TESIS FINAL DE GRADO

Análisis de la eficiencia de los IDS open source Suricata y Snort en las PYMES

Propuesta de artículo presentado como requisito parcial para aprobar la maestría:

Auditoria en tecnologías de información

Por los estudiantes:

Alfonso Zambrano B.
Franklin Guailacela.

Bajo la dirección de:

.
Rubén Pacheco

Universidad Espíritu Santo
Facultad de Ingeniería en Sistemas Telecomunicaciones y Electrónica.
Guayaquil - Ecuador

Febrero de 2019

Análisis de la eficiencia de los IDS open source Suricata y Snort en las PYMES

Analysis of the efficiency of open source IDS Suricata and Snort in PYMES

Alfonso Zambrano B.[1]
Franklin Guailacela[2]

Resumen

El avance de las tecnologías de información en redes de computadoras, las ha convertido en recursos estratégicos de apoyo para los objetivos de una empresa y al mismo tiempo en objetivos de ataques informáticos, por ejemplo, los equipos de borde o acceso a Internet que se han limitado a cerrar o encubrir puertos de comunicación. De ahí la necesidad de un equipo que alerte y permita tomar medidas de prevención o respuesta, como los sistemas de detección de intrusos (IDS), que son una capa adicional de seguridad precisamente para el aumento de ataques y amenazas reportados anualmente por las firmas antivirus. El objetivo de este artículo es analizar la eficiencia de dos IDS, Snort y Suricata, dentro de una infraestructura virtualizada con una configuración que organice el tráfico de paquetes para su mejor análisis y dispositivos de almacenamiento que minimicen la latencia de escritura y lectura de datos, de manera que se pueda determinar que IDS en condiciones de altas cargas de trabajo es el más eficiente y así una empresa pueda tomar la decisión más compatibles con sus objetivos estratégicos.
Como resultados se concluyó que el redireccionamiento del tráfico le resume trabajo al IDS con la ayuda de la segmentación de redes, además la utilización de un disco sólido que elimina la alta latencia de lectura y escritura y optimiza el tiempo de análisis de la detección, por otra parte el tiempo de detección, consumo de memoria y procesador sometido a un flujo de tráfico de 1000 mbps determinó que no existe mucha diferencia ni sobrepasa el umbral básico de una computadora básica de trabajo por lo que ambos IDS están en capacidad de detectar los principales ataques reportados en una infraestructura, sin embargo Snort fue el que más se demoró en su instalación.

Palabras clave: IDS, suricata, snort, seguridad, pymes

Abstract

El avance de las tecnologías de la información en la red de las computadoras, el de los medios de comunicación para los objetivos de una empresa y el mismo tiempo en los objetivos de los ataques informáticos, por ejemplo, los equipos de borde o acceso a Internet que se han Limitado a cerrar o encubrir puertos de comunicación. De ahí la necesidad de un equipo que alerte y permita que se tomen las medidas de prevención de respuesta, como los sistemas de detección de intrusos (IDS), que son una capacidad adicional para la seguridad de los ataques y las respuestas informados de las firmas antivirus. . El objetivo de este artículo es analizar la eficiencia de dos IDS, Snort y Suricata, dentro de una infraestructura virtualizada con una configuración que organiza el tráfico de paquetes para su mejor análisis y dispositivos de almacenamiento que minimicen la latencia de escritura y lectura de datos, de manera que se puede determinar que IDS en condiciones de altas cargas de trabajo es más eficiente y así como una empresa puede tomar la decisión más compatible con sus objetivos estratégicos. Como resultado, se concluyó que el redireccionamiento del tráfico se reanudó el trabajo IDS con la ayuda de la segmentación de redes, además de la utilización de un disco sólido que elimina la alta latencia de lectura y escritura y optimiza el tiempo de análisis de la detección, por otra parte el tiempo de detección, consumo de memoria y procesador a un flujo de tráfico de 1000 mbps determinó que no existe mucha diferencia ni sobrepasa el umbral básico de una computadora básica de trabajo para ambos IDS están en la capacidad de los principales Informes de ataques en una infraestructura, sin embargo, se produjo en su instalación.

Key words: IDS, suricata, snort, security, SMEs

[1] Maestrante de Auditoria de Tecnologías de Información, Universidad Espíritu Santo – Ecuador. E-mail alzambranob@uees.edu.ec.
[2] Maestrante de Auditoria de Tecnologías de Información, Universidad Espíritu Santo – Ecuador. E-mail fguailacela@uees.edu.ec.

INTRODUCCIÓN

La revolución de las tecnologías de información y las comunicaciones (TIC) se han expandido por todo el mundo cambiando la operación y conducción de los negocios y políticas de gobiernos (Nye Jr, 2010), esta misma revolución inicio la disminución de costos en el procesamiento, transmisión y búsqueda de información (Nye, 2014). Una de esas tecnologías es el Internet que desde 1990 ha tenido un ritmo de crecimiento acelerado, contribuyendo con el desarrollo humano, económico y político en Latinoamérica (Salzman & Albarran, 2011). De hecho, el efecto de esta revolución de la información resalta el avance de las tecnologías, las comunicaciones e innovaciones vinculadas con la organización y la gestión, convirtiendo a la información en un recurso estratégico para las empresas, valioso e influyente (Arquilla & Ronfeldt, 1993)

Y es precisamente el uso de tecnologías como el Internet, lo que ha brindado apoyo a las pequeñas y medianas empresas (PYMES) para extender su participación en los mercados internacionales (Kula & Tatoglu, 2003), demostrando que las que usan más el Internet en su modelo de negocio son las que tienen un alto y sostenido crecimiento (Amorós, Planellas, & Foguet, 2010). Es así que, según estudios de De Berranger, Tucker, and Jones (2001) destacan los principales beneficios que aportan, como: el acceso a mercados locales e internacionales, venta directa, imagen que proyecta la empresa, la capacidad de aprendizaje e aumento de ingresos financieros. Así mismo, en Daniel and Grimshaw (2002) se demostró que el comercio electrónico ofrece una mejor atención al cliente, comunicación y relación con competidores y proveedores, convirtiéndose también, junto con las tecnologías relacionadas a la comunicación, en uno de los factores de adopción del Internet para las PYMES y como un factor de preocupación para esta adopción la privacidad y seguridad (Dholakia & Kshetri, 2004)

.

Por otro lado, el fácil acceso a estas tecnologías combinado con la falta de control y riesgo han confabulado para que el Internet sea también un medio de ataque para el ciberterrorismo y la ciberdelincuencia, a tal punto de llegar a pensar que la mejor opción de defensa es tener apagado el computador (Sánchez Medero, 2012) y es por esta razón que el Internet también se ha convertido en un espacio de acción para actividades de las organizaciones terroristas y delictivas (Medero, 2018), por ello gracias a este nuevo panorama no solo estados y organizaciones, sino toda persona con un computador se convierte en un enemigo potencial(de Assis, 2017).

En consecuencia MIcro (2015) informó que el 21% de operadores de infraestructura crítica reportan o conversan con sus gobiernos sobre temas de seguridad cibernética, del mismo modo Americanos (2016) reveló que solamente 6 países: Argentina, Brasil, Chile, Colombia, México y Uruguay tenían un nivel intermedio de preparación para estos mismos temas. En Ecuador, sin embargo, a pesar de no estar en el listado mencionado, es consciente de la infraestructura técnica de sostenibilidad del Internet, formada por protocolos, estándares, elementos activos de conectividad y cables, además de la seguridad que la envuelve (Delgado, 2014)

También, una de las empresas más grandes en telecomunicaciones y seguridad informa que más de la mitad de los ataques, es decir 53% costaron más de USD 500.000 en pérdida de ingresos, clientes y oportunidades, con amenazas que evolucionan en el tiempo, como malware, botnets, ataques distribuidos de denegación de servicio (DDOS) y exploits (CISCO, 2018). Amenazas que van en aumento cada año como 92% de nuevas variantes de malware, 46% solo de ransomware, 8500% de aumento en minería de criptomonedas, 55% de aumento de spam y 600% de ataques que utilizan Internet de las cosas (IoT) relacionados con botnets y DDOS, solo en el 2017 (SYMANTEC, 2018). Tanto así es el impacto que Forum (2018) ubica a los ciberataques en la posición número 6 de zona de impacto en el gráfico de Gartner, detrás de crisis del agua, cambio climático, desastres naturales, acontecimientos climáticos y armas de destrucción masiva.

Con respecto a Ecuador Telecomunicaciones (2017) asevera que frente al tema del compromiso de los Estados en seguridad informática se encuentra en el sexto puesto de 19 países de América Latina: Uruguay, Brasil, Colombia, Panamá Argentina, Ecuador, Perú, Venezuela, Chile, Costa Rica, Paraguay, El Salvador, República Dominicana, Nicaragua, Bolivia, Guatemala, Cuba y Honduras. Así mismo, Ecuador junto a Venezuela tienen el mayor índice de infecciones en ransomware 22%, en lo que respecta a botnet Perú, México, Ecuador y Colombia tiene un 85% del total de detecciones (ESET, 2018), pero no solo la intrusión es el problema sino también la extrusión por lo que DIGIWARE (2015) indica que Ecuador es el principal país donde se originan ataques de SQL-Injection de Latinoamérica

Es por esto que, como un aspecto de postura defensiva la identificación como tal en la defensa en profundidad empieza con la elaboración de

políticas de seguridad orientadas a la misión de la empresa para así asegurar y configurar adecuadamente los firewall e IDS (McHugh, Christie, & Allen, 2000) Por lo tanto, los sistemas de detección de intrusos (IDS) han ganado popularidad, puesto que fortalecen la seguridad del departamento de tecnología (TI) en sus tres componentes principales de seguridad: prevención, detección y respuestas. Y aunque en las empresas se hace énfasis en la prevención se han dado cuenta que la detección y respuesta son necesarias siempre y cuando se base en la configuración adecuada del IDS (Cavusoglu, Mishra, & Raghunathan, 2005)

El objetivo de la investigación en la que se basa este documento es analizar la eficiencia en la detección de los principales ataques informáticos del Ecuador como botnet, SQL injection y DDoS, utilizando las dos principales implementaciones de IDS: Snort y Suricata, dadas su gran popularidad y su característica de *open source*, que ahorra costos frente al hardware o software de IDS propietarios, todo esto con base en la literatura revisada en Alhomoud, Munir, Disso, Awan, and Al-Dhelaan (2011); Day and Burns (2011), Albin (2011); White, Fitzsimmons, Matthews, and Coulter (2012); Thongkanchorn, Ngamsuriyaroj, and Visoottiviseth (2013); Ridho (2014); Brumen and Legvart (2016); Shah and Issac (2018); Hu, Asghar, and Brownlee (2018); Lukaseder, Fiedler, and Kargl (2018) y la virtualización como escenario de infraestructura de red, procesadores multihilos, discos sólidos y memoria de última generación, para que así, se pueda seleccionar la mejor herramienta en una pyme y se pueda ayudar al departamento de tecnología en la ardua labor de prevenir, monitorear y detectar estos ataques con respuestas eficientes.

MARCO TEÓRICO

IDS

Los IDS están diseñados para proteger la disponibilidad, confidencialidad e integridad de la información en ambientes críticos desde 1980 con el objetivo de aumentar la perspectiva de la seguridad informática (Bass, 2000). Para Bace and Mell (2001) la detección de intrusos es una técnica que se basa en el monitoreo y análisis de cualquier evento en un sistema de red o computadoras para detectar señales de intrusiones con el objetivo de proteger la información. Estas intrusiones son causadas por agentes externos, es decir, que provienen de Internet, e internos, es decir, usuarios locales y comunes que quieren accesos privilegiados, por lo que, los IDS son las herramientas de hardware y software que automatizan este monitoreo y análisis.

Un IDS como herramienta de software sirve para detectar en todo tipo de tráfico de red, ingresos a un computador o un acceso indebido no autorizado, es un ente de monitoreo dinámico que se complementa con el monitoreo estático del firewall. Para llevar a cabo su tarea trabaja en modo promiscuo recolectando paquetes en busca de infracciones en reglas utilizando algoritmos de reconocimiento de patrones de ataque, para una vez detectado el ataque, enviar una alerta al administrador (Patcha & Park, 2007). El objetivo de un IDS es detectar las intrusiones; aunque no las detenga, avisará a un administrador inteligente para tomar la respuesta adecuada y corregir el ataque (Mukhopadhyay, Chakraborty, & Chakrabarti, 2011)

Arquitectura de un IDS

Según Garcia-Teodoro, Diaz-Verdejo, Maciá-Fernández, and Vázquez (2009) el grupo de trabajo de detección de intrusiones (GTDI) creado por la agencia de investigación avanzada de proyectos (DARPA) definió una arquitectura general de IDS basado en cuatro módulos: cajas de eventos (E), que están compuestos por sensores que monitorean el sistema; cajas de base de datos (D), donde se almacena la información obtenida de las cajas de eventos; caja de análisis (A) donde se analizan los eventos y se detectan posibles anomalías y se generará una posible alarma, si es necesario; cajas de respuesta (R) donde se ejecutan las respuestas a las amenazas detectadas; tal cual se evidencia en la figura 1

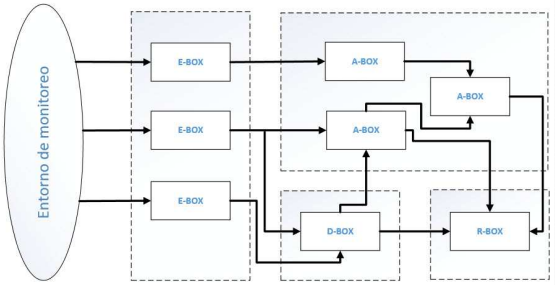

Figura 1: Arquitectura general para sistemas IDS.
Fuente: Elaboración propia adaptada a (Garcia-Teodoro et al., 2009)

Taxonomía de IDS

Aunque se han propuesto varias clasificaciones, no existe una taxonomía universal aceptada, la más profunda se puede leer en Alessandri et al. (2001) como lo muestra la figura 2

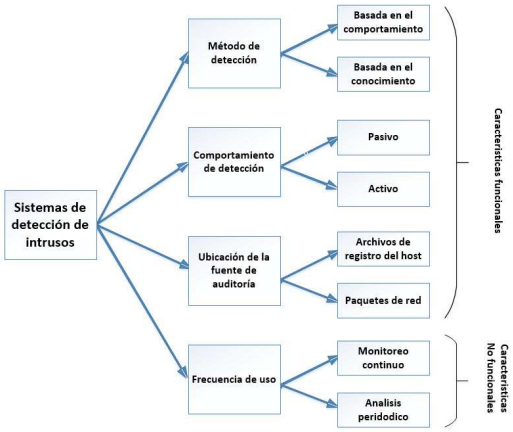

Figura 2: Características de IDS
Fuente: Elaboración propia adaptada a (Alessandri et al., 2001)

Sin embargo, para la evaluación propuesta en este documento, se tomará en consideración la característica de la ubicación en la fuente de auditoria.

Los primeros IDS se basaban en host (HIDS) debido a que las computadoras eran centralizadas como los mainframes y todos los usuarios eran locales por lo que la detección se enfocaba solo en amenazas internas, en ausencia de la interactividad con el exterior (Lunt et al., 1988), un ejemplo en la actualidad de estos HIDS es Ossec (2018) un HIDS que monitorea toda la actividad de un sistema como integridad de archivos, registros, directorios y procesos además de ser código abierto y gratuito; pero con la llegada de las redes de computadoras, el enfoque cambió y se trabajó en la conexión entre los IDS basados en host dando lugar a los IDS basados en red (NIDS) (Anderson, Frivold, & Valdes, 1995), un ejemplo en la actualidad es Snort y Suricata, los cuales se detallarán en el marco teórico del documento. Luego en 1990 se debatió entre las ventajas de los IDS basados en red sobre los IDS basados en host, por lo que se proporcionó una solución integrada de ambas tecnologías llamada IDS híbridos como por ejemplo VMFence (Jin et al., 2009) que es un sistema personalizado de prevención de intrusiones basado en máquinas virtuales para la prevención de ataques maliciosos en entornos informáticos virtuales.

En otras palabras, hay dos tipos de IDS los basados en host (HIDS) que poseen sensores solo para el host donde residen y recopilan información en sistemas individuales analizando actividades en base a registros y auditorias del sistema operativo y así poder determinar qué procesos o usuarios forman parte de un ataque en el sistema residente, también están los IDS basados en red NIDS que poseen sensores basados solo en red, estos trabajan capturando y analizando paquetes de red que afectan a todos los host de una red de datos para después reportar el ataque a una consola de administración central (De Berranger et al., 2001), (Bace & Mell, 2001).

NIDS es el sistema más completo y por estar diseñado para redes, contiene información no solo sobre todos los hosts de la red, sino también del resto de equipos como enrutadores y los correspondientes resultados de monitoreo (Sun, Osborne, Xiao, & Guizani, 2007), por eso es el eje de la investigación. Además según Gascon, Orfila, and Blasco (2011) contribuyen en las políticas de seguridad de la organización y protegen sus activos de los ataques de red.

Técnicas de detección

Los dos planteamientos principales de análisis de ataque para los IDS son: la detección de mal uso y la detección de anomalías (Zhang, Lee, & Huang, 2003). La detección de mal uso es el método más utilizado y eficaz para los ataques conocidos con una tasa de falsos positivos baja, sin embargo no detecta ataques nuevos (Kabiri & Ghorbani, 2005). Esta técnica se basa en firmas con patrones de amenazas conocidas que se comparan con paquetes de red o entradas de registro para identificar la posible intrusión. En cambio, la detección de anomalías compara definiciones de un tarea normal y si se encuentra una desviación significativa en su comportamiento se identificará como posible intrusión (Scarfone & Mell, 2007), en esta técnica se utilizan varias tecnologías como detección de anomalías estadísticas, aprendizaje automático basado en detección de anomalías y detección de anomalías basada en minería de datos, que se pueden revisar de manera general en Patcha and Park (2007)

PROBLEMAS EN LA DETECCIÓN DE INTRUSIÓN

Entre los problemas más relevantes para el objetivo del articulo está la eficacia, que se refiere a la capacidad en detectar las intrusiones, en qué medida las detecta y que tanto rechaza por falsos positivos o falsas alarmas, y la eficiencia que se refiere al tiempo de ejecución del IDS, recursos que consume, análisis en tiempo real, entre otros (Axelsson, 2000). Los falsos positivos se producen cuando se ejecuta una alerta sin existir una actividad maliciosa en el tráfico, causando un esfuerzo innecesario a los expertos en seguridad y la preocupación de que el tráfico benigno este bloqueado mientras que los falsos negativos no generan ninguna alerta cuando si se trata de tráfico malicioso lo que resulta peligroso ya que pone en riesgo al sistema y redes informáticas (Ho, Lai, Chen, Wang, & Tai, 2012)

Métricas de desempeño

Según Porras and Valdes (1998) y Alessandri et al. (2001) existen algunas características deseadas para un IDS como el rendimiento de predicción, que se refiere a si la predicción es o no es la adecuada por lo que debe satisfacer dos condiciones: identificar debidamente las intrusiones y no confundir las legítimas como intrusión, y la otra característica que es el tiempo de rendimiento, que no es otra cosa que el tiempo que se demora el IDS en detectar una intrusión dependiendo del tiempo de procesamiento y propagación, que es el tiempo que se demora en llegar la alerta al analista de seguridad.

Considerando la literatura revisada con el objetivo en la introducción del documento las métricas a considerar son son rendimiento del procesado y memoria con alto tráfico de red

Snort

Snort se concibió para ser un sistema ligero y flexible capaz de ser utilizado en redes pequeñas y grandes con todas las capacidades de un IDS, por lo que se ha vuelto la alternativa perfecta a los IDS comerciales (Roesch, 1999), es de código abierto y fue creado por Martin Roesch en 1998 (Lippmann et al., 2000). Además, se basa en un modelo de firmas que contienen patrones de ataques conocidos que se comparan con los patrones de los paquetes de red para identificar intrusiones (Singh, Patel, Borisaniya, & Modi, 2016)

Arquitectura de snort

Snort permite la implementación de preprocesadores que permite a los usuarios y desarrolladores agregar reglas adicionales y así ampliar la funcionalidad del IDS (snort-org, 2019), además, estos preprocesadores permiten la decodificación de datos y detección de anomalías, funciones más complicadas que la captura de paquetes y su respectiva decodificación (Ghafir, Prenosil, Svoboda, & Hammoudeh, 2016), tal como se muestra en su arquitectura, en la figura 3

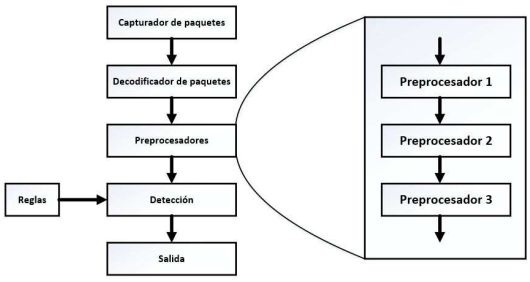

Figura 3: Arquitectura de Snort
Fuente: Elaboración propia adaptada a (Ghafir et al., 2016)

Suricata

Fue concebido con la premisa de construir una alternativa a snort por la Open Information Security Foundation OISF en el 2009, esto con el apoyo del Departamento de Seguridad Nacional de E.U.A y la empresa privada y, aunque su código es original, no se puede disimular las similitudes con snort, a tal punto que las mismas reglas son utilizadas por los dos (White et al., 2012). Un ejemplo de la estructura de estas reglas se puede observar en la figura 4

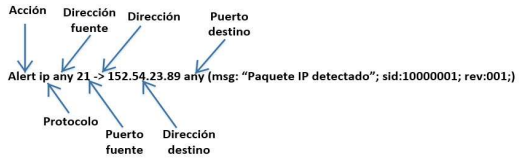

Figura 4: Ejemplo de regla para detección de ataques
Fuente: Elaboración propia adaptada a (Jinsheng Xu & Yu, 2011)

Arquitectura de Suricata

La arquitectura de Suricata es similar a la de Snort con la diferencia de que la característica del preprocesador que se define en snort está definida en dos partes: decodificación que agrega información a la representación de los paquetes internamente y la detección que se basa en esa representación para adicionar sentencias claves en la ejecución de las reglas (Ghafir et al., 2016), tal como se puede observar en su arquitectura, en la figura 5

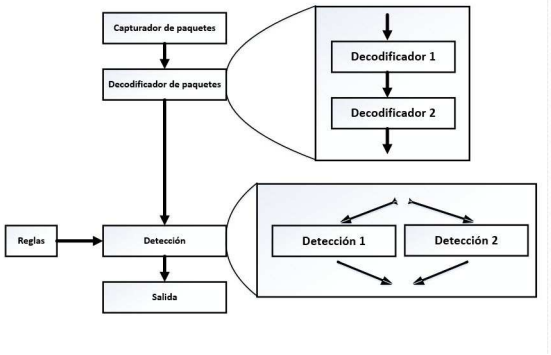

Figura 5: Arquitectura de Suricata
Fuente: Elaboración propia adaptada a (Ghafir et al., 2016)

FALLAS DE SEGURIDAD

Los tres tipos fallas que derivan en una violación de seguridad son: ataque.- que es cuando el atacante intenta violar la seguridad con un intento de intrusión; vulnerabilidad.- que son errores en desarrollo de aplicaciones u configuraciones fácilmente explotables para realizar una intrusión, e intrusión.- que es una falla incitada o ataque desde el exterior con el objetivo de explotar una vulnerabilidad (Powell & Stroud, 2001)

Ataques

Para Bace and Mell (2001), Lazarevic, Kumar, and Srivastava (2005) y Hoque, Mukit, Bikas, and Naser (2012) los principales ataques en las redes de información, y principales retos para los IDS

son: los ataques de denegación de servicios, escaneo de puertos y penetración. Todos estos ataques se relacionan con los descritos en los reportes ya mencionados en la introducción, por ende Behal and Kumar (2011) afirman que en los últimos años todos los ataques, escaneos y penetraciones en redes y sistemas tienen como fuente principal un malware , estos se aprovechan de vulnerabilidades, puertas traseras o errores de software para instalarse con o sin consentimiento del usuario utilizando varias técnicas de escaneo (Freiling, Holz, & Wicherski, 2005)

Malware

Es un programa o código que se define como malicioso, como por ejemplo los virus, troyanos y gusanos (Christodorescu, Jha, Seshia, Song, & Bryant, 2005) y pueden ocasionar un caos terrible en cualquier sistema informático Gordon, Loeb, Lucyshyn, and Richardson (2005)

Ataques de DoS

Es un ataque creado para que un computador o red no pueda prestar sus servicios debido a que esta intencionalmente bloqueado por alguna actividad maliciosa (Douligeris & Mitrokotsa, 2004)

Botnet

Es una colección de computadoras conectadas a Internet y secuestradas para ser controladas de manera remota con fines maliciosos, como generación de malware, vulneración de contraseñas o denegación de servicios (Chanthakoummane, Saiyod, & Khamphakdee, 2015)

Inyección de lenguaje de consulta estructurado (SQL)

Es una inserción de código SQL en forma de consulta por medio de una aplicación o interfaz web con el objetivos de leer, alterar o borrar datos confidenciales de un base de datos (owasp, 2016).

Herramientas de pentesting

Kali Linux

Kali Linux es una distribución Linux basada en Debian, financiada y mantenida por Offensive Security, que se utiliza en auditorias y pruebas de penetración, consta de una variedad de herramientas orientadas a la seguridad, informática forense e ingeniería inversa (Security, 2019)

Metasploit

Es un proyecto de código abierto que se basa en una plataforma que proporciona una infraestructura de herramientas utilizadas para realizar pruebas de penetración y auditorias de seguridad con el fin de buscar y explotar vulnerabilidades (Rapid7, 2019)

Wireshark

Es un analizador de protocolos que permite analizar la red en modo promiscuo en línea para luego ser analizada fuera de línea, es multiplataforma y utiliza una interfaz gráfica como también una utilidad de comandos llamada tshark (Combs, 2019)

Tcpreplay

Es una herramienta de código abierto y gratuita para reproducir tráfico de red guardado o capturado en un archivo de datos y enviarlo a través de una red de datos, se lo utiliza para evaluar IDS, reproduciendo tráfico hacia estos sistemas (appneta, 2019)

Virtualización

La virtualización es un concepto que se inicia en la década de 1960 y se basa en la creación de instancias lógicas o virtuales del hardware de un computador, estos sistemas son gobernados por un hipervisor que es un software que permite compartir a las máquinas virtuales recursos, como procesadores, memoria, disco duro, red etc. del sistema anfitrión. Además de proporcionar un aprovisionamiento elástico de recursos que permite adaptar el sistema a las variaciones de la carga de trabajo por lo que ha generado gran interés, tanto en la industria, como en la academia (Aleksandar Milenkoski, 2015).

Este concepto ha permitido abordar problemas de confiabilidad, seguridad, costo y complejidad de la red informática, y gracias a la utilización de técnicas de computación autónoma, se ha garantizado los requisitos de equilibrio de carga, el intercambio de recursos y la asignación de tareas entre máquinas virtuales (Xu, Zhao, Fortes, Carpenter, & Yousif, 2007). Por otra parte, esta tecnología es utilizada ampliamente en la nube, lo que cada día se hace más importante Liao, Lin, Lin, and Tung (2013)

TRABAJOS PREVIOS

Hall and Wiley (2002) propone una metodología para una evaluación estandarizada de la capacidad del NIDS, después Mutz, Vigna, and Kemmerer (2003) propone un nuevo enfoque de prueba y una herramienta que genera tráfico de red utilizando firmas de Snort para la evaluación. Luego Antonatos, Anagnostakis, and Markatos (2004) estudia cómo generar cargas de trabajo realistas para la evaluación del desempeño en los IDS; posteriormente Paulauskas and Skudutis (2008) hacen un estudio de rendimiento para el IDS Snort, y Akhlaq et al. (2009) hace un estudio de la evaluación de rendimiento de Snort en la virtualización.

A continuación en Alhomoud et al. (2011) evalúa la eficiencia de los IDS Snort y Suricata en redes de alta velocidad para que después Day and Burns (2011) hicieran lo mismo poniendo más énfasis en la precisión y consumo de recurso de las dos implementaciones IDS Snort y Suricata. Akhlaq et al. (2011) vuelven a medir un IDS en redes de alta velocidad y el consumo de recursos, esta vez solo para Snort, luego Albin (2011) hace un estudio exhaustivo de Snort y Suricata evaluando su precisión y rendimiento, después White et al. (2012) presentan una evaluación cuantitativa en torno al procesamiento de Snort y Suricata

Además, Thongkanchorn et al. (2013) evalúan de nuevo la precisión y rendimiento de los IDS Snort y Suricata como también Bro. En seguida Wang, Kordas, Hu, Gaedke, and Smith (2013) proponen una metodología para evaluar el rendimiento imparcialmente a los IDS tomando en cuenta a los IDS Snort, Ourmon y Samhain, luego Bulajoul, James, and Pannu (2013) ponen a prueba a Snort en la velocidad de la red y rendimiento.

A continuación Kalam, Rab, and Deswarte (2014) proponen un enfoque multi-modelo para la clasificación y generación de los ataques creando una herramienta para la evaluación de los IDS y probándola con Snort y Bro, así también, Ridho (2014) hacen un estudio de las ventajas y desventajas de estos mismos IDS, Snort, Suricata y Bro. Alqahtani and John (2016) con base en un análisis de eficacia entre Snort y Suricata proponen un mejor enfoque para la seguridad en la nube y su disponibilidad, además A. Milenkoski, Jayaram, Antunes, Vieira, and Kounev (2016) no solo proponen un nuevo enfoque de evaluación de IDS sino también una nueva métrica de evaluación para la precisión y rendimiento de un IDS en un entorno virtualizado usando Snort y en seguida Brumen and Legvart (2016) ofrecen un análisis exhaustivo de Snort y Suricata en seguridad y rendimiento utilizando varios sistemas operativos.

Así pues Karim, Vien, Le, and Mapp (2017) aporta con una investigación de IDS en diversos escenarios de una red, considerando parámetros como velocidad, tráfico y tamaño de paquetes utilizando Snort, también (Saber, Belkasmi, Chadli, Emharraf y Farissi, 2017) evalúan la eficacia y rendimiento de Snort en redes de alta velocidad, a continuación Shah and Issac (2018) realizan un estudio del rendimiento y precisión de Snort y Suricata en redes informáticas, también Hu et al. (2018) evalúan el rendimiento de Snort, Suricata y Bro en redes de alta velocidad y por ultimo Lukaseder et al. (2018) propone un nuevo enfoque de evaluación de rendimiento para los IDS Snort y Suricata

METODOLOGIA

Se eligieron los IDS más populares, mencionados en la literatura de trabajos previos, a saber: Snort en su versión 2.9.12 y Suricata, en su versión 3.1, luego se construyó y virtualizó una infraestructura acorde a una mediana empresa. Después se configuró y se inyectó tráfico de red para intentar simular el entorno apropiado y realizar los ataques planificados y así recopilar la información necesaria para el análisis y conclusión. La hoja de ruta del proceso de investigación se aprecia en la figura 5.

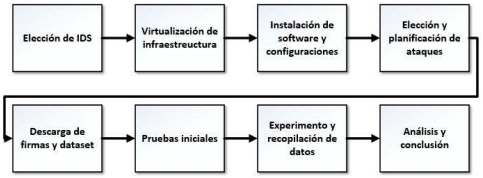

Figura 5: Metodología del experimento
Fuente: Elaboración propia

Virtualización de la infraestructura

La red virtualizada consta de un switch de capa 3 y un firewall. En el equipo perimetral se ejecutan configuraciones para la segregación de red llamadas redes de área local virtual (VLAN'), para esta simulación se crearon 6 VLAN's de acceso en el equipo de comunicación, o switch, las VLAN de acceso y troncales son asignadas en los puertos de comunicación como se muestra en la figura 6. Las troncales permiten el paso de todas las VLAN creadas y las de acceso solo permiten el paso de la VLAN asignada en el puerto, y cada VLAN consta de equipos que prestan servicios como se aprecia en la Tabla 1.

Además, se ha configurado un puerto SPAN en el puerto de una interfaz del switch para copiar el tráfico de una interfaz específica y redirigirla a otra interfaz donde capta o replica todo el tráfico y así poder utilizar alguna herramienta de análisis de red como tcpdump o wireshark. En el caso del experimento a desarrollar se creó una sesión de monitoreo donde todas las interfaces de servicios son fuentes y la interfaz destino es la del IDS a evaluar.

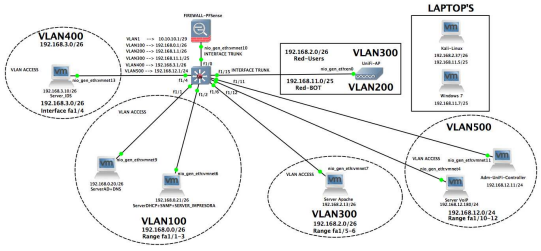

Figura 6: Diseño de red para escenario del experimento
Fuente: Elaboración propia

Instalación del software y configuraciones

En cada VLAN se instalaron equipos para simular un entorno empresarial. En la VLAN 100 se virtualizó 2 Equipos Windows server R2 con un

servicio de directorio activo y otro de DHCP respectivamente, en la VLAN 200, un Kali Linux para los ataques, en la VLAN 300 un servidor web Apache en un Debian 8, y para la simulación de la *botnet* un Kali Linux con un servicio Ares, en la VLAN 400 se instalaron los IDS Snort y Suricata, ambos en CentOS 7 Linux y en la VLAN 500 un servidor de Telefonía IP Elastix como se detalla en la Tabla 1.

***Tabla 1*:** Especificaciones de la red en el escenario del experimento

Configuraciones						
hardware			Sistema operativo	Servicio	Network	VLAN
CPU	Memoria RAM (Gigas)	Disco duro VHD (Gigas)				
Intel core i7-2640M 3 core por 2,793 GHZ	12	250	Vmware ESXi 5,50		DHCP	
1 Core	1	15	Free BSD AMD 64 bits	Firewall PfSense v. 2.44 release p2	WAN (DHCP), LAN 10.10.10.1/29, VLANSERVERS: 192.168.0.1/26, VLANDMZ: 192.168.1.1/26, VLANUSERS: 192.168.2.1/26, VLANIDS: 192.168.3.1/26, VLANVOIP: 192.168.12.254/24	
1 Core	2	25	MS Windows Server 2012 R2 64 bits	Directorio Activo + DNS	192.168.0.20/26	100
1 Core	1	5	MS Windows Server 2012 R2 64 bits	DHCP	192.168.0.21/26	100
2 Core	2	40	Linux debian 8 64 Bits	Apache	192.168.2.13/26	300
2 Core	1	20	Centos release 7.5 64 Bits	IDS Suricata 3,1	192.168.3.10/26	400
2 Core	1	20	Centos release 7.5 64 Bits	IDS Snort 2.9	192.168.3.10/26	400
1 Core	2	30	MS Windows 10 Pro	Administración de elastix	192.168.12.11/24	500
1 Core	2	30	Kali Linux 64 Bits 2018,4	Pentesting	192.168.2.31/26	200/300
1 Core	0,125		GNS3 (Switch Cisco c3725)	Switch de Capa 3	192.168.2.31/26	

Fuente: Elaboración propia

Elección y planificación de ataques

Según los reportes actuales de firmas de antivirus y empresas relacionadas con la seguridad detalladas en la introducción del artículo, el aumento de ataques de DDOS y Malware es evidente y para ello se utilizan redes controladas por atacantes llamadas *botnet*, que son grupos de computadores controlados por un atacante que escanearon o aprovecharon una vulnerabilidad, y que por medio de un malware lograron su control con el objetivo de distribuir otros malware o realizar ataques de DDOS

Por lo tanto, se simularán ataques de escaneo de puertos con la herramienta *nmap* debido a que el escaneo es la principal técnica para la investigación y planificación de un ataque y por la que se puede obtener información valiosa como: puertos abiertos, *escucha* de servicios, o las principales vulnerabilidades de un computador o red.

También se simularán ataques de DDOS debido al impacto que este tipo de ataques suelen tener sobre los servicios, al grado que pueden dejar sin reacción a cualquier página web o servicio de red, para lo cual utilizaremos un exploit con la herramienta metasploit de la distribución Kali Linux que es usada para la realización de auditorías de seguridad. Además, se simulará una *botnet* y se creará un malware genérico para simular la captura de un computador para lo cual se utilizará la *botnet* Ares.

Por ultimo, simularemos un ataque de inyección SQL para demostrar que los ataques no deben de ser revisados como intrusión sino también como extrusión para lo cual se utilizará una página de prueba como DVWA con un servicio Apache y una herramienta de ataque como *sqlmap* incluida en la distribución Kali Linux

A continuación, se adjunta un resumen de la planificación de los ataques en la Tabla 2.

Tabla 2: Resumen de simulaciones de ataques

Ataque	Herramienta	Origen	Destino
DDOS	Metsploit	Kali Linux	Servidor Web
Botnet y Malware	Ares	Computador de usuario	Kali Linux
Inyección SQL	sqlmap	Kali Linux	Servidor Web

Fuente: Elaboración propia

Descarga de firmas y conjunto de datos

Con base en las investigaciones realizadas, se logró descargar las reglas relacionadas con los principales ataques propuestos para la evaluación, como se observa en la Tabla 3.

Tabla 3: Reglas utilizadas en los ataques

No	Ataque	Regla	Referencia
1	DDOS	alert tcp $HOME_NET any -> $HOME_NET 80 (flags: S; msg:"Possible TCP DoS"; flow: stateless; threshold: type both, track by_dst, count 70, seconds 10; sid:10000001;rev:1;)	github (2019)
2	Botnet	alert ip $HOME_NET any -> [192.168.11.5] 8080 (msg:"Ares"; sid:2404150; rev:3546;)	Francis (2017)
3	Inyección SQL	alert tcp any any -> any $HTTP_PORTS (msg:"SQL Injection - Start Attacks4.....-SQL"; flow:established,to_server; pcre:"/((\?)[^\n]*(\=)[^\n]*((\%55)|(u)|(\%75))|(\%4e)|(n)|(\%6e))|(\%69)|(i)|(\%49))|((\%6f)|(o)|(\%4f))|((\%4e)|(n)|(\%6e)))/i"; classtype: Web-application-attack; sid:9397; rev:28;)	Sayantan (2019)

Fuente: Elaboración propia

Del mismo modo se seleccionó el conjunto de datos que se utilizó para la generación de tráfico en la red de datos como lo son el conjunto de datos de evaluación de detección de intrusos en (UNB, 2019) de los cuales se utilizaron los siguientes archivos con extensión .pcap. Ver listado proporcionado en la Tabla 4.

Tabla 4: Conjunto de datos usados en los ataques

Ataque	Conjunto de datos
Escaneo de puertos	Friday-WorkingHours.pcap
DDOS	Wednesday-WorkingHours.pcap
Botnet	Friday-WorkingHours.pcap
Inyección SQL	Thursday-WorkingHours.pcap

Fuente: Elaboración propia

Pruebas iniciales

Una vez descargadas las herramientas para los ataques, reglas, conjunto de datos y los respectivos IDS se procede a probar las configuraciones y el funcionamiento de la red virtual trabajando en conjunto con toda la infraestructura armada.

Se procede probando la conexión con todos los equipos involucrados, realizando una captura de tráfico utilizando Wireshark, y vigilando el funcionamiento de los IDS, ante los ataques preparados y utilizando las reglas con el conjunto de datos descargados para la simulación de tráfico.

Experimentación y recopilación de datos

Se implementaron 3 escenarios para cada uno de los IDS evaluados: 1. Ataque de DOS, 2. Registro de *botnet* por malware y 3. Inyección de SQL. Cada uno de estos escenarios fueron sometidos a 1000 mbps de tráfico de red y se utilizaron reglas básicas de detección

Escenario 1: Ataque de DOS.

Se procedió a realizar un ataque de DOS utilizando el *exploit synflood* de la herramienta *metasploit* desde una máquina virtual con Kali Linux al servidor web, obteniendo los resultados descritos en la Tabla 5.

Tabla 5: Escenario 1: "Ataque de DOS"

IDS	TIPO ATAQUE	TIEMPO DE ALERTA [Seg]	CPU [MHz]	MEMORIA [Gigabytes]	RED [MBps]
SNORT	DOS	2,95	437,6	0,35	4,84
SURICATA	DOS	2,55	476,5	0,36	1,37

Fuente: Elaboración propia

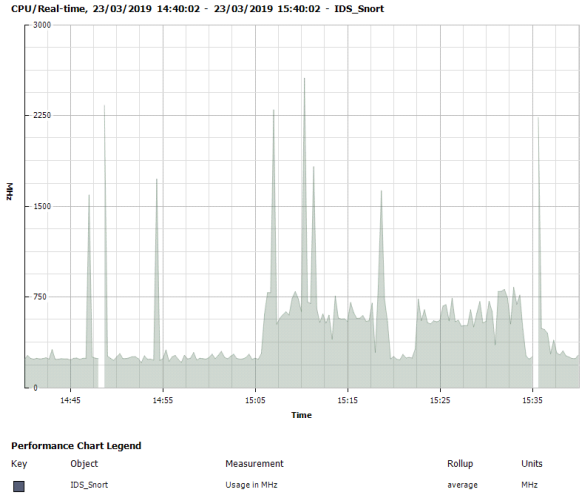

Figura 7: Medición de CPU Snort escenario 1
Fuente: Elaboración propia

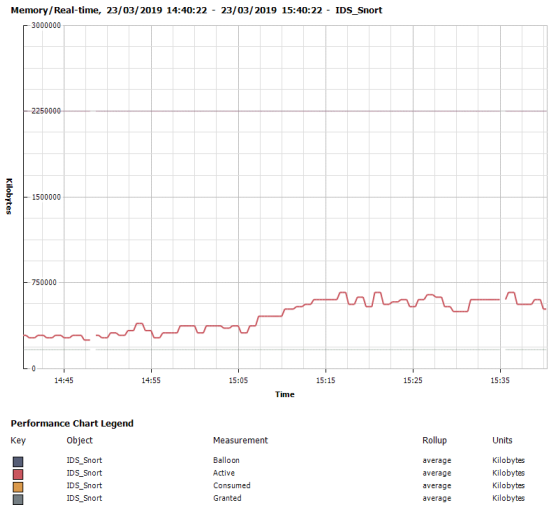

Figura 8: Medición de memoria Snort escenario 1
Fuente: Elaboración propia

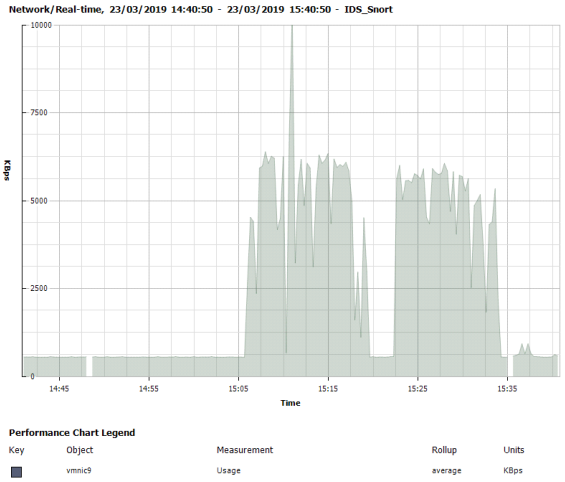

Figura 9: Medición de red Snort escenario 1
Fuente: Elaboración propia

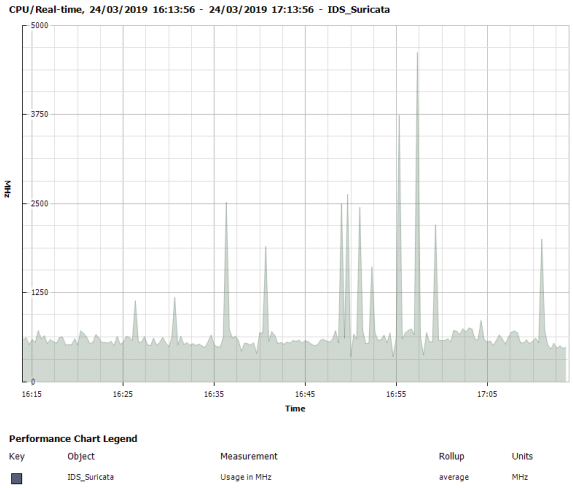

Figura 10: Medición de CPU Suricata escenario 1
Fuente: Elaboración propia

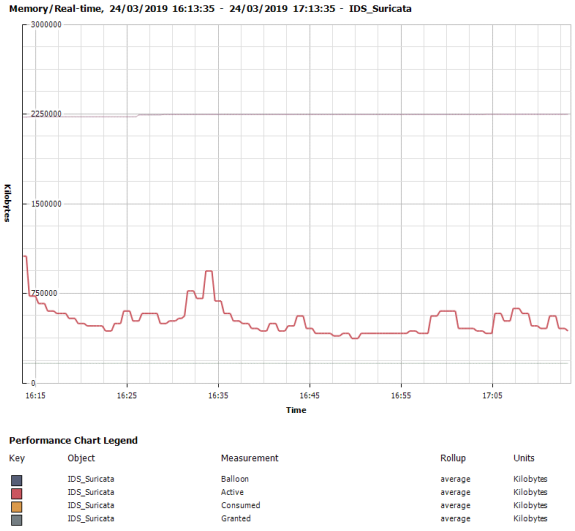

Figura 11: Medición de memoria Suricata escenario 1
Fuente: Elaboración propia

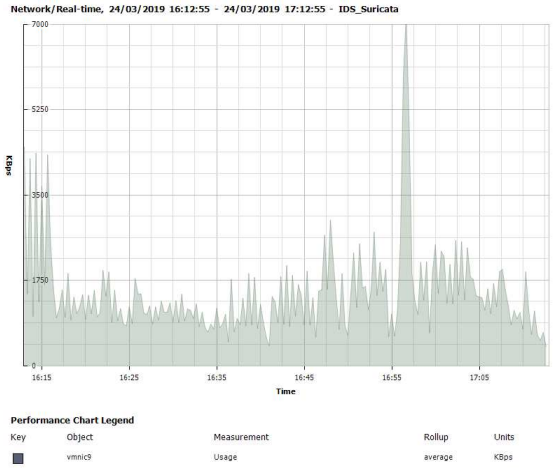

Figura 12: Medición de red Suricata escenario 1
Fuente: Elaboración propia

Escenario 2: Registro de botnet por malware.

Se procedió a instalar la *botnet* Ares en la máquina virtual de Kali Linux y a crear un archivo ejecutable para Windows utilizando la misma herramienta para ser ejecutada desde un computador virtual de un usuario dentro de la misma infraestructura virtualizada, para simular la infección de un computador por una *botnet* obteniendo los resultados provistos en la Tabla 6.

Tabla 6: Escenario 2: "Registro de botnet por malware"

IDS	TIPO ATAQUE	TIEMPO DE ALERTA [Seg]	CPU [MHz]	MEMORIA [Gigabytes]	RED [MBps]
SNORT	BOTNET	4,6	807,15	0,45	7,85
SURICATA	BOTNET	4,35	840,55	0,62	6,79

Fuente: Elaboración propia

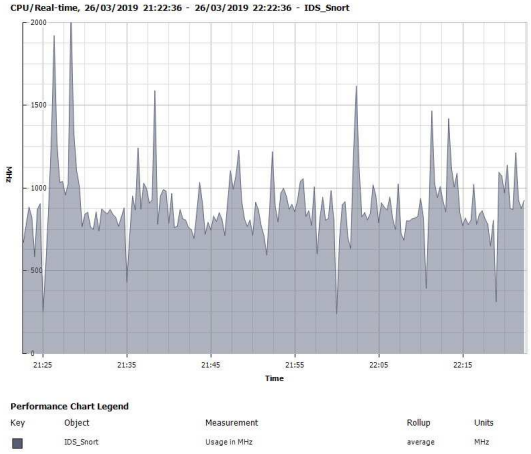

Figura 13: Medición de CPU Snort escenario 2
Fuente: Elaboración propia

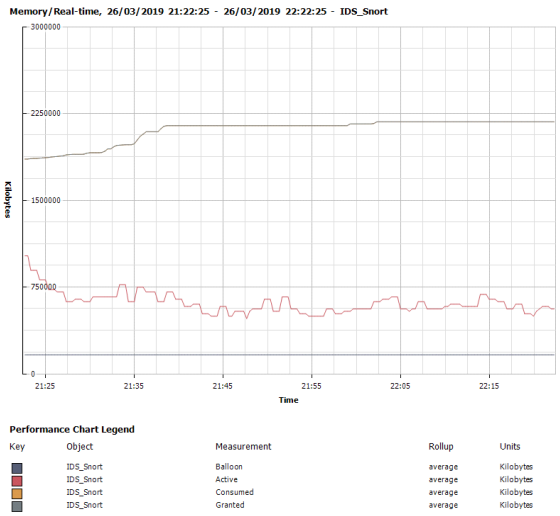

Figura 14: Medición de memoria Snort escenario 2
Fuente: Elaboración propia

Figura 15: Medición de red Snort escenario 2
Fuente: Elaboración propia

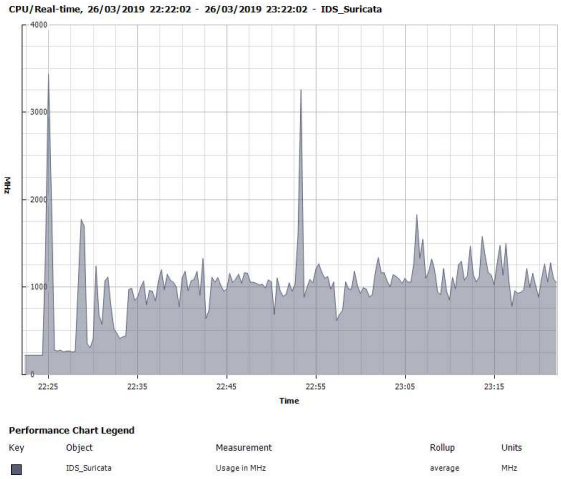

Figura 16: Medición de CPU Suricata escenario 2
Fuente: Elaboración propia

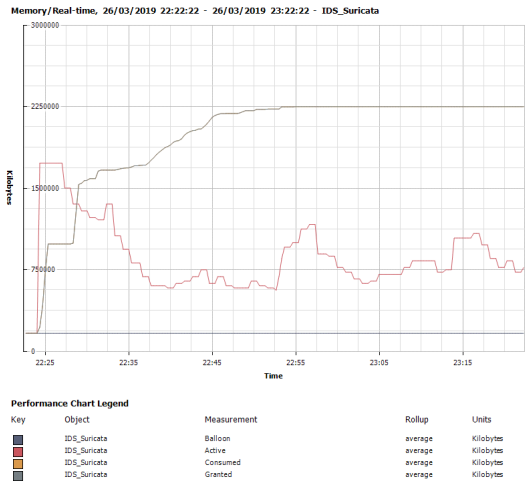

Figura 17: Medición de memoria Suricata escenario 2
Fuente: Elaboración propia

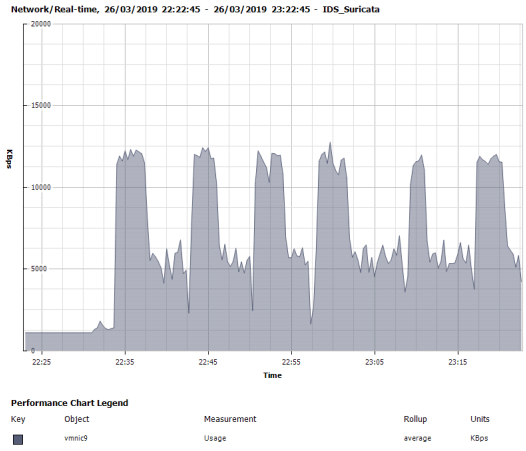

Figura 18: Medición de red Suricata escenario 2
Fuente: Elaboración propia

Escenario 3: Ataque de inyección SQL.

Se procedió a utilizar la herramienta SQLmap desde una máquina virtual con Kali Linux para atacar el servidor web de la infraestructura virtualizada obteniendo los resultados provistos en la tabla 7

Tabla 7: Escenario 3: "Ataque de inyección SQL"

IDS	TIPO ATAQUE	TIEMPO DE ALERTA [Seg]	CPU [MHz]	MEMORIA [Gigabytes]	RED [MBps]
SNORT	INYECCION SQL	34,85	679,05	0,47	9,11
SURICATA	INYECCION SQL	24,5	675,75	0,42	8,48

Fuente: Elaboración propia

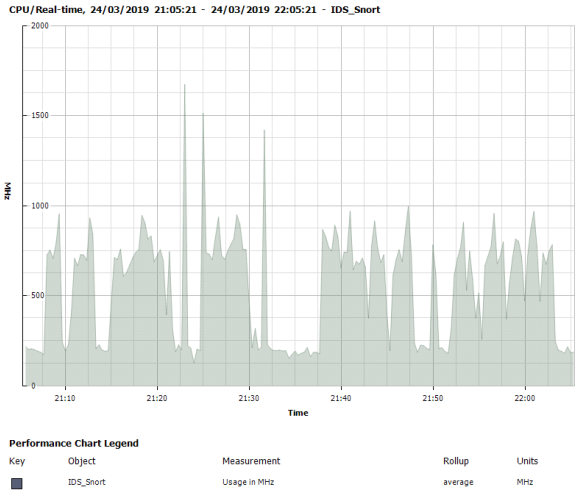

Figura 19: Medición de CPU Snort escenario 3
Fuente: Elaboración propia

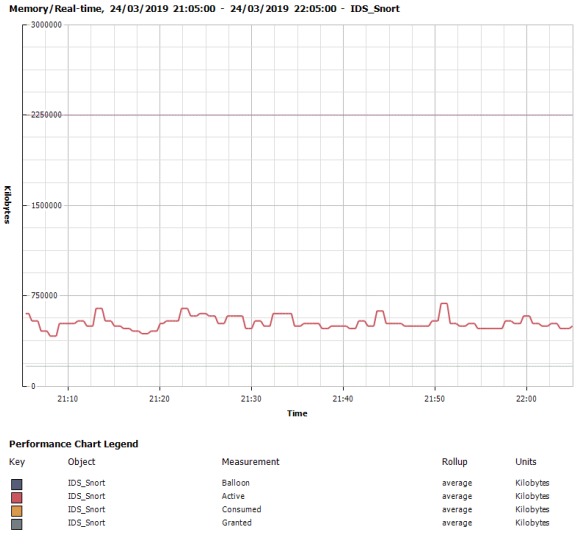

Figura 20: Medición de memoria Snort escenario 3
Fuente: Elaboración propia

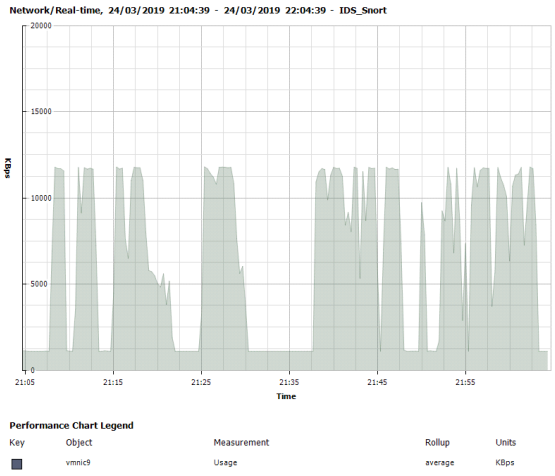

Figura 21: Medición de red Snort escenario 3
Fuente: Elaboración propia

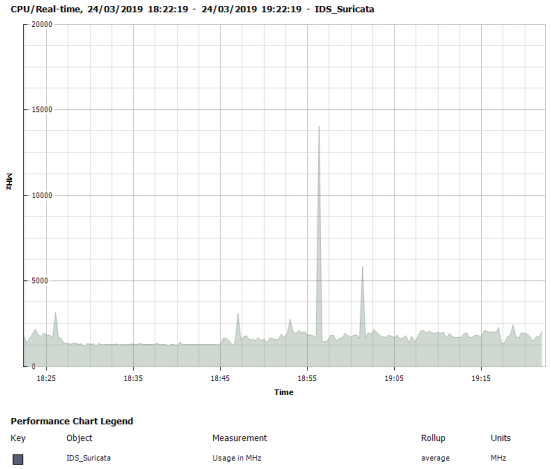

Figura 22: Medición de CPU Suricata escenario 3
Fuente: Elaboración propia

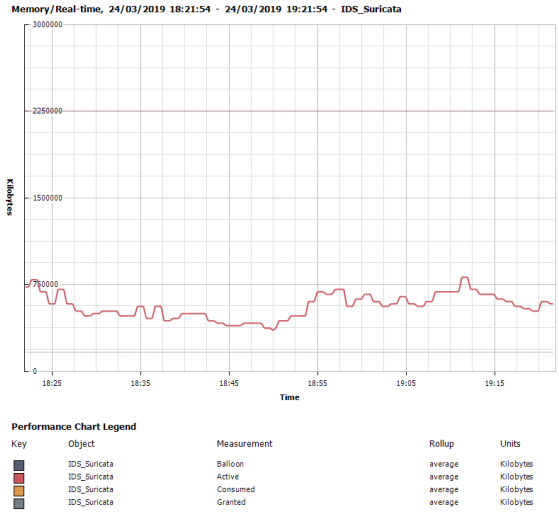

Figura 23: Medición de memoria Suricata escenario 3
Fuente: Elaboración propia

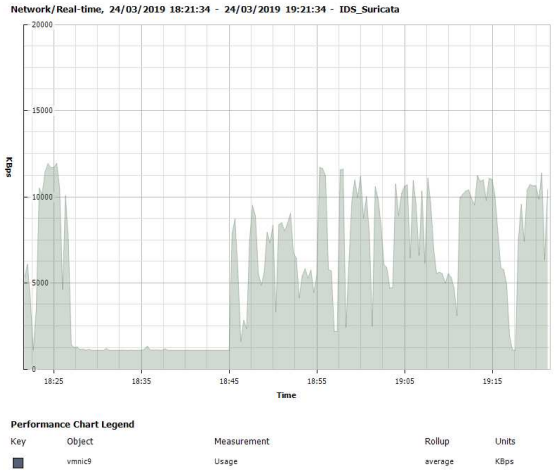

Figura 24: Medición de red Suricata escenario 3
Fuente: Elaboración propia

ANALISIS DE LOS RESULTADOS

En el Escenario 1, tabla 5, ataques de DOS el tiempo de alerta es casi igual en ambos IDS 2.95 segundos para snort y 2.55 segundos para suricata, sin embargo en el consumo de CPU suricata presenta un aumento de 476.5 MHZ en relación con los 437.6 MHZ usados por snort; en lo que respecta a memoria el consumo para ambos es casi igual 0.35 Gb para snort y 0.36 para suricata pero el consumo de paquetes de red es mayor para snort 4.84 Mbps en relación a los 1.37 Mbps de suricata

En el escenario 2, tabla 6, registro de botnet por malware, el tiempo de detección no varía mucho 4.6 segundos para snort y 4.35 segundos para suricata; en el consumo de CPU suricata vuelve a estar más alto con 840.55 MHZ para los 807.25 MHZ de snort; en memoria suricata también consume más con 0.62 Gb en relación con los 0.45 Gb de snort y en el consumo de red nuevamente snort tiene un valor alto de 7.85 Mbps en relación a los 6.79 Mbps de suricata

En el escenario 3, tabla 7, ataque de inyección SQL hay ya un aumento significativo en tiempo de alerta y es para snort 34.85 segundo en relación a los 24.5 segundos de suricata; en consumo de CPU ahora es snort que tiene el valor más alto con 679.05 MHZ en relación a los 675.75 MHZ de suricata; en memoria también snort consume más con 0.47 Gb en relación a los 0.42 Gb de suricata y por último en consumo de paquetes se mantiene con un valor mayor para snort de 9.11 Mbps en relación a los 8.44 Mbps de suricata

En todos los escenarios se puede observar la variación de consumo en el tiempo, tanto para la CPU: figuras 7, 13, 19 para snort y figuras 10, 16, 22 para suricata; memoria: figuras 8, 14, 20 para snort y figura 11, 17, 23 para suricata; Red: figura 9, 15, 21 para snort y figuras 12, 18, 24 para suricata

Tabla 8: Todos los escenarios de ataque

IDS	TIPO ATAQUE	TIEMPO DE ALERTA [Seg]	CPU [MHz]	MEMORIA [Gigabytes]	RED [MBps]
SNORT	DOS	2,95	437,6	0,35	4,84
	INYECCION SQL	34,85	679,05	0,47	9,11
	BOTNET	4,6	807,15	0,45	7,85
SURICATA	DOS	2,55	476,5	0,36	1,37
	INYECCION SQL	24,5	675,75	0,42	8,48
	BOTNET	4,35	840,55	0,62	6,79

Fuente: Elaboración propia

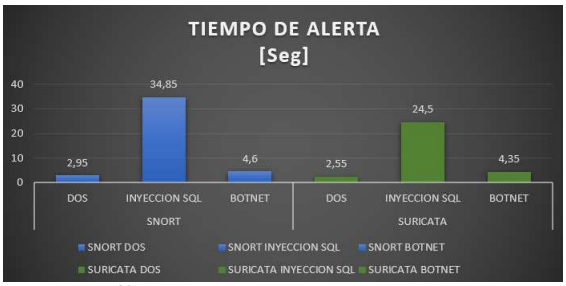

Figura 25: gráfico tiempo de alerta
Fuente: Elaboración propia

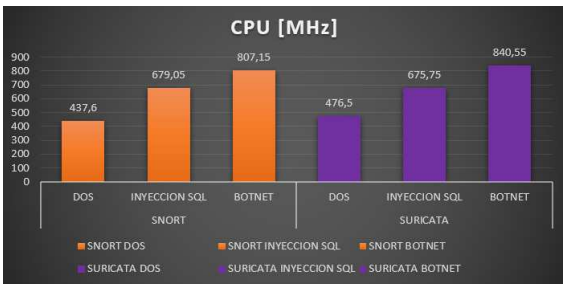

Figura 26: gráfico consumo de CPU
Fuente: Elaboración propia

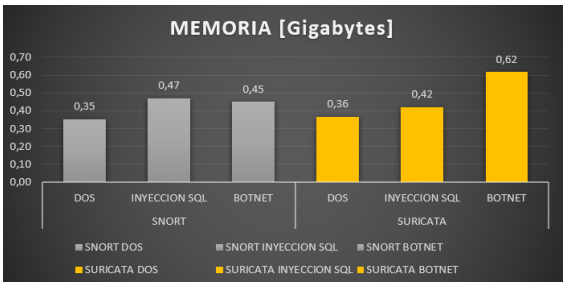

Figura 27: gráfico consumo de memoria
Fuente: Elaboración propia

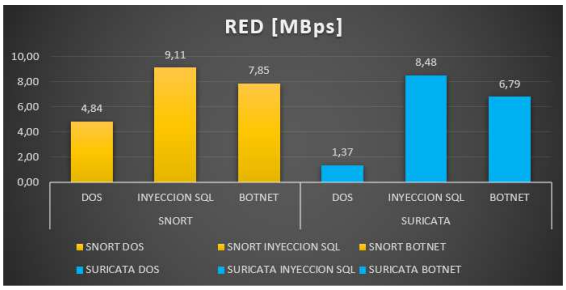

Figura 28: gráfico consumo de Red
Fuente: Elaboración propia

A continuación, utilizando la tabla 8, todos los escenarios de ataque, se puede resaltar en la figura 25 del gráfico tiempo de alerta, que en los ataques de DOS y botnet no hay una diferencia significante entre ambos IDS, sin embargo en el ataque de inyección SQL suricata tiene un 30% menos del tiempo de detección que snort (34.85 Segundos) por lo que suricata es más rápido en detectar este ataque. En la figura 26, gráfico consumo de CPU, snort consumió menos CPU en el ataque de DOS, un 8% menos del valor de suricata (476.5 MHZ) y en el ataque de botnet consumió un 4% menos que el valor de suricata (840.55 MHZ) pero en el ataque de inyección SQL suricata llevó ventaja de 0.5% de consumo menos en relación al valor de snort (679,05 MHZ), por lo que snort es más eficiente en el tratamiento de procesos debido al valor del ataque de inyección SQL que no es muy significativo . En la figura 27, gráfico de consumo de memoria, en el ataque de DOS snort consumió un 3% menos del valor de suricata (0.36 Gb) y en el ataque de botnet un 28% menos del valor de suricata (0.62 Gb) pero en el ataque de inyección SQL fue suricata que consumió un 11% menos del valor de snort (0.47 Gb), snort administra de mejor manera la memoria aunque el valor de suricata en el ataque de inyección SQL es relevante . Por ultimo en la figura 28, gráfico consumo de red, suricata procesó menos paquetes que snort, en el ataque de DOS un 72% menos del valor de snort (4.84 Mbps), en el ataque de botnet un 14% menos del valor de snort (7.85 Mbps) y en el ataque de inyección SQL un 7% menos del valor de snort (9.11 Mbps), demostrando que snort es más eficiente en el tratamiento de paquetes de red

Discusión

Según el análisis de los 2 IDS evaluados: snort y suricata, suricata demuestra que es más eficaz en el momento de comparar reglas y alertar, pero consume más CPU y memoria que snort en los ataques de DOS y botnet y por eso causa un desgaste en el procesamiento de paquetes causando más perdida de paquetes que snort. Solo en el ataque de inyección SQL suricata es más eficiente que snort en el tartamiento de procesador y memoria por lo que el resultado del rendimiento de estos 2 IDS depende mucho del tipo de ataque que evalúen, y aunque snort demostró mejor aprovechamiento de recursos, la diferencia no es relevante en relación a lo que hizo suricata. Además, en los tres escenarios de ataque ningún IDS sobrepasó el umbral básico de un computador que está representado en el experimento con 2770 MHZ de procesador y 1Gb de memoria.

En contraste con lo analizado Alhomoud et al. (2011) dice que por la virtualización hay una cantidad considerable de paquetes perdidos en los experimentos y esto se puede apreciar en el trabajo realizado, también menciona que suricata se desempeñó mejor en Linux que en otras plataformas; Day and Burns (2011) alega que suricata tiene una alta tasa de precisión aunque tiene un mayor consumo de CPU lo que se puede notar en el trajo realizado, Albin (2011) menciona que el uso de CPU y memoria por parte de suricata es casi el doble que el de snort, que coincide con el aumento de estos recursos en el trabajo realizado; White et al. (2012) propone que suricata supera a snort, incluso con un solo

núcleo y que suricata tiene problemas en la escalabilidad de la red pasado los 3 núcleos asignados, lo que puede reflejarse en menor tiempo de procesamiento em paquetes con el trabajo realizado; Thongkanchorn et al. (2013) comenta que la tasa de tráfico tiene una relación directa con el consumo de CPU, paquetes perdidos y número de alertas, además que cada IDS actúa diferente para cada ataque, lo que es compatible con la conclusión del trabajo realizado; Ridho (2014) contribuye en decir que tanto snort y suricata consumen grandes recursos y son fáciles de instalar, no es muy acertada con la conclusión de que snort utiliza más recursos pero si en que es difícil de instalar; Brumen and Legvart (2016) expone que snort y suricata utilizan más recursos en Linux, un 25% más que en Windows, y que también depende el ataque simulado, además suricata usaba más recursos, lo que se asemeja al consumo de recursos por parte de suricata y conclusión en el trabajo realizado; Shah and Issac (2018) concluye que suricata requiere más recursos de CPU y memoria que snort debido a su arquitectura multiproceso y que la memoria consumida por snort es una pequeña fracción de memoria de un equipo moderno, lo que es evidente en el trabajo realizado; Hu et al. (2018) dice que snort consume más CPU que suricata y que suricata consume más memoria que snort con las configuraciones básicas de ambos IDS, algo que solo ocurrió en el ataque de inyección SQL del trabajo realizado y por ultimo Lukaseder et al. (2018) comenta que suricata utiliza más memoria que snort y que esta no tiene incidencia en el ancho de banda y cantidad de ataques, también en lo que respecta a CPU snort y surica aumentan considerablemente su uso a medida que se aumenta el ancho de banda de red a más de 1Gbps, lo que no podemos discutir en vista de que el trabajo realizado solo se limitó a probar con 1 Gbps de tráfico sin embargo en consumo de memoria son los mismos resultados dando ventaja a snort.

CONCLUSIONES, LIMITACIONES Y TRABAJOS FUTUROS

En conclusión, una pyme está en capacidad de comprar un equipo básico para configurar cualquiera de los 2 IDS analizados sin embargo snort aprovecha mejor los recursos en un escenario ideal con poca carga de trabajo de red. Suricata es ideal para infraestructuras más complejas por sus características de multiprocesamiento, y aunque consuma más recursos tiene un tiempo de detección mejor y es más fácil de instalar que snort.

Al mismo tiempo se pudo organizar y controlar el tráfico de red gracias a la tecnología de puertos SPAN, que logra un mejor tratamiento de los paquetes de red y monitoreo, también evita la sobrecarga de recursos del IDS, sobre todo en la tarjeta de red. De igual manera la utilización de un disco sólido para la instalación y operación de la infraestructura virtualizada logró disminuir considerablemente la latencia de lectura y escritura en el disco, lo que agilizó el rendimiento del disco duro.

Como recomendaciones se podría considerar mejorar la estructura utilizando bonding en las tarjetas de red del IDS en Linux o link agregation en los switch de capa 3 para un mejor tratamiento en paquetes de red, también actualizar automáticamente en línea las reglas de los IDS con el mismo fabricante para evitar ataques del día cero y optimizar la detección de ataques por parte de los IDS y por ultimo utilizar la última versión del hipervisor operado en el experimento porque tiene características de guardar infraestructuras en la nube, de esta manera se logrará tener mejor movilidad y respaldo de las infraestructuras virtualizadas.

Entre las limitaciones encontradas se puede cotar el hardware requerido para las pruebas puesto que para realizar una simulación virtualizada se necesita mínimo 24 Gb en el equipo huésped y se necesita ejecutar 12 Gb en la VMware ESXi y 12 Gb en la compartición del emulador de red y análisis de red con Wireshark, para así tener más disponibilidad de recursos. Otra limitación importante fue la tarea de buscar y conseguir malwares adecuados y confiables como *botnets*, debido a que algunas páginas o servicios web han sido cerrados por temas de abuso o mal uso, por lo que se tuvo que realizar configuraciones de *botnets* básicas y analizarlas por listas negras.

Para ampliar la confiabilidad de los resultados observados en los escenarios construidos, se debería evaluar la eficacia en el mismo entorno de red propuesto para los IDS estudiados, pero con una variedad más amplia de reglas y un conjunto de datos generados más actualizados, también hacer un análisis de tiempo en las alertas automatizadas que no sean del log del IDS sino de algún tipo de herramienta que avise al administrador de sistemas, por ultimo este mismo análisis compararlos en diferentes plataformas de sistemas operativos diferentes a Linux y con otros tipos de ataques.

BIBLIOGRAFÍA

Akhlaq, M., Alserhani, F., Awan, I., Mellor, J., Cullen, A. J., & Al-Dhelaan, A. (2011). Implementation and evaluation of network intrusion detection systems *Network performance engineering* (pp. 988-1016): Springer.

Akhlaq, M., Alserhani, F., Awan, I. U., Mellor, J., Cullen, A. J., & Mirchandani, P. (2009). *Virtualization Efficacy for Network Intrusion Detection Systems in High Speed Environment.* Paper presented at the International Conference on Information Security and Digital Forensics.

Albin, E. (2011). *A comparative analysis of the snort and suricata intrusion-detection systems.* Monterey, California. Naval Postgraduate School.

Alessandri, D., Cachin, C., Dacier, M., Deak, O., Julisch, K., Randell, B., . . . Wüest, C. (2001). Towards a taxonomy of intrusion detection systems and attacks. *MAFTIA Deliverable D, 3*.

Alhomoud, A., Munir, R., Disso, J. P., Awan, I., & Al-Dhelaan, A. (2011). Performance evaluation study of intrusion detection systems. *Procedia Computer Science, 5*, 173-180.

Alqahtani, S. M., & John, R. (2016). A Comparative Study of Different Fuzzy Classifiers for Cloud Intrusion Detection Systems' Alerts.

Americanos, B. I. d. D. y. l. O. d. E. (2016). Cybersecurity: Are We Ready in Latin America and the Caribbean. from https://publications.iadb.org/handle/11319/7449?locale-attribute=en&locale-attribute=pt&localeattribute=es&.

Amorós, J. E., Planellas, M., & Foguet, J. M. B. (2010). Influencia de la utilización de Internet en el crecimiento de las pequeñas y medianas empresas: un estudio empírico en una economía en desarrollo. *Universidad & Empresa, 8*(10), 89-113.

Anderson, D., Frivold, T., & Valdes, A. (1995). Next-generation intrusion detection expert system (NIDES): A summary.

Antonatos, S., Anagnostakis, K. G., & Markatos, E. P. (2004). *Generating realistic workloads for network intrusion detection systems.* Paper presented at the ACM SIGSOFT Software Engineering Notes.

appneta. (2019). Tcpreplay. from https://tcpreplay.appneta.com/

Arquilla, J., & Ronfeldt, D. (1993). Cyberwar is coming! *Comparative Strategy, 12*(2), 141-165.

Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security (TISSEC), 3*(3), 186-205.

Bace, R., & Mell, P. (2001). NIST special publication on intrusion detection systems: BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

Bass, T. (2000). Intrusion detection systems and multisensor data fusion. *Communications of the ACM, 43*(4), 99-105.

Behal, S., & Kumar, K. (2011). *An experimental analysis for malware detection using extrusions.* Paper presented at the Computer and Communication Technology (ICCCT), 2011 2nd International Conference on.

Brumen, B., & Legvart, J. (2016). *Performance analysis of two open source intrusion detection systems.* Paper presented at the Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2016 39th International Convention on.

Bulajoul, W., James, A., & Pannu, M. (2013). *Network intrusion detection systems in high-speed traffic in computer networks.* Paper presented at the e-Business Engineering (ICEBE), 2013 IEEE 10th International Conference on.

Cavusoglu, H., Mishra, B., & Raghunathan, S. (2005). The value of intrusion detection systems in information technology security architecture. *Information Systems Research, 16*(1), 28-46.

CISCO. (2018). Reporte anual de ciberseguridad. from https://www.cisco.com/c/dam/global/es_mx/solutions/pdf/reporte-anual-cisco-2018-espan.pdf

Combs, G. (2019). About Wireshark from https://www.wireshark.org/

Chanthakoummane, Y., Saiyod, S., & Khamphakdee, N. (2015). *Evaluation snort-IDS rules for botnets detection.* Paper presented at the National Conference on Infomation Technology.

Christodorescu, M., Jha, S., Seshia, S. A., Song, D., & Bryant, R. E. (2005). *Semantics-aware malware detection.* Paper presented at the Security and Privacy, 2005 IEEE Symposium on.

Daniel, E. M., & Grimshaw, D. J. (2002). An exploratory comparison of electronic commerce adoption in large and small enterprises. *Journal of Information Technology, 17*(3), 133-147.

Day, D., & Burns, B. (2011). *A performance analysis of snort and suricata network intrusion detection and prevention engines.* Paper presented at the Fifth International Conference on Digital Society, Gosier, Guadeloupe.

de Assis, C. G. (2017). La nueva era de la información como poder y el campo de la

ciberinteligencia. *URVIO: Revista Latinoamericana de Estudios de Seguridad*(20), 94-109.

De Berranger, P., Tucker, D., & Jones, L. (2001). Internet diffusion in creative micro-businesses: identifying change agent characteristics as critical success factors. *Journal of Organizational Computing and Electronic Commerce, 11*(3), 197-214.

Delgado, J. A. (2014). Internet Governance in Ecuador: Infrastructure and Access.

Dholakia, R. R., & Kshetri, N. (2004). Factors impacting the adoption of the Internet among SMEs. *Small Business Economics, 23*(4), 311-322.

DIGIWARE. (2015). DESCUBRIENDO EN EL SUBCONSCIENTE DE LA COTIDIANIDAD LO CONSCIENTE DE LA AMENAZA. from http://www.digiware.net/sites/default/files/Tendencias-Seguridad-2015.pdf

Douligeris, C., & Mitrokotsa, A. (2004). DDoS attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks, 44*(5), 643-666.

ESET. (2018). eset security report latinoamérica 2018. from https://www.welivesecurity.com/wp-content/uploads/2018/06/ESET_security_report_LATAM2018.pdf

Forum, W. E. (2018). The Global Risks Report 2018 13th Edition from http://www3.weforum.org/docs/WEF_GRR18_Report.pdf

Francis, T. (2017). Known Bot Command and Control Rules from https://docs.emergingthreats.net/bin/view/Main/BotCC

Freiling, F. C., Holz, T., & Wicherski, G. (2005). *Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks.* Paper presented at the European Symposium on Research in Computer Security.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security, 28*(1-2), 18-28.

Gascon, H., Orfila, A., & Blasco, J. (2011). Analysis of update delays in signature-based network intrusion detection systems. *computers & security, 30*(8), 613-624.

Ghafir, I., Prenosil, V., Svoboda, J., & Hammoudeh, M. (2016, 22-24 Aug. 2016). *A Survey on Network Security Monitoring Systems.* Paper presented at the 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW).

github. (2019). Snort-Rules. from https://github.com/Simon1207/Snort-Rules/blob/master/local.rules

Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richardson, R. (2005). 2005 CSI/FBI computer crime and security survey. *Computer Security Journal, 21*(3), 1.

Hall, M., & Wiley, K. (2002). *Capacity verification for high speed network intrusion detection systems.* Paper presented at the International Workshop on Recent Advances in Intrusion Detection.

Ho, C.-Y., Lai, Y.-C., Chen, I.-W., Wang, F.-Y., & Tai, W.-H. (2012). Statistical analysis of false positives and false negatives from real traffic with intrusion detection/prevention systems. *IEEE Communications Magazine, 50*(3).

Hoque, M. S., Mukit, M., Bikas, M., & Naser, A. (2012). An implementation of intrusion detection system using genetic algorithm. *arXiv preprint arXiv:1204.1336*.

Hu, Q., Asghar, M. R., & Brownlee, N. (2018). Effectiveness of Intrusion Detection Systems in High-speed Networks. *International Journal of Information, Communication Technology and Applications, 4*(1), 1-10.

Jin, H., Xiang, G., Zhao, F., Zou, D., Li, M., & Shi, L. (2009). *VMFence: a customized intrusion prevention system in distributed virtual computing environment.* Paper presented at the Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication.

Jinsheng Xu, J. Z., Triveni Gadipalli, Xiaohong Yuan and, & Yu, H. (2011). *Learning Snort Rule By Capturing Intrusions in Live Interwork Traffic Replay*. Paper presented at the Proceedings of the 15th Colloquium for Information Systems Security Education (CISSE).

Kabiri, P., & Ghorbani, A. A. (2005). Research on intrusion detection and response: A survey. *IJ Network Security, 1*(2), 84-102.

Kalam, A. A. E., Rab, M. G. E., & Deswarte, Y. (2014). A model-driven approach for experimental evaluation of intrusion detection systems. *Security and Communication Networks, 7*, 1955-1973.

Karim, I., Vien, Q.-T., Le, T. A., & Mapp, G. (2017). A Comparative Experimental Design and Performance Analysis of Snort-Based Intrusion Detection System in Practical Computer Networks. *Computers, 6*, 6.

Kula, V., & Tatoglu, E. (2003). An exploratory study of Internet adoption by SMEs in an emerging market economy. *European Business Review, 15*(5), 324-333.

Lazarevic, A., Kumar, V., & Srivastava, J. (2005). *Chapter 2 INTRUSION DETECTION : A SURVEY*.

Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., & Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications, 36*(1), 16-24.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., . . . Cunningham, R. K. (2000). *Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation.* Paper presented at the DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings.

Lukaseder, T., Fiedler, J., & Kargl, F. (2018). *Performance Evaluation in High-Speed Networks by the Example of Intrusion Detection.* Paper presented at the DFN-Forum Kommunikationstechnologien.

Lunt, T. F., Jagannathan, R., Lee, R., Listgarten, S., Edwards, D. L., Neumann, P. G., . . . Valdes, A. (1988). *Ides: The enhanced prototype-a real-time intrusion-detection expert system.* Paper presented at the SRI International, 333 Ravenswood Avenue, Menlo Park.

McHugh, J., Christie, A., & Allen, J. (2000). Defending yourself: The role of intrusion detection systems. *IEEE software, 17*(5), 42-51.

Medero, G. S. (2018). Internet: Una herramienta para las guerras en el siglo XXI. *Revista Política y Estrategia*(114), 224-242.

MIcro, T. (2015). Report on Cybersecurity and Critical Infrastructure in the Americas., from https://www.trendmicro.de/cloud-content/us/pdfs/security-intelligence/reports/critical-infrastructures-west-hemisphere.pdf

Milenkoski, A. (2015). *Evaluation of Intrusion Detection Systems in Virtualized Environments.* Paper presented at the RAID.

Milenkoski, A., Jayaram, K. R., Antunes, N., Vieira, M., & Kounev, S. (2016, 23-27 Oct. 2016). *Quantifying the Attack Detection Accuracy of Intrusion Detection Systems in Virtualized Environments.* Paper presented at the 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE).

Mukhopadhyay, I., Chakraborty, M., & Chakrabarti, S. (2011). A comparative study of related technologies of intrusion detection & prevention systems. *Journal of Information Security, 2*(01), 28.

Mutz, D., Vigna, G., & Kemmerer, R. A. (2003). *An Experience Developing an IDS Stimulator for the Black-Box Testing of Network Intrusion Detection Systems.* Paper presented at the ACSAC.

Nye Jr, J. S. (2010). Cyber power: HARVARD UNIV CAMBRIDGE MA BELFER CENTER FOR SCIENCE AND INTERNATIONAL AFFAIRS.

Nye, J. S. (2014). The information revolution and soft power.

Ossec. (2018). Welcome to OSSEC's documentation! , from https://www.ossec.net/docs/

owasp. (2016). SQL Injection. from https://www.owasp.org/index.php/SQL_Injection

Patcha, A., & Park, J.-M. (2007). An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer networks, 51*(12), 3448-3470.

Paulauskas, N., & Skudutis, J. (2008). Investigation of the intrusion detection system "snort" performance. *Elektronika ir elektrotechnika, 87*(7), 15-18.

Porras, P. A., & Valdes, A. (1998). *Live Traffic Analysis of TCP/IP Gateways.* Paper presented at the NDSS.

Powell, D., & Stroud, R. (2001). Conceptual Model and Architecture, Deliverable D2, Project MAFTIA IST-1999-11583. *IBM Zurich Research Laboratory Research Report RZ, 3377*.

Rapid7, C. d. c. a. y. (2019). Getting Started. from https://metasploit.help.rapid7.com/docs

Ridho, M. F. (2014). *Analysis And Evaluation Snort, Bro, And Suricata As Intrusion Detection System Based On Linux Server.* Universitas Muhammadiyah Surakarta.

Roesch, M. (1999). *Snort: Lightweight intrusion detection for networks.* Paper presented at the Lisa.

Saber, M., Belkasmi, M. G., Chadli, S., Emharraf, M., & Farissi, I. E. (2017). *Implementation and Performance Evaluation of Intrusion Detection Systems under high-speed networks.* Paper presented at the BDCA.

Salzman, R., & Albarran, A. B. (2011). Internet Use in Latin America. *Palabra Clave, 14*(2), 297-313.

Sánchez Medero, G. (2012). Ciberespacio y el Crimen Organizado. Los nuevos desafíos del siglo XXI. *Revista Enfoques: Ciencia Política y Administración Pública, 10*(16).

Sayantan, B. (2019). Detect SQL Injection Attack using Snort IDS. from https://www.hackingarticles.in/detect-sql-injection-attack-using-snort-ids/

Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (idps). *NIST special publication, 800*(2007), 94.

Security, O. (2019). About Kali Linux. from https://www.kali.org/about-us/

Shah, S. A. R., & Issac, B. (2018). Performance Comparison of Intrusion Detection

Systems and Application of Machine Learning to Snort System. *Future Generation Comp. Syst., 80*, 157-170.

Singh, D., Patel, D., Borisaniya, B., & Modi, C. (2016). Collaborative ids framework for cloud. *International Journal of Network Security, 18*(4), 699-709.

snort-org. (2019). Preprocessors. from http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node17.html

Sun, B., Osborne, L., Xiao, Y., & Guizani, S. (2007). Intrusion detection techniques in mobile ad hoc and wireless sensor networks. *IEEE Wireless Communications, 14*(5).

SYMANTEC. (2018). Informe sobre las amenazas para la seguridad de Internet. from https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf

Telecomunicaciones, U. I. d. (2017). Global Cybersecurity Index from https://www.itu.int/dms_pub/itu-d/opb/str/D-STR-GCI.01-2017-PDF-E.pdf

Thongkanchorn, K., Ngamsuriyaroj, S., & Visoottiviseth, V. (2013). *Evaluation studies of three intrusion detection systems under various attacks and rule sets.* Paper presented at the TENCON 2013-2013 IEEE Region 10 Conference (31194).

UNB. (2019). Intrusion Detection Evaluation Dataset (CICIDS2017). from https://www.unb.ca/cic/datasets/ids-2017.html

Wang, X., Kordas, A., Hu, L., Gaedke, M., & Smith, D. (2013). *Administrative evaluation of intrusion detection system.* Paper presented at the Proceedings of the 2nd annual conference on Research in information technology.

White, J. S., Fitzsimmons, T. T., Matthews, J. N., & Coulter, W. H. (2012). *Quantitative Analysis of Intrusion Detection Systems : Snort and Suricata*.

Xu, J., Zhao, M., Fortes, J., Carpenter, R., & Yousif, M. (2007). *On the use of fuzzy modeling in virtualized data center management.* Paper presented at the Autonomic Computing, 2007. ICAC'07. Fourth International Conference on.

Zhang, Y., Lee, W., & Huang, Y.-A. (2003). Intrusion detection techniques for mobile wireless networks. *Wireless Networks, 9*(5), 545-556.