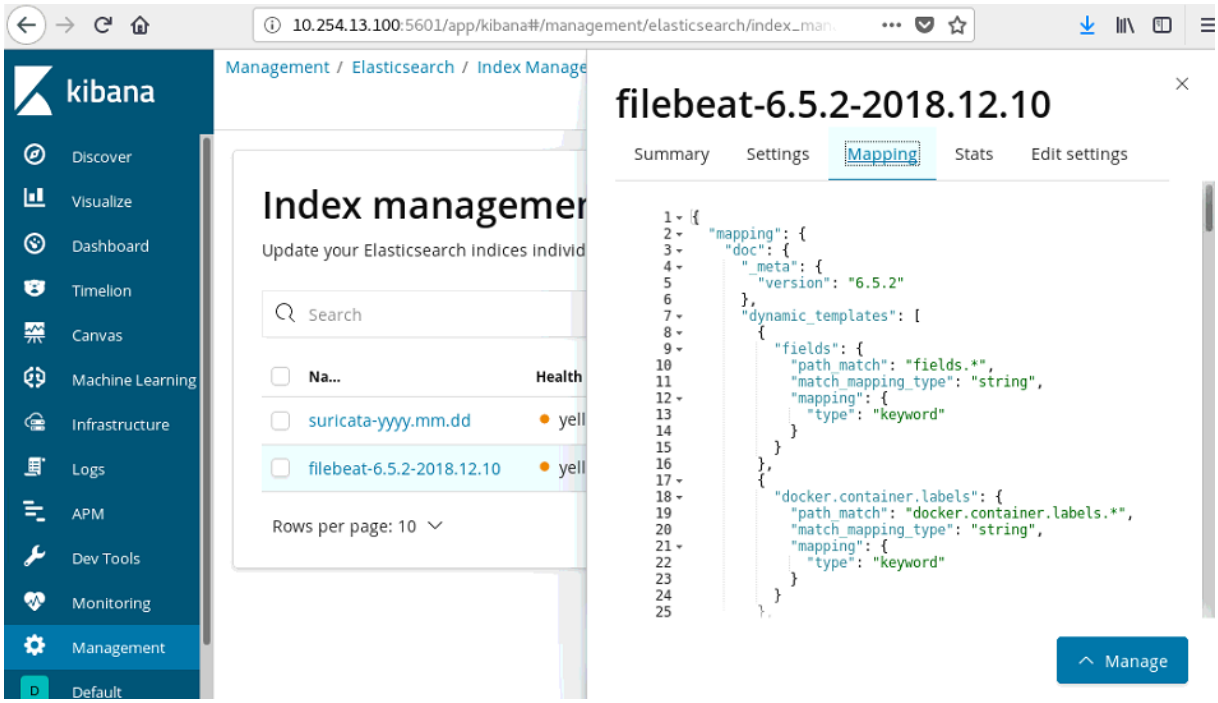


Figura 5.24 Mapping del índice Filebeat

Después hay que ir a la pestaña de visualización, en donde se muestran los eventos de acuerdo a los siguientes parámetros:

- Alert Oveview: Un resumen de las alertas generadas en nuestro sensor, visible en la pestaña Dashboard de Kibana. (ver la figura 5.25)
- Events Overview: Un resumen de los eventos generados en nuestro sensor, visible en la pestaña Dashboard de Kibana.
- Activity Types over Time: Muestra los tipos de actividades a lo largo del tiempo de captura de eventos por parte de nuestro sensor.
- Alert – Destination Location: Muestra la localización geoip de los eventos capturados. (ver la figura 5.26)
- Alert – Source Location: Es muy similar al anterior parámetro.
- Alerts – Top Destination Countries: Muestra las principales alertas de los eventos generados desde mi red a un país de destino.
- Alerts – Top Source Countries: Muestra las principales alertas de los eventos generados desde un país de origen (ver la figura 5.27)
- Event Types: Muestra el tipo de eventos capturados por Snort

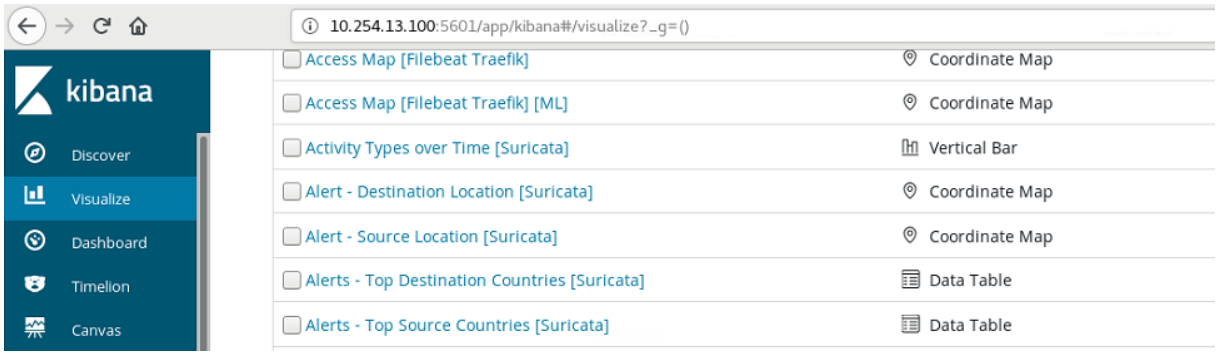


Figura 5.25 Visualización Eventos en Kibana

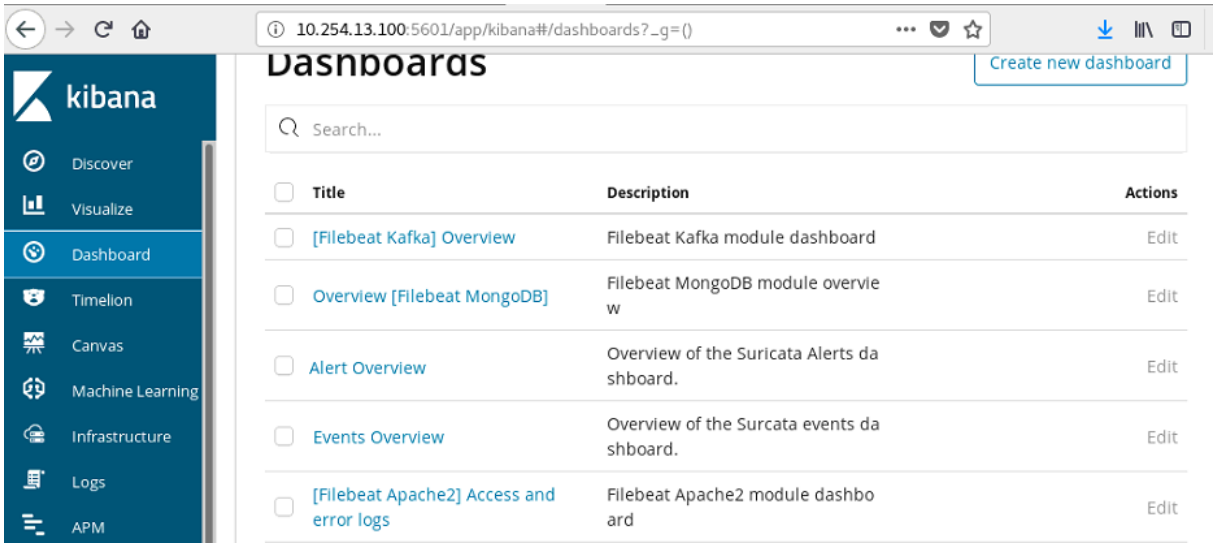


Figura 5.26 Dashboard de Kibana

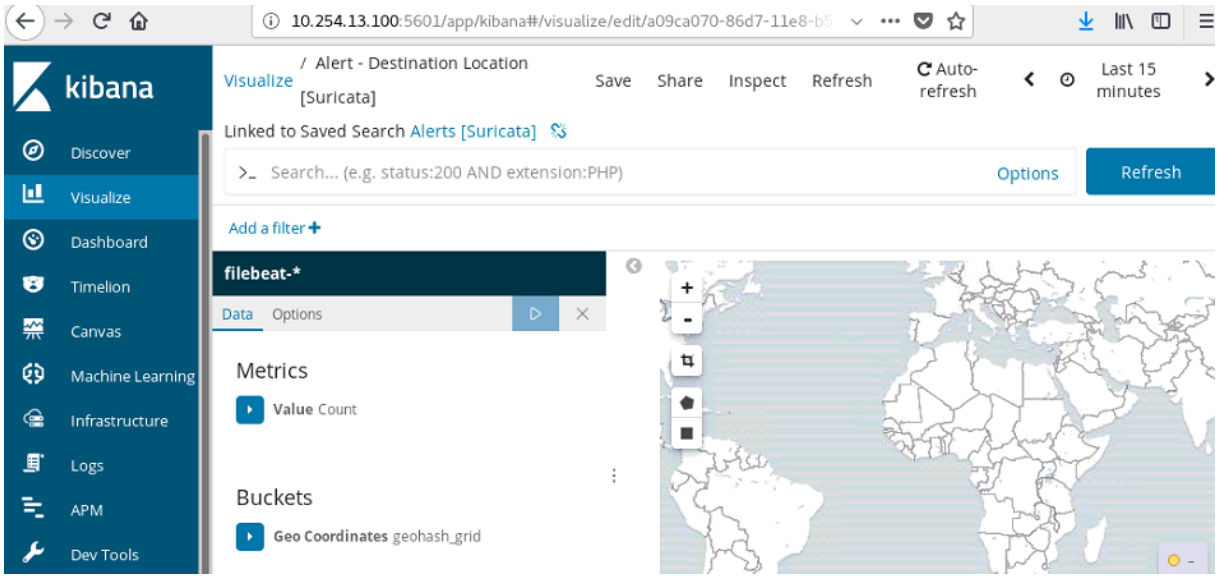


Figura 5.27 Alert – Destination Location

En la pestaña Discover de Kibana, se puede visualizar todos los eventos capturados en nuestro sensor (ver la figura 5.28), pero sin filtros. Estos filtros pueden ser añadidos de acuerdo a la información que se desea mostrar.

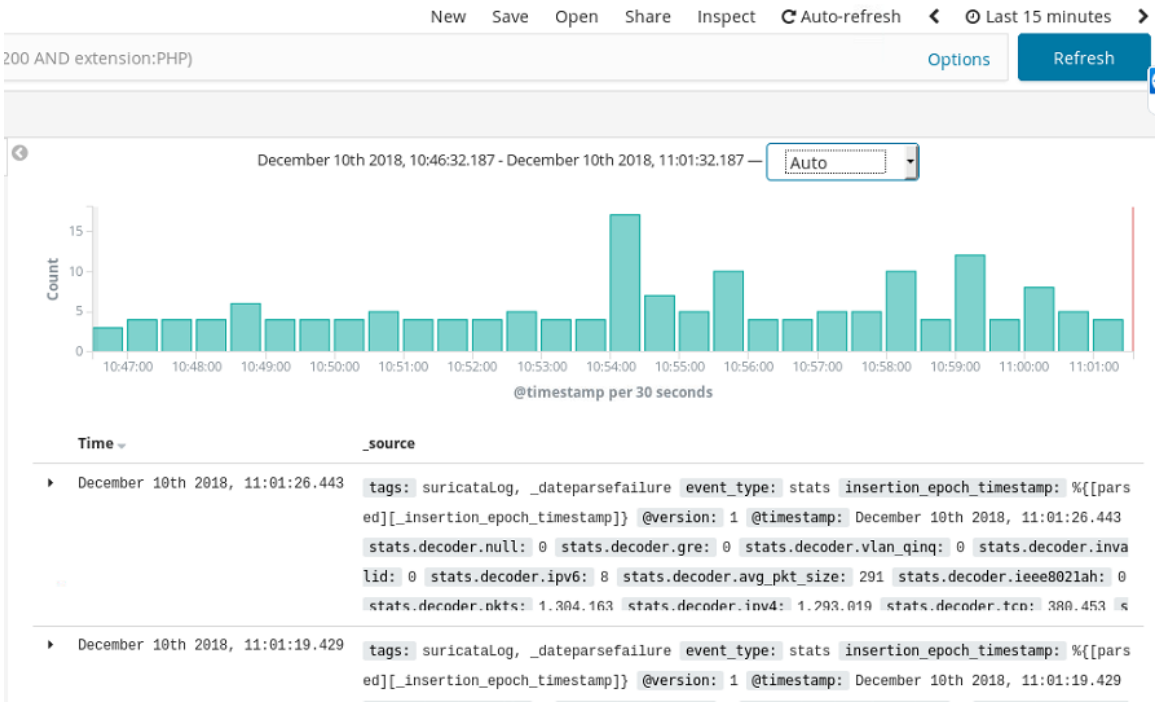


Figura 5.28 Eventos capturados por Suricata

Finalmente, en la pestaña de logs se puede ver nuestros logs capturados por fechas, o a su vez observarlos en tiempo real. Ver la figura 5.29

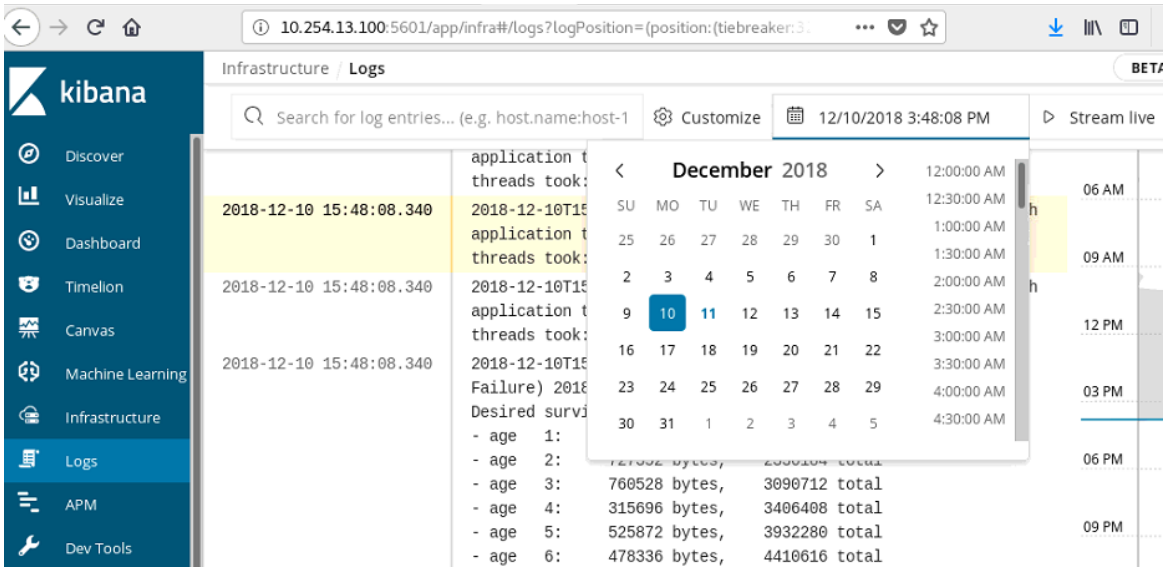


Figura 5.29 Vista de logs por fechas o en tiempo real

El valor que tiene los datos en las actividades de las organizaciones, empresas o instituciones educativas, han dado una importancia superior a la recopilación y análisis de la mayor cantidad posible de datos. El número de eventos generados llegan a ser tan numerosos, que, si se quisiera, analizar con técnicas tradicionales tomará demasiado tiempo. El Stack ELK proporciona un conjunto de utilidades, las cuales tiene un propósito diferente, que, cuando se combinan, crean una potente plataforma de búsqueda y análisis. Es el caso de estudio se tiene un registro centralizado, procesado por un sistema de administración y utilizado para generar información de importancia para la institución educativa.

El Stack ELK permite tener un mejor acceso y procesamiento de datos, útiles para la toma de decisiones centrada en los resultados, a continuación, se muestra las principales ventajas de utilizar Stack ELK:

- Cuenta con una intuitiva interfaz de usuario que agiliza las tareas de análisis de datos para que pueda dedicar tiempo a explorar y revisar los datos, ya que los resultados se presentan de acuerdo a lo que se esté buscando. Entre los gráficos de la interfaz que presenta Kibana, se pueden utilizar barras, gráficos circulares, histogramas, mapas de localización, mapas de calor o a su vez elegir personalizar alguno de ellos.
- Tiene un sin número de filtros para la el uso y administración de logs, y estos pueden escalar según las necesidades de la institución. Lo cual permite distinguir el tipo de ataques y actuar proactivamente para su bloqueo, eliminación o mitigación.

Por otro lado, los enlaces de comunicaciones de una institución educativa pueden ser susceptibles a ataques, por ejemplo, eavesdropping, man-in-the-middle o ataques de denegación de servicio (DoS). Este tipo de ataque es probable a que algunas conexiones de datos están basadas en tecnologías estandarizadas como WiFi. Por eso es importante tener en cuenta que la existencia y uso de enlaces inalámbricos, pueden presentar una vulnerabilidad. Entre las vulnerabilidades más comunes se encuentra:

- Inyección SQL: Inyecciones que se detectan en las bases de datos
- Control de autenticación y acceso: Debido a la utilización de herramientas de ataque o ingeniería social.
- Puertas traseras: un atacante mantiene la opción de acceso remoto a un sistema
- Exposición de datos sensibles: captura de datos sensibles a través de software malicioso como keylogger.

- Insuficiente monitoreo de logs: incorrecta monitorización y gestión de logs para poder valorar la seguridad de las aplicaciones en caso de un ataque o de una intrusión

Gracias al alto ancho de banda que poseen las instituciones educativas es posible ser blanco de los atacantes, el éxito dependerá de las vulnerabilidades existentes. Las vulnerabilidades más comunes son las siguientes:

- Respecto a métodos HTTP: Los ataques HTTP pueden convertirse en ataques de denegación de servicio (DoS) debido a que el atacante envía las solicitudes HTTP de poco en poco a un servidor. Si la solicitud HTTP no es completa, entonces el servidor mantiene sus recursos ocupados a la espera de todos los datos.
- Control de autenticación y acceso: se espera obtener las credenciales con usuario root y así tener control a través de SSH al o los dispositivos de la red.

En nuestro caso de estudio existe la sospecha de la presencia de la vulnerabilidad tipo botnet, debido a cierto comportamiento anómalo de algunos equipos (servidores, computadoras personales, dispositivos inalámbricos de estudiantes). La sospecha se da por comportamientos anormales como:

- Un dispositivo se recalienta, los ventiladores se prenden repentinamente, aunque el procesamiento de la maquina sea muy poco.
- El sistema operativo se pone lento, se cuelga, no responde a las peticiones del usuario.
- Comportamiento anormal de servidor, como peticiones o conexiones a segmentos de red desconocidos o poco habituales.
- Los computadores personales se reinician en cualquier momento
- Las descargas son muy lentas
- Se han desactivado los antivirus y firewalls.

Debido a este tipo de comportamiento se sospecha de la presencia de este tipo de vulnerabilidad, para ello se realiza el análisis de los eventos generados en la red. Y allí es donde un sensor como Snort puede detectar haciendo uso de las reglas definidas o personalizadas. Entre las complicaciones dadas al momento de analizar las vulnerabilidades se encuentra las siguientes:

- Integración Snort con Stack ELK: las configuraciones de cada uno de los programas deben hacer correctamente caso contrario no se puede enviar los datos desde el sensor hasta la interfaz gráfica. Por eso se debe revisar bien los archivos *snort.confd, filebeat.yaml, logstash.yml, elasticsearch.ymal y kibana.yml*
- Análisis de los datos: se requiere hacer un buen uso de los filtros existentes en Kibana para así mostrar los datos relevantes de acuerdo a los tipos de vulnerabilidades encontrados.

Finalmente se puede decir que a pesar que el sensor captura eventos, muchísimos de ellos no son relevantes como por ejemplo el tráfico que sale a un destino o también el tráfico de paquetes de servicios comunes. Y esto no permite visualizar correctamente todo lo esperado. En este caso se pueden hacer uso de otras técnicas o herramientas mencionadas en el capítulo 3 y 4 de este proyecto.

5.4.2.2. Vulnerabilidades tipo botnet detectadas

Después de realizar la captura de datos se llega a la parte en donde se debe analizar los datos para detectar vulnerabilidades o tráfico amenazante también denominado tráfico malicioso. Esta sección nos mostrará principalmente, cómo identificar las vulnerabilidades más comunes y de forma especial aquellas de tipo botnet.

Al referirnos a tráfico malicioso se hace referencia al tráfico que no concuerda con el tráfico normal de una red. El tráfico puede estar fuera de lo normal cuando se encuentra usando un protocolo diferente al habitual, un puerto diferente, la frecuencia con la que este paquete aparece, peticiones irregulares, etc. El tráfico malicioso es aquel que no es habitual en las comunicaciones. Esto puede darse no sólo por ataques e intromisiones en la red, sino que es posible debido al mal comportamiento de ciertas aplicaciones, mala configuración, errores del usuario o aparatos defectuosos.

Para poder identificar el tráfico malicioso, se necesita identificar lo que es un tráfico común. Es aquí donde la institución provee un listado del tráfico que se considera normal. Si no se posee un registro de en la institución, se debe analizar junto con el analista de redes el tráfico capturado, para determinar si es o no un tráfico malicioso, otra alternativa es utilizando una herramienta para análisis de datos capturados y determinar si son normales o no. A continuación, se detallan los tipos de ataques encontrados.

Es importante que los archivos de registro de eventos generados por snort, sean de tipo json. JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos, permite a los usuarios leer y escribir. Este tipo de archivos son fáciles de analizar y generar para el análisis de eventos del tráfico capturado [73]. Aunque existen otros tipos de datos como .csv o .pcap, la mejor opción es .json, gracias a la posibilidad de analizar y mapear los datos que más relevantes como, direcciones ip origen, direcciones ip destino, sitios web, protocolos utilizados, puertos disponibles, entre otros.

5.4.2.2.1. Identificar Ataques tipo DoS (Denegación de servicios)

Un ataque DoS ocurre cuando el atacante intenta evitar que un usuario legítimo de la red, acceda a la información, servicios que esta provee. El caso más común de DoS se da cuando el atacante sobrecarga la red con HTTP request (una llamada a una cierta página en internet), o cuando envía paquetes incompletos por lo que los dispositivos esperan los paquetes restantes. El servidor procesa una cierta cantidad de peticiones a la vez, pero si el servidor con una gran cantidad de otras peticiones, la petición original no va a poder ser procesada.

Ahora si se visualiza las figuras 5.30 y 5.31. Ataque DOS identificado con Wireshark, se tiene un caso en el que se tiene dos máquinas, un cliente y un servidor. El cliente tiene una dirección IP y el servidor tiene otra dirección IP con el puerto 80 en escucha para alguna conexión, al analizar los datos en este segmento de red, y desplegando el menú Statistics->Flow Graph, nos muestra que el cliente está enviando una gran cantidad de mensajes SYN al servidor para realizar una conexión, el servidor trata de resolver la MAC de esta dirección IP algunas veces como se puede observar en el paquete 3231, pero al no recibir la respuesta de la dirección física de la máquina, no puede enviar el ACK-SYN.

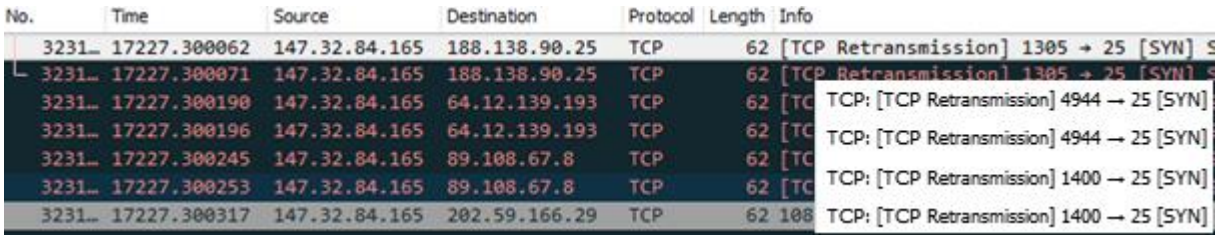


Figura 5.30 Ataque DoS identificado.

Esto lleva a que el servidor espere que le llegue la dirección física del dispositivo que está tratando de establecer la conexión, siga recibiendo peticiones de conexión (mensajes SYN)

que provocará que el servidor por cada petición trate de resolver una nueva conexión. Se puede acabar con los recursos del servidor generando con esto que no admita más solicitudes de conexión, y así anulando al servidor para que ningún otro cliente pueda acceder a algún servicio de este.

5.4.2.2.2. Identificar ataques tipo DNS

Suplantación de identidad por nombre de dominio. Se trata del falseamiento de una relación "Nombre de dominio-IP" ante una consulta de resolución de nombre, es decir, resolver con una dirección IP falsa un cierto nombre DNS o viceversa [74]. Dentro del DNS Spoofing. Todos sufren los mismos problemas de eficacia en función de la ubicación del origen y el destino, así como de la existencia de firewalls al igual que en el caso del DoS en cuanto a suplantación se refiere. En la figura 5.31 se muestra un ataque tipo DNS.

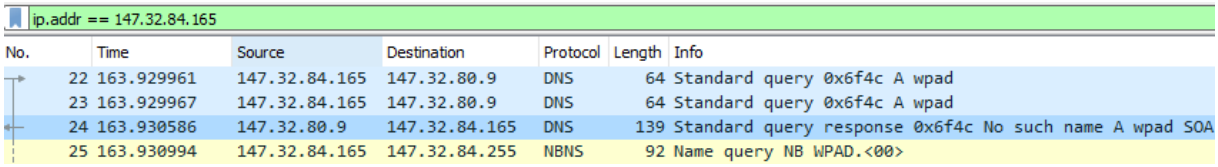


Figura 5.31 Ataque DNS identificado

Los nombres de dominio aparecen en el análisis, algunos de ellos son legítimos, creados por humanos, legibles y relativamente fáciles de recordar. Por ejemplo trajano.us.es. Mientras tanto las posibilidades de que estos dominios sean creados automáticamente son muy bajas. Los dominios AGD tienden a ser caóticos, impronunciables e intercalan consonantes, vocales, números y símbolos a placer. Ejemplos pueden ser añsldkfjañsdlkjasdf.es o j8xmiukqw3—2fla66.tk

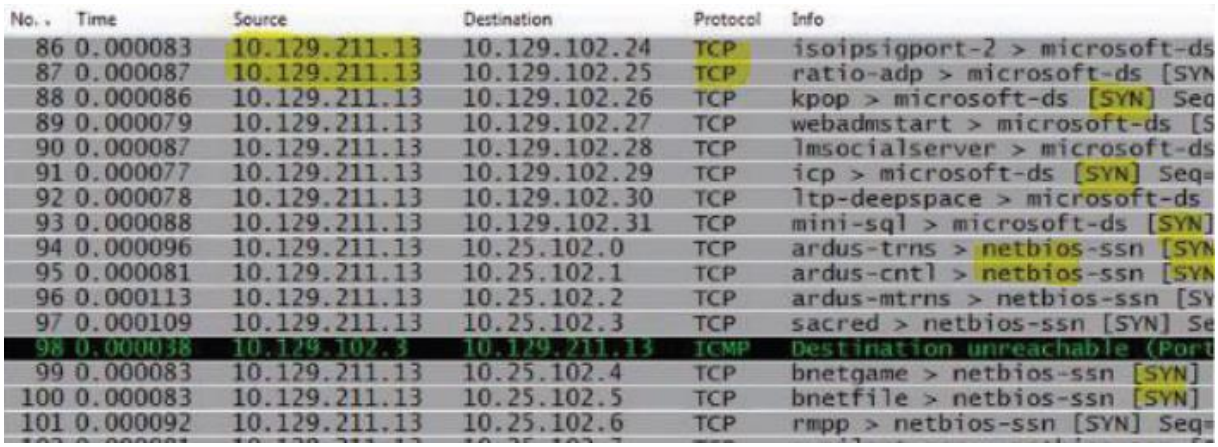


Figura 5.32 Bot malicioso

El tráfico capturado en la Figura 5.32. Escaneo de puertos por bot malicioso, indica un host con dirección 10.129.211.3 enviando paquetes TCP hacia una gran cantidad de dispositivos tratando de encontrar otros dispositivos en la red para poder realizar una conexión con ellos, además los paquetes enviados poseen una bandera SYN característica para establecer una conexión con alguna de ellas. Este escaneo lo está realizando a través del puerto de la NetBIOS (puerto 139).

5.4.2.2.3. Identificar Ataques de ARP Spoofing

Este método se lo puede considerar agresivo ya que generalmente es utilizado como ataque a la red, consistiendo en una intromisión no autorizada en la red entre uno o varios equipos, para capturar, interceptar e incluso modificar los paquetes que llegan a esta red. Este ataque está basado en el protocolo ARP.

Básicamente lo que realiza este protocolo es encontrar la dirección MAC que corresponde a una determinada dirección IP. Para esto lo que se envía es un “ARP request”, a la dirección broadcast de la red (dirección con la que se realiza un llamado a toda la red) que contiene la dirección IP solicitada, y entonces se espera un “ARP Replay” con la dirección MAC que corresponde. Cada equipo al que se hace llamado en la red mantiene la dirección traducida y lista en su caché para así evitar retardo en la carga. Ver la figura 5.33

	Time	Source	Destination	Protocol	Length	Info
7	53.086840	Cisco_db:19:c3	Broadcast	ARP	60	Who has 147.32.84.165? Tell 147.32.84.1
8	59.086131	Cisco_db:19:c3	Broadcast	ARP	60	Who has 147.32.84.165? Tell 147.32.84.1
9	160.084662	PcsCompu_b5:b…	Broadcast	ARP	60	Gratuitous ARP for 147.32.84.165 (Request)
10	160.084668	PcsCompu_b5:b…	Broadcast	ARP	60	Gratuitous ARP for 147.32.84.165 (Request)
11	161.077511	PcsCompu_b5:b…	Broadcast	ARP	60	Gratuitous ARP for 147.32.84.165 (Request)

Figura 5. 33 Detección de ataque ARP Spoofing

Este proceso de ataque puede ser generado no solo por expertos, sino que existen herramientas relativamente fáciles en su manejo las cuales sin tener mucho conocimiento permiten realizar un ataque por lo que el riesgo de un ataque de este tipo es alto.

5.4.2.2.4. Malware

El malware, es usado por cibercriminales, piratas informáticos para irrumpir en las operaciones normales de un dispositivo, robar datos personales, cambiar controles de acceso, y acciones prejudiciales para el o los elementos de una red. Este software malicioso se presenta de una forma de código ejecutable, scripts, contenido activo u otra variante de software. El malware es una amenaza consistente y peligrosa, lo cual ha provocado la investigación y desarrollo de una serie de tecnologías mejoradas para la detección y prevención de este tipo de amenazas, Sin embargo, a medida que mejora la detección, los atacantes siguen transformando sus herramientas y así mantenerse un paso delante de los proveedores de seguridad.

Una opción existente es indagar en la página de VirusTotal [75], la cual nos permite visualizar los eventos detectados por Snort, lo que se evidencia en la figura 5.34. Para mostrar esta información es necesario cargar el archivo log generado por Snort.

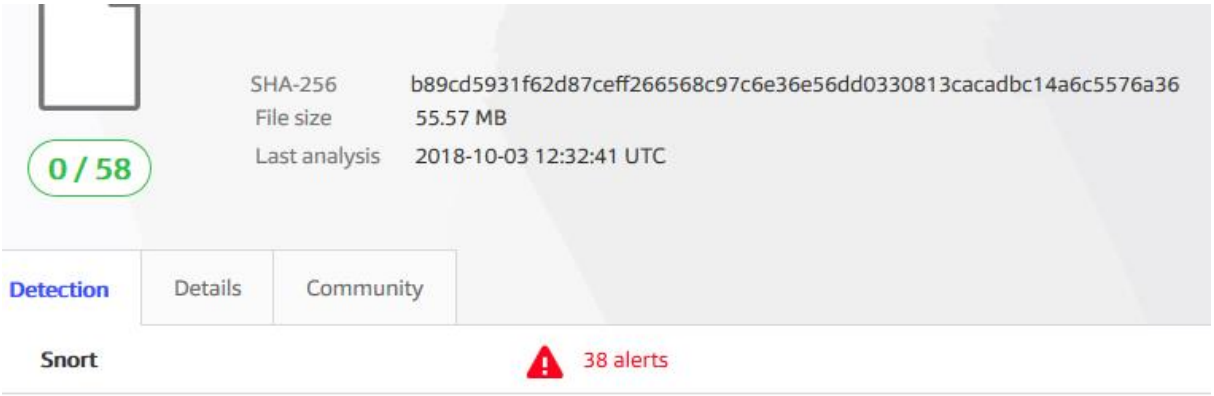


Figura 5.34 Eventos generados por Snort

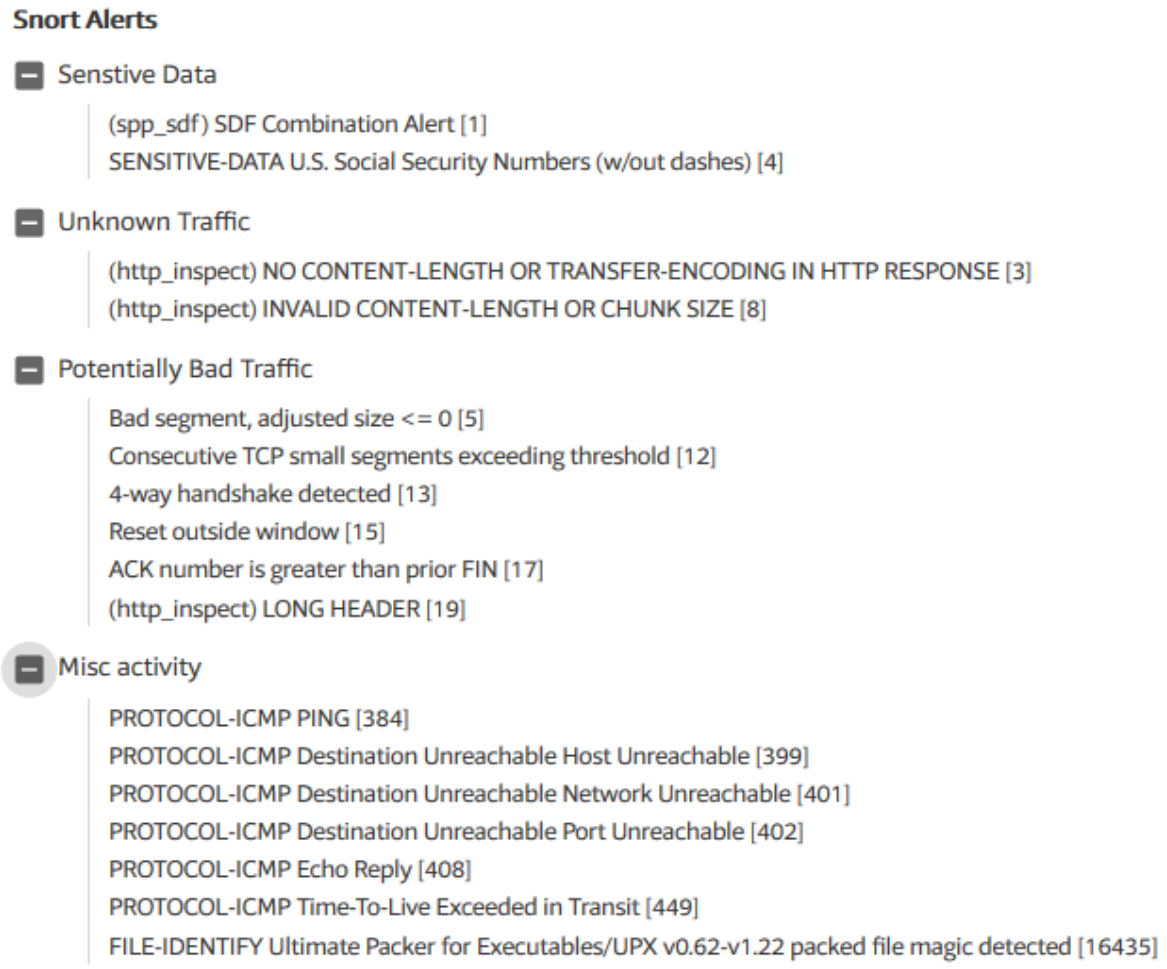


Figura 5.35 Detalle de eventos generados por Snort

En la figura 5.34 se puede visualizar un listado de las vulnerabilidades detectadas por snort, a continuación, se detalla cada una de ellas.

Sensitive Data

- (spp_sdf) SDF Combination Alerts: Esto se da cuando Snort detecta algún tipo de datos confidenciales, esta información puede contener correos electrónicos, claves personales o números de tarjetas de crédito.

Unknwon Traffic

- (http_inspect) INVALID CONTENT-LENGTH OR CHUNK SIZE: No son realmente alertas, más bien son un tipo de información sobre las conexiones. Son falsos positivos en base a la whitelist de http_inspect.

Potentially Bad Traffic

- Consecutive TCP small segments exceeding threshold: Tráfico autorizado que pasa a través del proxy del servidor.
- 4-way handshake detected: Sirve anunciar que se ha detectado un protocolo de enlace TCP de 4 vías.
- ACK number is greater than prior FIN: El cierre de la conexión TCP nunca es un protocolo de enlace de 3 vías sino más bien de 2 vías. Cualquiera de los dispositivos puede iniciar el cierre de la sesión enviando el paquete FIN y se pone en el estado FIN-WAIT1. Luego espera que el otro dispositivo envíe el FIN para que llegue al estado FIN-WAIT2. en este caso, la conexión puede ser reanudada por los otros dispositivos si él no desea terminar la conexión. Cuando ambos dispositivos acuerdan terminar la conexión enviando su FIN y obtiene el ACK para ellos. La conexión se termina.

Misc activity

- PROTOCOL-ICMP PING: ping envía un paquete de solicitud de eco ICMP a una dirección IP. Si un host está activo en esa dirección, responderá con una ICMP Echo Replay. La respuesta incluye la porción de datos del paquete de eco. Los datos incluidos en la solicitud de eco varían según las diferentes implementaciones del sistema operativo.
- PROTOCOL-ICMP Time-To-Live Exceded in Transit: Este evento se genera cuando un dispositivo de enrutamiento detecta que un paquete ha excedido el número máximo de saltos permitidos, esto puede ser una indicación de que un atacante intenta realizar un seguimiento de un host en su red.

6. ESTRATEGIAS DE CONTINGENCIA

Estas estrategias permitirán al personal del CSIRT identificar, mitigar y restablecer los sistemas informáticos ante intrusiones ilegales, caída de servicios, o fallos en la infraestructura tecnológica. Así mismo, los identifica como los únicos autorizados para realizar declaraciones sobre los incidentes ocurridos, sus causas y soluciones temporales y a largo plazo. Las combinaciones de las estrategias permitirán garantizar la operatividad de los dispositivos de la Escuela Politécnica Nacional y por extensión los servicios ofrecidos hacia los usuarios, para asegurar la continuidad de servicios ante atacantes, contemplando la indisponibilidad solo ante pequeños eventos.

6.1. Reconocimiento de botnets

La detección de un ciberataque es una tarea complicada para un usuario normal, por tal motivo, es necesario que el CSRIT con apoyo de la DGIP brinde pautas básicas, no solo en el tratamiento de la información existente en sus estaciones de trabajo, sino también en la información para los usuarios sobre ciertas acciones potencialmente perjudiciales. Se puede reconocer una computadora infectada con una botnet de la misma manera que se identifica una computadora infectada con otros tipos de malware. Las señales incluyen que la computadora funcione lentamente, que actúe de manera extraña, que muestre mensajes de error o que el ventilador se encienda repentinamente cuando la computadora está inactiva. Los posibles síntomas de que alguien usa su equipo de forma remota como parte de una botnet son los siguientes:

- Actividad de Disco Duro inusual.
- Archivos o directorios desconocidos.
- Procesos desconocidos o nombrados como procesos reales.
- Tráfico de red sospechoso, generalmente muy alto.
- Conexiones a una única Dirección IP.
- Alertas masivas del antivirus.
- Archivos sospechosos en carpetas temporales.
- Servicios instalados que no constan en el perfil autorizado.
- Contraseñas modificadas.

Los síntomas listados son válidos para equipos Linux y Windows, así como para servidores. Los ciberataques muchas veces no son detectados hasta que son visibles para el mundo, por ejemplo, el cambio de la página inicial de un sitio web, mostrando un contenido erróneo

o en otros casos no encontrándose disponible, aunque el servicio esté habilitado y funcionado en el servidor.

6.2. Estrategias para detener una botnet

Los análisis de las amenazas internas y externas han llevado a organizaciones, empresas o instituciones educativas a investigar sistemas que apoyen en la supervisión de redes y detección de ataques, entre los que se incluyen estrategias para ayudar a administrar los riesgos en tiempo real. La administración de riesgos de seguridad proporciona un enfoque proactivo que puede ayudar a planificar sus estrategias contra las amenazas existentes.

A la hora de desarrollar estrategias para reducir vulnerabilidades, es importante definir los puntos claves en que se puede implementar la prevención y/o detección. Cuando se trata de administrar riesgos no dependerá únicamente de un sólo dispositivo o tecnología como única línea de defensa. Los métodos preferidos deberían incluir un enfoque de niveles mediante mecanismos proactivos y reactivos a través de la red. Habrá que tener en cuenta lo siguiente para lograr estrategias eficaces de detección:

- Seguridad del edificio. ¿Quién tiene acceso al edificio?
- Seguridad del personal. ¿Cuáles son las restricciones de acceso de un usuario?
- Puntos de acceso a redes. ¿Quién tiene acceso a los equipos en red?
- Equipos del servidor. ¿Quién tiene derechos de acceso a los servidores?
- Equipos de las estaciones de trabajo. ¿Quién tiene derechos de acceso a las estaciones de trabajo?

Si cualquiera de estos elementos está en peligro, hay más riesgo de que una vulnerabilidad eluda los límites de defensa interna y externa para infectar un equipo de red. La protección del acceso a las instalaciones y a los sistemas informáticos debería ser un elemento fundamental de las estrategias de detección. Las estrategias para detener riesgos de botnets pueden abarcar todas las tecnologías y técnicas hasta ahora abordadas en este proyecto. Y una vez que se han validado las estrategias y las tecnologías implementadas, se deben aplicar revisiones de software y hardware cuando sea necesario para lograr una eficacia de seguridad continua. Los usuarios, y especialmente el personal del CSIRT, deben siempre estar al día de las últimas actualizaciones.

La colaboración entre proveedores de internet, administradores de red, usuarios, estudiantes, docentes y el CSIRT en conjunto con la DGIP. Va a permitir comprender las

vulnerabilidades y aplicar estrategias para detección de los riesgos de manera que se reduzca la posibilidad de éxito de un ataque. Es de suma importancia que cada actor de la comunidad politécnica que accede a equipos tecnológicos tome en cuenta las siguientes estrategias para la prevención, detección y eliminación de botnets.

1. En un equipo personal

- Tener siempre actualizado el sistema operativo y evitar hacer uso de software ilegal.
- Tener instalado en el equipo un software antivirus confianza y utilizar un antimalware, de esta manera se pueden advertir infecciones en el tráfico de red.
- Configurar el software antivirus y antimalware para que se actualicen automáticamente al igual que una configuración de revisiones periódicas.
- Contratar un servicio de filtrado web. Los servicios de filtrado web son una de las mejores formas de combatir los bots. Estos servicios exploran en busca de sitios web que exhiban un comportamiento inusual o actividad maliciosa conocida y bloquean esos sitios de los usuarios. Estos servicios monitorean Internet en tiempo real para encontrar sitios web involucrados en actividades sospechosas, como: descargar JavaScript, realizar capturas de pantalla. Algunos servicios tienen la capacidad de notificar a los usuarios que se ha descubierto malware.
- Modificar regularmente las contraseñas de cuentas personales.
- Evitar realizar descargas P2P o Torrent porque, en muchas ocasiones, es la principal entrada de malware botnet.
- Realizar un análisis con el antivirus, permitirá localizar el malware botnet y lo eliminará.
- Para combatir las botnets es importante concientizar sobre este tipo de vulnerabilidades. Es necesario que todo usuario pueda estar atento si el equipo que utiliza está infectado, este puede causar ataques. Por lo tanto, cada vez que se encuentra un equipo vulnerado, es necesario desconectar de la red, analizar con las herramientas de software instaladas (antivirus, antimalware) y si fuese necesario reportar el incidente al CSIRT.

2. En un navegador web

- Utilizar navegadores que no permitan actividades de malware. Como Mozilla Firefox. La misma estrategia puede ser aplicada en sistemas operativos. Es importante mantenerlos constantemente actualizados.

- Una estrategia más completa es deshabilitar los scripts por completo, aunque esto podría influir en el rendimiento de la productividad.
- Los usuarios que utilicen navegadores como Mozilla Firefox, pueden aplicar extensiones para proteger el navegador y no permitir que JavaScript, Java y otro contenido se ejecute sin autorización. De esta manera se evita la explotación de vulnerabilidades de seguridad sin dejar a un lado la navegabilidad.
- Microsoft Internet Explorer 7.0 incluye un bloqueador de ventanas emergentes, un filtro de phishing y varios niveles de seguridad. Además, se niega a descargar automáticamente imágenes dentro de correos a menos que lo acepte el usuario.
- Evitar dar clic en enlaces de correos de fuentes desconocidas (spam). En la mayoría de los casos, el usuario es direccionado a un sitio web donde el código malicioso puede crear vulnerabilidades. Si los usuarios permiten la ejecución de código malicioso, entrará automáticamente al sistema y convertirá el equipo en un bot para fines potencialmente peligrosos.
- Utilizar algoritmos de encriptación para él envió de información sensible.

3. En una organización o institución educativa

Las técnicas de protección de una red y sus equipos infectados se tratan con las siguientes estrategias. En un principio se asume que TODOS los equipos están infectados, a la espera de la notificación que confirme o niegue esta sospecha.

- Si se quiere investigar sobre el bot hay que aislarlo de la red para evitar la propagación del malware, realizar un análisis adecuado y aplicar las medidas de tratamiento necesarias.
- Hay que valorar la posibilidad de formatear los equipos y acabar con el malware detectado.
- Innovar en la infraestructura tecnológica para la adaptación dinámica a las crecientes vulnerabilidades.
- Promover una cultura de prevención, detección y mitigación de los posibles ataques.
- Promover y apoyar acuerdos entre las comunidades de seguridad, infraestructura y tecnología operativa en instituciones educativas del país y en todo el mundo.
- Aumentar la conciencia y la educación de la comunidad educativa o empresarial.

4. En general:

- Todos los equipos deben tener actualizado su sistema operativo, incluyendo las actualizaciones disponibles.

- Todas las aplicaciones de software deben ser licenciadas y estar actualizadas.
- Hacer uso de un firewall.
- Es aconsejable disponer de sistemas IDS e IPS con reglas específicas para botnets.
- La red deberá estar bien configurada al igual que sus equipos de acuerdo a las características que estos posean.
- Los usuarios no deberían ejecutar contenido desconocido y mucho menos con privilegios de administrador.
- Las comunicaciones de cualquier tipo deberían pasar a través de servicios proxies.
- Solo deberían permitirse consultas a DNS internos o confiables.

CONCLUSIONES Y RECOMENDACIONES

A continuación, se redactan las conclusiones y recomendaciones finales del proyecto.

Conclusiones

Las botnets son un tipo de malware que, debido a sus características de funcionamiento, se vuelven más peligrosas a medida que aumentan los dispositivos comprometidos. Los ataques que son producto de botnets pueden dejar fuera de línea cualquier sistema conectado a Internet. Imposibilitando la prestación del servicio a los distintos usuarios.

En comparación con otro tipo de vulnerabilidades, las botnets son más difíciles de controlar y eliminar, la detección se ha convertido en un problema complicado. En este trabajo se ha propuesto una solución para la detección, un seguimiento de los eventos capturados por un sensor Snort que envía dichos eventos a un administrador de estos, permitiendo visualizar estadísticas del tipo de ataque, dirección ip del atacante y así se puede minimizar el impacto en la organización.

La detección y seguimiento de dispositivos comprometidos en una botnet seguirá siendo una tarea difícil. La clasificación de tráfico malicioso es útil para identificar redes de bots. Es necesario una base de conocimiento actualizada para todos los bots lanzados en el mundo, que parece ser una misión imposible.

Hay también algunos otros temas que necesitan ser considerados. A lo mejor no se pueden evitar ataques derivados de botnets. Incluso si este ha sido detectado, no hay ninguna manera eficaz de rastrear o luchar contra él. En cambio, sólo se pueden bloquear los dispositivos comprometidos, a la espera de otros comandos como el escaneo de virus o reinstalar el sistema operativo. De hecho, lo que se necesita en primer lugar es evitar la propagación de los bots a través de la implementación de protocolos en los routers de la red local.

Una herramienta poderosa como Stack ELK permite realizar un análisis de una gran cantidad de eventos generados en la red, es de gran apoyo para dar una solución adecuada al problema que motivo este proyecto.

Cabe tomar en cuenta que se utilizaron otras herramientas para complementar la información. Las herramientas fueron Wireshark y la página VirusTotal, las cuales ayudan para el análisis de tráfico de red.

En este trabajo, se ha detectado botnets en la red, las cuales tienen el mismo comportamiento en red que distingue del tráfico normal. Así, se puede crear varios clusters (Elasticsearch) y eventos de la red (Snort) y encontrar un comportamiento similar y detectar políticas de clusters de botnet basado en filtrado básico, blacklist y whitelist, patrones, motor de correlación, reglas e información.

Recomendaciones

Antes de la aplicación de la solución propuesta para realizar la detección de vulnerabilidades tipo botnet, se recomienda realizar un adecuado análisis de todas las técnicas existentes y elegir la mejor dependiendo de las necesidades de la organización. Al igual que un correcto levantamiento de activos requeridos para el análisis de datos del sistema. Esta solución debe evaluarse con el jefe del área o coordinador del departamento, presentando una hoja de ruta en la cual se detalla la arquitectura y el software adecuado para el respectivo análisis de datos. Recordando que se está manejando información sensible para la organización, por lo que se debe también realizar el correcto resguardo de activos para evitar causar algún problema.

Cuando los dispositivos infectados no se comportan como usualmente lo hacen, puede ser difícil de detectar una posible amenaza. Puesto que la actual tecnología de detección depende de evento ocurrido, no existe una garantía de encontrar cada dispositivo comprometido, sobre todo porque muchos de ellos son dispositivos móviles (laptops, smartphones, tablets). Un tema interesante sobre detección de botnets es la eficiencia de tiempo. Si se presenta un ataque y se puede capturar en primer lugar la anomalía y solucionar los problemas pertinentes antes de su uso para fines maliciosos, se logra que la detección de esta anomalía sea eficiente.

El IDS Snort es una herramienta eficaz de detección de intrusiones en la red debido a que puede controlar el comportamiento anormal. En este trabajo se utilizaron reglas predefinidas por el sensor. Y también se pueden aplicar reglas personalizadas de detección de botnets en función de las actividades maliciosas capturadas por el sensor.

REFERENCIAS

[1] R. A. R. S. L. M. S. K. A. B. M. G. Sundara Krishnan, Computational Intelligence, Syber Security and Computacional Models, New Delhi: Springer India, 2014.

[2] C. G. R. a. I. d. Seguridad, «CSIRTEPN,» [En línea]. Available: https://csirt.epn.edu.ec/index.php/servicios/vulnerabilidades/27-botnets. [Último acceso: 26 09 2017].

[3] S. Cobb, «welivesecurity,» 27 10 2014. [En línea]. Available: https://www.welivesecurity.com/laes/.

[4] I. Society, «Internet Society,» 01 02 2016. [En línea]. Available: https://www.internetsociety.org/sites/default/files/ISOC-PolicyBrief-Botnets-20151030-es.pdf.

[5] S. García, «Detección de botnets basada en algoritmos genéticos,» Instituto de investigación ISISTAN, Facultad de Ciencias Exactas, Universidad Nacional del Centro de la Provincia de Buenos Aires, Buenos Aires, Argentina, 2011.

[6] K. Lab, «Ataques DDoS en el segundo trimestre de 2018,» SecureList, 24 07 2018. [En línea]. Available: https://securelist.lat/ddos-report-in-q2-2018/87217/.

[7] K. Lab, «Los ataques DDoS en el tercer trimestre de 2015,» Securelist, 2015. [En línea].

[8] I. 27001, Estandar Internacional ISO, 2015.

[9] C. y. Tecnología, «Universidad Internacional de Valencia,» 18 04 2018. [En línea]. Available: https://www.universidadviu.es/las-4-claves-la-seguridad-la-informacion/. [Último acceso: 02 08 2018].

[10] I. 13335, ISO/IEC 13335-1, 2004.

[11] TECON, «Soluciones Informaticas,» [En línea]. Available: https://www.tecon.es/la-seguridad-de-la-informacion/. [Último acceso: 02 08 2018].

[12] A. Mendoza, «gb Advisors,» 24 01 2018. [En línea]. Available: http://www.gb-advisors.com/es/control-de-activos-de-ti/. [Último acceso: 02 08 2018].

[13] M. d. H. y. A. Publicas, «CCN - CERT,» de *MAGERIT LibroI - Método*, 2016, p. 127.

[14] P. J. Torres, «Nociones de la Seguridad de la Información según ISO 27001,» de *Conceptos seguridad informacion ISO 27000*, 2015.

[15] A. Reinoso, «Repositorio Digital EPN,» 28 07 2017. [En línea]. Available: http://bibdigital.epn.edu.ec/handle/15000/17542. [Último acceso: 03 08 2018].

[16] I. 27035, INTERNATIONAL STANDARD ISO/IEC 27035-2, 2016.

[17] L. E. Valdivia, «Universidad Nacional Autónoma de México,» 2013. [En línea]. Available: http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/3042/Tesis.pdf?sequence=1. [Último acceso: 02 08 2018].

[18] K.-K. R. Choo, «Australian Intitute of Criminology,» [En línea]. Available: https://aic.gov.au/publications/tandi/tandi333. [Último acceso: 02 08 2018].

[19] H. H. M. H. F. Zahra Bazrafshan, «A Survey on Heuristic Malware Detection,» Department of Computer Science and Engineering Shiraz University, Irán, 2013.

[20] Y. FM, «XATAKA BASICS,» 2 04 2017. [En línea]. Available: https://www.xataka.com/basics/que-es-un-ataque-ddos-y-como-puede-afectarte. [Último acceso: 05 08 2018].

[21] D. Security, «Techopedia,» [En línea]. Available: https://www.techopedia.com/definition/23763/spamming. [Último acceso: 05 08 2018].

[22] J. Pagliery, «CNN Tech,» 20 12 2016. [En línea]. Available: https://money.cnn.com/2016/12/20/technology/ad-fraud-online-methbot/.

[23] Google, «AdSense Help,» [En línea]. Available: https://support.google.com/adsense/answer/16737?hl=en. [Último acceso: 05 08 2018].

[24] P. Security, «Zone Alarm,» 11 03 2015. [En línea]. Available: http://www.zonealarm.com/blog/2015/03/software-update-malware/. [Último acceso: 05 08 2018].

[25] J. Marcos, «Redseguridad,» [En línea]. Available: http://www.redseguridad.com/sectores-tic/industria-y-utilities/ciberseguridad-en-el-espionaje-industrial. [Último acceso: 05 08 2018].

[26] «Kaspersky Lab,» 12 09 2017. [En línea]. Available: https://www.kaspersky.com/about/press-releases/2017_mining-botnets-are-back-infecting-thousands-of-pcs. [Último acceso: 05 08 2018].

[27] R. B. S. S. Ahmad Karim, «Botnet detection techniques: review, future trends, and issues*,» Faculty of Computer Science and Information Technology University of Malaya, Kuala Lumpur, Malaysia, 2014.

[28] A. Berger, «Simon Fraser University,» [En línea]. Available: https://www.cs.sfu.ca/~mhefeeda/Papers/npsec09.pdf. [Último acceso: 29 09 2018].

[29] E. Ripoll, «Detección y bloqueo de botnets basada en el tráfico de red,» 2015.

[30] G. Gu, R. Perdisc, J. Zhang y W. Lee, «Usenix,» [En línea]. Available:

https://www.usenix.org/legacy/event/sec08/tech/full_papers/gu/gu_html/index.html. [Último acceso: 06 08 2018].

[31] H. Project, «Honeynet Project,» [En línea]. Available: http://old.honeynet.org/papers/virtual/. [Último acceso: 19 09 2018].

[32] C. d. h. d. a. y. b. interacción, «Repositorio EPN,» 2016. [En línea]. [Último acceso: 20 09 2018].

[33] R. Torres, «Repositorio ESPE,» 21 01 2014. [En línea]. Available: https://repositorio.espe.edu.ec/bitstream/21000/10470/1/T-ESPE-048394.pdf. [Último acceso: 19 09 2010].

[34] Gonzalez, «DGonzalez Paper,» [En línea]. Available: https://www.dgonzalez.net/papers/ids/html/cap04.htm. [Último acceso: 20 09 2018].

[35] P. V. F. T. Hossein Rouhani Zeidanloo, «Botnet Detection Based on Traffic Monitoring,» International Conference on Networking and Information Technology, Kuala Lumpur, 2010.

[36] V. A. S. Arnur G. Tokhtabayev, «Non-Stationary Markov Models and Anomaly Propagation Analysis in IDS,» IEEE , Manchester, UK, 2007 .

[37] J. C. M. Elizabeth Stinson, «Characterizing Bots' Remote Control Behavior,» Standford, 2008.

[38] S. C. Lei Liu, «BotTracer: Execution-Based Bot-Like Malware Detection,» Springer-Verlag Berlin, Heidelberg ©2008, Taipei, Taiwan, 2008.

[39] M. Szymczyk, «Detecting Botnets in Computer Networks Using Multi-Agent Technology,» IEEE, 2009.

[40] Y. Z. Y.-j. O. Ying Lin, «The Design and Implementation of Host-Based Intrusion Detection System,» IEEE , Jinggangshan, China, 2010.

[41] K. K. S. Murugan, «System and methodology for unknown malware attack,» ResearchGate, 2011.

[42] J. H. G. Creech, «A Semantic Approach to Host-Based Intrusion Detection Systems Using Contiguous and Discontiguous System Call Patterns,» IEEE, 2014.

[43] R. B. S. M. S. S. A. A. S. Ahmad Karim, «Botnet detection techniques: review, future trends, and issues*,» Faculty of Computer Science and Information Technology, Kuala Lumpur, Malaysia, 2014.

[44] M. A. Sridhar Venkatesan, «Detecting Stealthy Botnets in a Resource-Constrained,» Center for Secure Information Systems George Mason University Fairfax, VA, USA, 2017.

[45] S. García, «Detección de botnets basada en algoritmos genéticos,» Instituto de investigación ISISTAN, Facultad de Ciencias Exactas, Universidad Nacional del Centro de la Provincia de Buenos Aires, Buenos Aires, Argentina, 2011.

[46] I. C. D. D. W. L. Roberto Perdisci, Detecting Malicious Flux Service Networks through Passive Analysis of Recursive DNS Traces, Honolulu, HI, USA : IEEE, 2009.

[47] T. E. D. Su Chang, P2P botnet detection using behavior clustering & statistical tests, Chicago, Illinois, USA: ACM New York, NY, USA, 2009.

[48] Y. X. Q. K. Fang Yu, SBotMiner: Large Scale Search Bot Detection, Mountain View, CA 94043 USA: Microsoft Research Silicon Valley, 2010.

[49] R. P. J. Z. W. L. Guofei Gu, BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection, San Jose, California: USENIX Association Berkeley, CA, USA, 2008.

[50] J. Binkley, «Ourmon - Network Monitoring and Anomaly Detection System,» [En línea]. Available: http://ourmon.sourceforge.net/. [Último acceso: 26 09 2018].

[51] K. Binkley, «sourceforge.net,» 12 08 2005. [En línea]. Available: https://sourceforge.net/projects/ourmon/. [Último acceso: 26 09 2018].

[52] K. B. D. H. G. E. Craig Schiller, Botnets - The Killer Web App, SYNGRESS, 2007.

[53] R. Fekolkin, «Intrusion Detection and Prevention Systems: Overview of Snort and Suricata,» Luleå University of Technology, Luleå, Suecia, 2016.

[54] D. Patel, «Implement Suricata IDS/IPS with ELK,» SecPy Blog, 2016.

[55] 2. Cisco, «Snort Org,» [En línea]. Available: https://www.snort.org/. [Último acceso: 14 10 2018].

[56] EcuRed, «EcuRed,» [En línea]. Available: https://www.ecured.cu/Snort. [Último acceso: 14 10 2018].

[57] P. C. Polainos, «e-REdING,» ETSI de la Universidad de Sevilla, [En línea]. Available: http://bibing.us.es/proyectos/abreproy/12077/fichero/memoria%252Fpor_capitulos%252F04.snort.pdf. [Último acceso: 15 10 2018].

[58] S. Org, «Suricata Open Source IDS / IPS / NSM engine,» [En línea]. Available: https://suricata-ids.org/. [Último acceso: 18 10 2018].

[59] T. B. Project, «Bro Network Security Monitor,» [En línea]. Available: https://www.bro.org/sphinx/intro/index.html#features. [Último acceso: 17 10 2018].

[60] Bricata, «Bricata Headquarters,» [En línea]. Available: https://bricata.com/blog/what-is-bro-

ids/. [Último acceso: 22 10 2018].

[61] O. Uludag, «Master's Thesis: Descriptive study and experimental analysis of the ELK stack applicability for Big Data use cases,» Technische Universität München, Germany, 2016.

[62] Elastic, «Elasticsearch B.V.,» [En línea]. Available: https://www.elastic.co/guide/en/logstash/current/introduction.html. [Último acceso: 02 11 2018].

[63] developerlover, «developerlover.com,» [En línea]. Available: http://developerlover.com/monitorizacion-logs-stack-elk-elasticsearch-logstash-kibana/. [Último acceso: 03 11 2018].

[64] Kibana, «Elastic B.V.,» [En línea]. Available: https://www.elastic.co/guide/en/kibana/current/introduction.html. [Último acceso: 03 11 2018].

[65] D. C. P. Pozo, «DSpace JSPUI - Escuela Politécnica Nacional,» 25 01 2016. [En línea]. Available: http://bibdigital.epn.edu.ec/handle/15000/15030. [Último acceso: 24 10 2018].

[66] J. Ruostemaa, «How to install Snort on Centos7,» UpCloud, [En línea]. Available: https://upcloud.com/community/tutorials/installing-snort-on-centos/. [Último acceso: 30 11 2018].

[67] G. Cánepa, «TecMint,» 7 9 2016. [En línea]. Available: https://www.tecmint.com/install-elasticsearch-logstash-and-kibana-elk-stack-on-centos-rhel-7/. [Último acceso: 12 11 2018].

[68] D. Berman, «The Complete Guide to the ELK Stack – 2018,» logz.io, 26 6 2018. [En línea]. Available: https://logz.io/learn/complete-guide-elk-stack/#intro. [Último acceso: 15 11 2018].

[69] Elasticsearch, «Elasticsearch Reference,» 2018. [En línea]. Available: https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html. [Último acceso: 20 11 2018].

[70] Elasticsearch, «Kibana User Guide,» 2018. [En línea]. Available: https://www.elastic.co/guide/en/kibana/current/index.html. [Último acceso: 20 11 2018].

[71] Elasticsearch, «Filebeat Reference,» 2018. [En línea]. Available: https://www.elastic.co/guide/en/beats/filebeat/current/index.html. [Último acceso: 20 11 2018].

[72] Elasticsearch, «Logstash Reference,» 2018. [En línea]. Available: https://www.elastic.co/guide/en/logstash/current/index.html. [Último acceso: 20 11 2018].

[73] D. Crockford, «Introducing JSON,» [En línea]. Available: https://www.json.org/. [Último acceso: 27 12 2018].

[74] F. d. I. d. Coruña, «Ataques a servidores DNS,» [En línea]. Available: https://www.tic.udc.es/~nino/blog/psi/2010/ataque-dns.pdf. [Último acceso: 10 12 2018].

[75] «Analyze suspicious files and URLs to detect types of malware,,» [En línea]. Available: https://www.virustotal.com. [Último acceso: 10 12 2018].