UNIVERSIDAD NACIONAL SAN CRISTÓBAL DE HUAMANGA

FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

"Snort Open Source como sistema de detección de intrusos para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020".

Tesis presentada por **:** Bach. Hayde Luisa Huamani Santiago

Para Optar el título profesional de: Ingeniero de Sistemas

Tipo de Investigación **:** Aplicada, descriptiva

Área de Investigación **:** Seguridad Informática

Asesor **:** Mg. Ing. Hubner Janampa Patilla

Ayacucho - Perú

2020

DEDICATORIA

A mis padres, por inculcarme el deseo de superación e incondicional apoyo, pero sobre todo, por el amor incondicional que me brindan en todo momento.

AGRADECIMIENTO

A la Universidad Nacional de San Cristóbal de Huamanga, en especial a la plana docente de la Escuela de ingeniería de sistemas, quienes con sentido crítico y vocación supieron guiarme a lo largo de mi formación profesional.

A mi asesor, el Mg. Ing. Hubner Jananpa Patilla, por el tiempo dedicado y su valiosa guía en la elaboración de este proyecto.

A todas aquellas personas y amistades que de forma directa o indirecta hicieron posible la elaboración de esta tesis.

CONTENIDO

CAPITULO III

METODOLOGÍA DE LA INVESTIGACIÓN

RESUMEN

Actualmente los ataques informáticos se han ido incrementando en el Perú, afectando a diferentes empresas y organizaciones. A su vez ha provocado que los sistemas de detección de intrusiones (IDS) sean imprescindibles en el esquema de seguridad de las redes empresariales, esto debido a que los ataques informáticos son cada vez más sofisticados y difíciles de detectar, pero un sistema de prevención y detección de intrusiones en la red mejora la detección de paquetes de red, monitorea el tráfico de red entrante y saliente e identifica el uso no autorizado y el mal manejo de las redes de los sistemas informáticos. Sin embargo, la mayoría de las Pymes todavía no cuentan con este esquema de seguridad por diferentes motivos entre ellas el costo que significaría implementar un sistema de detección de intrusiones (IDS).

El objetivo de la investigación fue implementar Snort Open Source como sistema de detección de intrusos en la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú. Para alcanzar los objetivos de la investigación, con la ayuda de una máquina virtual se implementó un escenario donde se utilizaron varias herramientas para llevar a cabo los experimentos de investigación e implementación, para luego validar el funcionamiento de Snort Open Source.

Del Sistema de Detección de Intrusiones Snort, propuesto a nivel de red para las Pymes se obtiene las siguientes conclusiones: a) Se logró implementar y mostrar la utilidad del IDS snort, b) Se logró implementar reglas en el motor de reglas, realizando el filtrado de los eventos y vulnerabilidades, c) Se logró monitorear con éxito el tráfico de la red de la pyme, logrando así monitorear el tráfico autorizado y no autorizado.

Palabra clave: Snort Open Source, Sistema de Detección de intrusiones, Pymes, Ataques informáticos.

INTRODUCCIÓN

Snort Open Source es un sistema de detección de intrusos en red, libre y gratuito, Sistema de Detección de intrusiones basado en la red (NIDS). Que implementa un motor de detección de ataques y barrido de puertos que permite registrar, alertar y responder ante cualquier anomalía en tiempo real. (Costas Santos, 2014).

Los riegos de la ciberseguridad en las Pymes son más frecuentes debido a que son pequeñas empresas que están en crecimiento. En primer lugar, está la falta de concientización de los dueños de las empresas y de los colaboradores en proteger la información más crítica de la empresa, y en segundo lugar, la falta de conocimiento de cómo proteger la información crítica, pese a que algunos controles para su implementación no tiene costo (Farro Flores, 2019).

Mi motivación para implementar Snort Open Source, es que las Pymes que no cuentan con ningún tipo de seguridad frente a posibles ataques informáticos, puedan tener una alternativa de solución open source y tomar en cuenta medidas de seguridad a nivel de red, entre estas medidas de seguridad se encuentra Snort como Sistema de Detección de Intrusos. Este sistema tiene como objetivo detectar las acciones que intentan comprometer la confidencialidad, disponibilidad e integridad de un recurso mediante la supervisión de los eventos que ocurren a nivel de red. Para así lograr contar con seguridad dentro las Pymes teniendo en cuenta que la información es uno de los recursos más relevantes y debe ser protegida de forma adecuada.

Los objetivos específicos son: a) Diseñar el Snort Open Source como sistema de detección de intrusos a nivel de interfaz de red, en la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020. b) Diseñar el Snort Open Source como sistema de detección de intrusos para métodos de captura, en la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020. c) Diseñar el Snort Open Source como sistema de detección de intrusos a nivel de motor de reglas y filtrado de eventos, para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020.

CAPITULO I
PLANTEAMIENTO DE LA INVESTIGACIÓN

1.1. DIAGNOSTICO Y ENUNCIADO DEL PROBLEMA

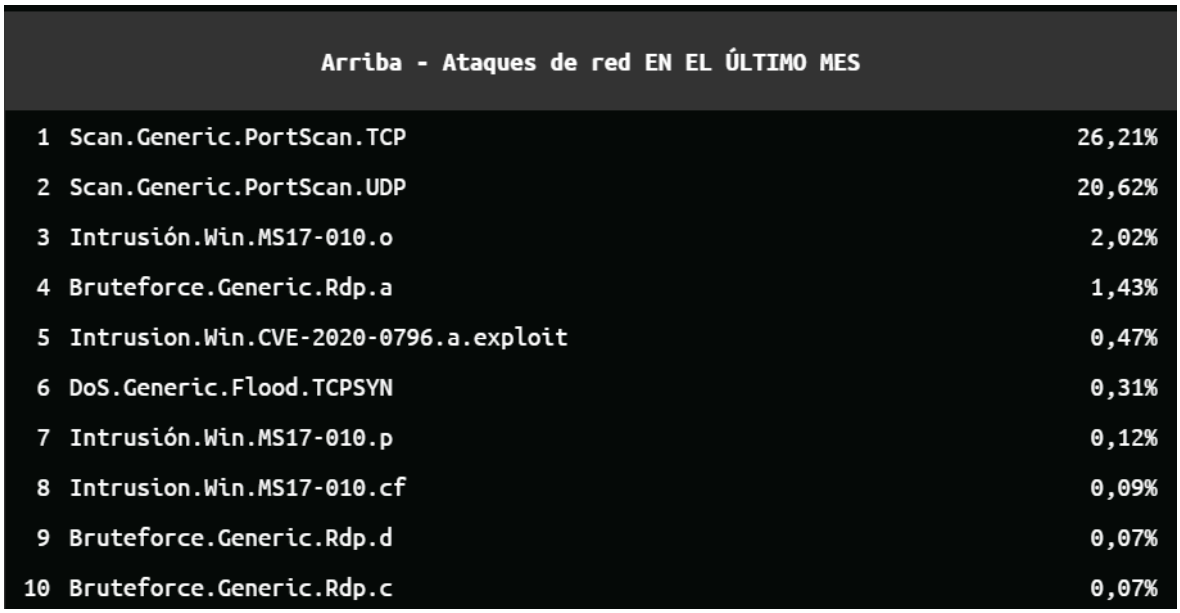
Para Francois Carpentier (2016), los riesgos que hacen que una empresa sea vulnerable son las amenazas externas e internas, software malicioso y ataques que afectan al sistema de información. Hoy en día el sistema de información se extiende en gran medida fuera de las fronteras de la empresa, es importante establecer estrategias de protección.

Según el Diario El Peruano (2020), el Perú es el segundo país con menos ciberseguridad en América Latina, después de Brasil. Asimismo, a nivel mundial Perú ocupa el puesto 17. Donde incluye factores como porcentaje de infecciones de malware móvil, número de ataques de malware financiero, infecciones de malware informático, ataques telnet, ataques de criptomineros, preparación ante ciberataques y legislación sobre ciberseguridad. Los puntajes se basaron en el porcentaje de usuarios atacados durante el tercer trimestre de 2019 en 76 países.

En la figura 1.1 indica la cantidad de ataques de red registrados en el Perú entre el 28 de mayo del 2020 al 27 de junio del 2020. De donde podemos indicar que se registró mayor incidencia el 4 de junio del 2020, llegando a registrar 430 267 de ataques en solo ese día.

Figura 1.1. Estadística de Perú, ataque a redes (https://cybermap.kaspersky.com/, 2020).

Como se puede observar en la figura 1.2, los ataques con mayor frecuencia durante este período son de tipo scan.



Arriba - Ataques de red EN EL ÚLTIMO MES		
1	Scan.Generic.PortScan.TCP	26,21%
2	Scan.Generic.PortScan.UDP	20,62%
3	Intrusión.Win.MS17-010.o	2,02%
4	Bruteforce.Generic.Rdp.a	1,43%
5	Intrusion.Win.CVE-2020-0796.a.exploit	0,47%
6	DoS.Generic.Flood.TCPSYN	0,31%
7	Intrusión.Win.MS17-010.p	0,12%
8	Intrusion.Win.MS17-010.cf	0,09%
9	Bruteforce.Generic.Rdp.d	0,07%
10	Bruteforce.Generic.Rdp.c	0,07%

Figura 1.2. Tipos de ataques frecuentes (https://cybermap.kaspersky.com/, 2020).

Para Farro Flores (2019), los riesgos de ciberseguridad en las Pymes son más frecuentes debido a que son pequeñas empresas que están en crecimiento, y muchas de ellas no tienen personal de TI dedicado. En primer lugar, está la falta de concientización de los dueños de las empresas y de los colaboradores en proteger la información más crítica de las empresa y en segundo lugar, la falta de conocimiento de cómo proteger la información crítica a pesar que algunos controles para su implementación no tiene costo.

Según Alva (2019) en una nota publicada en el diario La Gestión, durante el periodo de abril y setiembre del 2019, las empresas peruanas sufrieron más de 3,000 millones de intentos de ciberataques.

Según Bardales (2019) en una nota publicada en el diario La Gestión, los delitos cibernéticos no solo afectan a las grandes empresas, ya que estos actos también tienen en la mira a las pequeñas y medianas empresas (Pymes) y una de cada cinco de estas son víctimas de este tipo de ataques. En el caso de las Pymes no implementan medidas de seguridad apropiadas en el manejo de su información o no confían el manejo de su información a compañías especializadas en servicios tecnológicos.

1.2. DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN

PROBLEMA GENERAL

¿Como el **Snort Open Source** implementa el sistema de detección de intrusos para la **seguridad de la infraestructura de red en entornos libres aplicado** a Pymes del Perú, 2020?

PROBLEMAS ESPECÍFICOS

a. ¿Como el **Snort Open Source** implementa el sistema de detección de intrusos **a nivel de interfaz de red**, para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020?

b. ¿Como el **Snort Open Source** implementa el sistema de detección de intrusos **como métodos de captura**, para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020?

c. ¿Como el **Snort Open Source** implementa el sistema de detección de intrusos **a nivel de motor de reglas y filtrado de eventos**, para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020?

1.3. OBJETIVO GENERAL

Implementar **Snort Open Source** como sistema de detección de intrusos en la **seguridad de la infraestructura de red en entornos libres aplicado** a Pymes del Perú, 2020.

1.4. OBJETIVOS ESPECÍFICOS

a. Implementar el **Snort Open Source** como sistema de detección de intrusos **a nivel de interfaz de red**, en la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020.

b. Implementar el **Snort Open Source** como sistema de detección de intrusos **para métodos de captura**, en la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020.

c. Implementar el **Snort Open Source** como sistema de detección de intrusos **a nivel de motor de reglas y filtrado de eventos**, para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020.

1.5. JUSTIFICACIÓN

Actualmente las Pymes no invierten en seguridad informática debido a que existen muchos obstáculos para una implementación de un sistema de detección de intrusos a nivel de red (NIDS), ya sea porque involucra realizar costos al momento de implementarlos o por desconocimiento, esto hace que sean más vulnerables a los distintos tipos de ataques de la red. Debido a esta situación en el presente trabajo de investigación se implementó el Snort Open Source como sistema de detección de intrusos en la seguridad de la infraestructura de red en entornos libres y su aplicación a las Pymes del Perú. El sistema de detección de intrusos aporta a la red un grado de seguridad de tipo preventivo ante de cualquier actividad sospechosa, y consigue su objetivo a través de alertas anticipadas generadas por el motor de detección.

1.6. DELIMITACIÓN

La investigación se realizó sobre el sistema de detención de intrusiones Snort Open Source, abarcando la interfaz de red, métodos de captura, motor de reglas y filtrado de eventos.

CAPITULO II
REVISIÓN DE LA LITERATURA

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

Coyla Jarita (2019), en su investigación denominado "Implementación de un sistema de detección y prevención de intrusos (IDS/IPS), basado en la norma ISO 27001, para el monitoreo perimetral de la seguridad informática, en la red de la Universidad Peruana Unión – Filial Juliaca", de la Universidad Peruana Unión, concluye que se logró mostrar la utilidad de la herramienta Snort con respecto a la seguridad perimetral de la información de la red y la metodología ISO 27001 mejora la calidad de la implementación de sistemas de seguridad informática, al monitorear el tráfico y realizar las pruebas necesarias, logrando probar que en realidad nos muestra el resultado requerido.

Yauri Lozano (2017) en su investigación denominado "SnorUNI: Aplicación móvil para Sistemas de Detección y Prevención de Intrusiones basados en Snort", de la Universidad Nacional de Ingeniería, concluye que se logró implementar y configurar las funcionalidades requeridas para realizar la conexión y el intercambio de información con la aplicación móvil. Asimismo, en la fase de desarrollo de SnorUNI, se lograron implementar las funcionalidades planteadas en los objetivos. De esa forma, la aplicación cumple con su objetivo principal de brindar información rápida y organizada a los usuarios en cualquier lugar y a cualquier hora.

Izquierdo Cabrera y Tafur Callirgos (2017) en su investigación denominado "Mecanismos de Seguridad para Contrarrestar Ataques Informáticos en Servidores Web y Base de Datos", de la Universidad Señor de Sipán, concluyen que el mecanismo de seguridad Snort en Kali Linux, llegó a ser más preciso al momento de detectar el tráfico malicioso y el tráfico normal, refiriéndose a la especificidad. Mientras que el mecanismo de seguridad Honeynet obtuvo mayor porcentaje de exactitud y capacidad en la clasificación del tráfico normal.

Llopis Polvoreda (2017) en su investigación denominado "Sistema de Monitorización del IDS Snort", de la Universitat Politècnica de València, afirma que una buena alternativa para mejorar la seguridad de nuestros datos, consiste en la instalación de IDSes o Sistemas de Detección de Intrusos, cuya principal función es la de detectar accesos no deseados a nuestra red o comportamientos anómalos en el interior de la misma.

2.2. MARCO TEÓRICO

2.2.1. PROTOCOLOS TCP / IP EN LA DETECCIÓN DE INTRUSOS

Los protocolos TCP / IP en el contexto de la detección de intrusos son:

A. TCP

Para Cox y Gerg (2004), TCP es un protocolo de capa de transporte orientado a la conexión diseñado para proporcionar una conexión confiable para el intercambio de datos entre dos sistemas. TCP garantiza que todos los paquetes se secuencian y reconocen correctamente, y que se establece una conversación antes de enviar los datos. Esto asegura que ambas máquinas estén listas para tener una conversación y que la información que se mueve de un sistema a otro lo haga sin perder nada. Algunas aplicaciones que utilizan TCP como método de comunicación son:

a) Protocolo de terminal virtual (puerto Telnet 23).
b) Protocolo de transferencia de archivos (puertos FTP 20 y 21).
c) Protocolo simple de transferencia de correo (puerto 25 de SMTP).
d) Secure Shell (puerto SSH 23).

TCP proporciona su fiabilidad mediante el uso de un reconocimiento (los geeks de la red llaman a esto un ACK). Un ACK es devuelto por una máquina receptora a una máquina emisora para decirle al remitente que el mensaje que se envió se recibió sin error. Si el remitente no recibe un ACK, reenvía el mensaje.

Para Gordon (2019), se hace referencia a TCP como protocolo orientado a la conexión. Lo que esto significa es que, antes de que se transfieran datos entre dispositivos, es necesario establecer una conexión. Al hacer esto, el dispositivo de envío está seguro de que el destinatario está listo para recibir los datos. Para formar esta conexión, TCP lleva a cabo un proceso conocido como protocolo de enlace de tres vías, que se muestra en la siguiente figura.

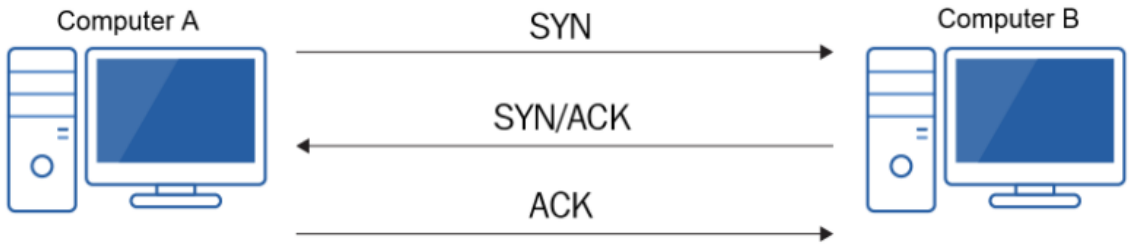


Figura 2.1. Protocolo de enlace de tres vías TCP (Gordon, 2019).

B. UDP

Para Cox y Gerg (2004) el UDP proporciona un sistema no confiable y sin conexión para entregar paquetes. En lugar de proporcionar mecanismos para garantizar la entrega y la secuencia, UDP permite que las aplicaciones de nivel superior se preocupen por los datos perdidos o fuera de secuencia. Este protocolo permite que se envíen mensajes (llamados datagramas con UDP) sin la sobrecarga involucrada con los ACK y el establecimiento de un enlace de comunicaciones. UDP se utiliza principalmente para comunicaciones de difusión o juegos de computadora con reconocimiento de red. Las aplicaciones de nivel superior que usan UDP como mecanismo de comunicación incluyen:

a) Protocolo trivial de transferencia de archivos (puerto TFTP 79)
b) Transmisiones
c) Sistema de archivos de red (puerto NFS 2049)
d) Unreal Tournament 2004 (puerto 7777)

UDP es un protocolo sin conexión. Esto significa que no se lleva a cabo un protocolo de enlace de tres vías antes de que se transmitan los datos. El dispositivo de envío envía literalmente los datos por el cable y espera que el dispositivo de destino los reciba. A menudo se lo denomina protocolo de mejor esfuerzo; no importa si los datos llegan o no. Esto puede parecer extraño porque seguramente desea recibir todos los datos en todo momento (Gordon, 2019).

UDP es un protocolo de capa de transporte simple, orientado a datagramas que preserva los límites de los mensajes. No proporciona corrección de errores, secuenciación, eliminación de duplicados, control de flujo o control de congestión. Puede proporcionar detección de errores e incluye la primera suma de verificación verdadera de extremo a extremo en la capa de transporte que hemos encontrado. Este protocolo proporciona una funcionalidad mínima en sí mismo, por lo que las aplicaciones que lo utilizan tienen un gran control sobre cómo se envían y procesan los paquetes. Las aplicaciones que deseen asegurarse de que sus datos se entreguen o secuencian de manera confiable deben implementar estas protecciones por sí mismas (Stevens y Fall, 2011).

C. IP

Para Cox y Gerg (2004), el Protocolo de Internet (IP) se usa para manejar servicios de datagramas entre hosts. Maneja el direccionamiento, el enrutamiento, la fragmentación y el reensamblaje de paquetes. Las direcciones IP tienen 32 bits de longitud y están organizadas en 4 octetos separados por puntos. Por ejemplo: 172.30.17.45.

D. ICMP

Para Cox y Gerg (2004) el Protocolo de mensajes de control de Internet (ICMP) realiza cuatro funciones principales:

a) **Control de flujo:** Cuando un sistema está demasiado ocupado para manejar flujos de datos entrantes, ICMP envía un mensaje de Fuente de enfriamiento para detener el flujo.
b) **Alertas de destinos inalcanzables:** Si no se puede acceder a un sistema, debido a que una dirección no coincide con un sistema en la red, o debido a una falla de enlace, un enrutador enviará un mensaje de Destino inalcanzable a la máquina emisora.
c) Redireccionando rutas: Una máquina de puerta de enlace puede dirigir una máquina de envío a otra puerta de enlace si sabe que existe una ruta preferencial a la red en la que reside el sistema de destino. Lo hace enviando un mensaje de redireccionamiento ICMP.
d) Comprobación de hosts remotos: Los mensajes de eco ICMP se utilizan para verificar la conectividad a un sistema de destino. Estos se llaman comúnmente pings.

E. ARP

Para Cox y Gerg (2004) la tarjeta de interfaz de red tiene un número de serie único asociado (llamada dirección MAC). En los niveles más bajos, este número de serie se utiliza para dirigir paquetes de red a hosts específicos en la red local. Para enviar un paquete a otra red, se requiere una dirección IP. Cartografía Las direcciones IP a estas direcciones MAC se manejan con el Protocolo de resolución de direcciones (ARP). Un sistema construirá una tabla de resolución de direcciones dinámicamente a medida que aprende de los hosts en la red local.

2.2.1.1. INTERFAZ DE RED TCP/IP

Según IBM Knowledge Center (s.f), la capa de interfaz de red **TCP/IP** formatea los datagramas IP de la capa de red en paquetes que las tecnologías de red específicas pueden interpretar y transmitir. Una interfaz de red es el software específico de red que se comunica con el controlador de dispositivo específico de red y la capa IP a fin de proporcionar a la capa IP una interfaz coherente con todos los adaptadores de red que puedan estar presentes.

TCP/IP soporta los tipos de interfaces de red:

a) Ethernet Versión 2 estándar (en)
b) IEEE 802.3 (et)
c) Red en anillo (tr)
d) SLIP (Serial Line Internet Protocol)
e) Bucle de retorno (lo)
f) FDDI
g) Óptica serie (so)
h) PPP (Point-to-Point Protocol - Protocolo de punto a punto)
i) Dirección IP virtual (vi)

Las interfaces Ethernet, 802.3 y de Red en anillo son para utilizarse con las redes de área local (LAN). La interfaz **SLIP** es para utilizarse con conexiones serie. La interfaz de bucle de retorno la utiliza un sistema principal para devolverse mensajes a sí mismo. La interfaz Óptica serie es para utilizarse con redes ópticas de punto a punto utilizando el Manejador de dispositivos de enlace óptico serie.

2.2.2. SISTEMA DE DETECCIÓN DE INTRUSIONES (IDS)

Para Arteaga (2020), la detección de intrusiones es la práctica de utilizar herramientas inteligentes y automáticas para detectar intentos de intrusión en tiempo real. Dichas herramientas se llaman Sistemas de Detección de Instrucciones.

Los sistemas de detección de intrusos (IDS) son una herramienta de monitoreo y detección de primera línea. La mejor ubicación para un IDS pasivo está conectada a un conmutador central donde fluye una parte significativa del tráfico de la red. La detección de intrusiones también puede funcionar en línea, bloqueando el tráfico según las coincidencias de reglas.

El error más común es el falso positivo, alertas que resultan no ser intrusiones reales (Thompson, 2020).

Para Messier (2019), un sistema de detección de intrusiones (IDS) de red puede tomar algunos de los mismos tipos de reglas de los cortafuegos y generar mensajes de registro. Una regla para un IDS generalmente puede basarse en cualquier capa en la pila de red.

Para Santos y Gregg (2019), los IDS se pueden dividir en dos categorías amplias: IDS basados en red (NIDS) e IDS basados en host (HIDS). Ambos tipos de sistemas se pueden configurar para monitorear ataques, rastrear los movimientos de un pirata informático o alertar a un administrador sobre ataques en curso. La mayoría de los IDS constan de más de una aplicación o dispositivo de hardware. Además, un IDS debe estar capacitado para buscar actividad sospechosa. La Figura 2.1 detalla la relación entre los IDS y los tipos de respuestas que pueden producir.

	VERDADEDO	**FALSO**
POSITIVO	Verdadero-Positivo Se generó una alarma, y una condición presente debería ser una alarma.	Falso-Positivo Se generó una alarma, y no se presentó ninguna condición para generarla.
NEGATIVO	**VERDADEDO**	**FALSO**
	Verdadero-Negativo no se generó una alarma, y no hay ninguna condición presente que deba alarmarse.	Falso-Negativo No se generó una alarma, y se presentó una condición que debería alarmarse.

Figura 2.2. Matriz IDS Verdadero/falso (Santos y Gregg, 2019).

Para Vacca (2012), el Sistema de Detección de Intrusos (IDS) recopila información de una variedad de recursos del sistema y de la red, pero en realidad captura paquetes de datos según lo definido por la pila de protocolos TCP/IP. El IDS en su función de sniffer captura todos los paquetes de datos o selecciona los especificados por el script de configuración. Este script

de configuración es un conjunto de reglas que le dice al analizador qué buscar en un paquete de datos capturados, luego hace una suposición informada por reglas y genera una alerta.

El Sistema de Detección de Intrusos (IDS) funciona mediante el examen de paquetes de datos en busca de indicadores de compromiso. Los indicadores se combinan con las directivas específicas de la plataforma IDS para formar reglas que instruyen al IDS sobre cómo ubicar eficientemente los indicadores dentro de los datos de la red. Cada vez que un IDS basado en firmas localiza datos que coinciden con el contenido encontrado en una firma, genera datos de alerta para notificar a los analistas. Los IDS basados en firmas se usan de manera efectiva para la detección de malware, asimismo para detectar malware relacionado con preocupaciones específicas y actuales (Smith y Sanders, 2013).

Según Rehman (2003), el Sistema de Detección de Intrusos (IDS) es un software, hardware o combinación de ambos que se utiliza para detectar la actividad de intrusos. Un IDS puede tener diferentes capacidades dependiendo de cuán complejos y sofisticados sean los componentes. Los dispositivos IDS que son una combinación de hardware y software están disponibles en muchas compañías, un IDS puede usar firmas, técnicas basadas en anomalías o ambas.

Orebaugh, Biles y Babbin (2009), El IDS genera registros y alertas, entonces, uno de los mayores problemas con la administración de una implementación de IDS es manejar el número potencialmente grande de alertas y registros. Si está configurado en una red pública que recibe mucho tráfico, podría ver miles de alertas por día, desde escaneos de script kiddy hasta gusanos y otras vulnerabilidades.

Según Baker y Esler (2005) el IDS a menudo almacena una base de datos de ataques conocidos. firmas y puede comparar patrones de actividad, tráfico o comportamiento que ve en los datos está monitoreando esas firmas para reconocer cuándo una coincidencia estrecha entre un se produce la firma y el comportamiento actual o reciente. En ese momento, el IDS puede emitir alarmas o alertas, tome varios tipos de acciones automatizadas que van desde apagar, enlaces de Internet o servidores específicos para iniciar trazas inversas y activar otros intenta identificar a los atacantes.

Para CERTSI (2017) existen diferentes clasificaciones de IDS según el enfoque, origen de los datos, estructura y comportamiento.

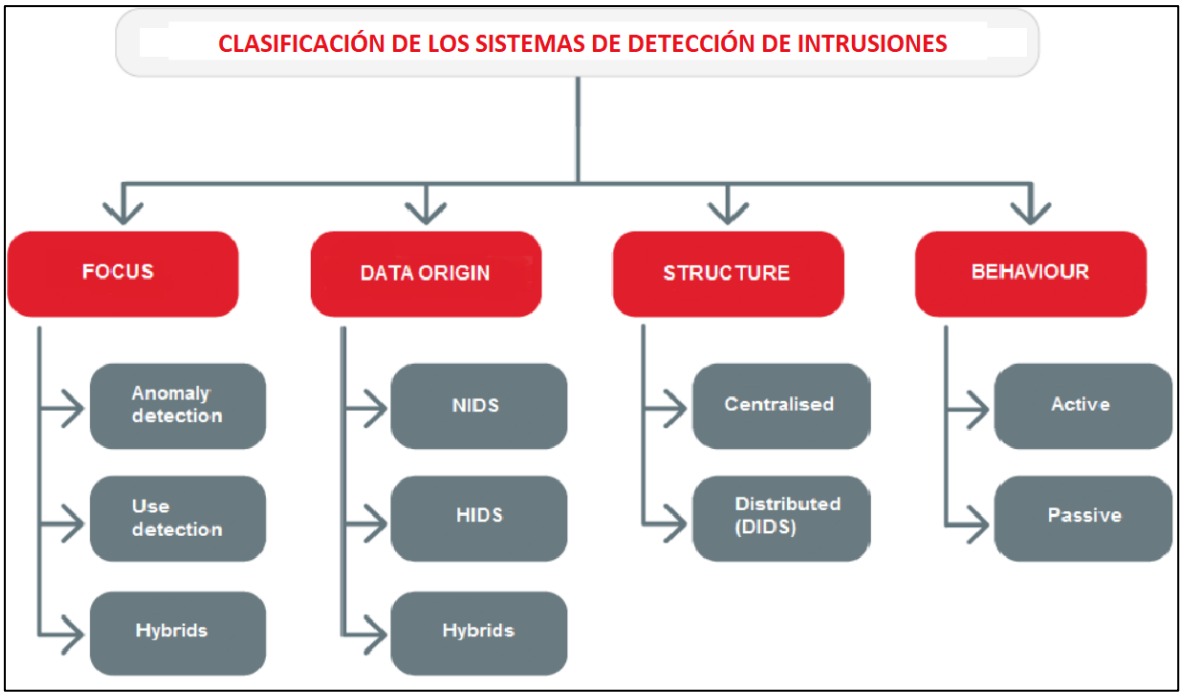


Figura 2.3. Clasificación de Sistema de Detección de Intrusos (CERTSI, 2017).

2.2.3. ATAQUES AL SISTEMA DE DETECCIÓN DE INTRUSIONES (IDS)

Para Chakrabarti, Chakraborty y Mukhopadhyay (2010), hay muchos tipos diferentes de ataques que pueden dañar un sistema. Por lo general, estos ataques se pueden agrupar en las siguientes categorías:

a) Confidencialidad: permite al atacante obtener acceso a la información sin autorización.
b) Integridad: permite que un atacante no autorizado afecte el estado del sistema. Esto podría significar afectar el estado del sistema o cualquier dato que resida o pase a través del sistema. Disponibilidad: se viola el principio de disponibilidad si el atacante puede evitar que un usuario autorizado acceda a un recurso del sistema.
c) Control: el ataque otorga a un atacante no autorizado un privilegio en violación de la política de control de acceso del sistema, el ataque está en el principio de control. Este ataque puede proporcionar medios para futuros ataques a la confidencialidad, integridad y / o disponibilidad.

2.2.3.1 ATAQUES DE ESCANEO

Los ataques de escaneo se pueden utilizar para asimilar información sobre el sistema atacado. Usando técnicas de escaneo, el atacante puede obtener información de topología, tipos de tráfico de red permitidos a través de un firewall, hosts activos en una red, SO y

kernel de hosts en una red, software de servidor en ejecución, números de versión del software, etc. En formación, el atacante puede lanzar ataques dirigidos a exploits más específicos. Lo anterior se recopiló mediante el lanzamiento de un escaneo SYN sigiloso. Este escaneo se llama sigilo porque en realidad nunca completa las conexiones TCP. Esta técnica a menudo se denomina medio análisis abierto, porque el atacante no abre una conexión TCP completa. El atacante envía un paquete SYN, como si estuviera abriendo una conexión TCP real. Si el atacante recibe un SYN/ACK, esto indica que el puerto está escuchando. Si no se recibe respuesta, el atacante puede asumir que el puerto está no abierto. Este es solo un tipo de técnica de escaneo, hay muchas más disponibles (Chakrabarti, Chakraborty y Mukhopadhyay, 2010).

2.2.3.2 ATAQUES DE DENEGACIÓN DE SERVICIO

Hay dos tipos principales de ataques de denegación de servicio (DoS): inundaciones y explotaciones de fallas. Los ataques de inundación a menudo se pueden implementar de manera muy simple. Por ejemplo, uno puede lanzar un ataque DoS simplemente usando el comando ping: ping -f victim. Esto resultará en enviar a la víctima una cantidad abrumadora de paquetes de ping. Si el atacante tiene acceso a un ancho de banda mayor que la víctima, esto abrumará a la víctima fácil y rápidamente. Como otro ejemplo, un ataque de inundación SYN envía una avalancha de paquetes TCP/SYN con una dirección de origen falsificada a una víctima. Esto hará que la víctima abra conexiones TCP medio abiertas; la víctima enviará un paquete TCP SYN/ACK y esperará un ACK en respuesta. Dado que el ACK nunca llega, la víctima eventualmente agotará los recursos disponibles esperando los ACK de un host inexistente (Chakrabarti, Chakraborty y Mukhopadhyay, 2010).

2.2.3.3 ATAQUES DE PENETRACIÓN

Los ataques de penetración contienen todos los ataques que le dan al atacante no autorizado la capacidad de obtener acceso a los recursos, privilegios o datos del sistema. Una forma común de que esto suceda es aprovechando una falla de software. Por ejemplo, en julio de 2002 se encontró un exploit en el código de manejo de respuesta de desafío sshd que permitía al atacante ejecutar código arbitrario como el usuario que ejecuta sshd (a menudo root). Este ataque se consideraría un ataque de penetración. Ser capaz de ejecutar código arbitrariamente como root le da al atacante cualquier recurso imaginable del sistema. Además, esto podría permitir al usuario lanzar otros tipos de ataque en este sistema, o incluso atacar otros sistemas desde el sistema comprometido (Chakrabarti, Chakraborty y Mukhopadhyay, 2010).

2.2.4 METODOS DE ANALISIS DE SISTEMA DE DETECCIÓN DE INTRUSIONES (IDS)

2.2.4.1 IDS BASADO EN FIRMA

Esta metodología se utiliza para detectar ataques desconocidos que ya están predefinidos en forma de firma y se guardan. Cuando se envían datos a la red, primero van al servidor donde el servidor lo analiza en busca de contenido malicioso. Se compara el paquete de red de la base de datos de la firma que es ya está almacenado en la red y si algún paquete coincide, entonces descarta el paquete o si no hay coincidencia, envía el paquete a la red (Sharma, Kumar y Tasneem, 2018).

Según Clarke (2017), un Sistema de Detección de Intrusos (IDS) está basado en firma, el IDS tiene un archivo de firma que enumera lo que se considera actividad sospechosa. Cuando el IDS captura la actividad, compara la actividad con la base de datos de firmas y, si hay una coincidencia, envía una notificación de una intrusión. El beneficio de un sistema basado en firmas es que tiene pocos falsos positivos, es decir, falsas alarmas mínimas, porque un sistema basado en firmas basa todo en las firmas que configuró en el sistema.

2.2.4.2 IDS BASADO PROTOCOLO

El NIDS identifica los elementos del protocolo y los analiza mientras busca una infracción. Algunos sistemas de detección de intrusos examinan campos de protocolo explícitos dentro de los paquetes inspeccionados. Otros requieren técnicas más sofisticadas, como el examen de la longitud de un campo dentro del protocolo o el número de argumentos. Por ejemplo, en SMTP, el dispositivo puede examinar comandos y campos específicos como HELO, MAIL, RCPT, DATA, RSET, NOOP y QUIT. Esta técnica reduce la posibilidad de encontrar falsos positivos si el protocolo que se analiza se define y aplica correctamente (Santos y Gregg, 2019).

2.2.4.3 IDS BASADO EN ANOMALIAS

Para Santos y Gregg (2019), el análisis basado en anomalías tiene limitación, hay que definir lo que se considera normal. Los sistemas y aplicaciones cuyo comportamiento puede considerarse fácilmente normal podrían clasificarse como sistemas basados en heurística. Sin embargo, a veces es difícil clasificar un comportamiento específico como normal o anormal en función de diferentes factores, que incluyen los siguientes:

a) Protocolos y puertos negociados

b) Cambios de aplicación específicos
c) Cambios en la arquitectura de la red

Estos tipos de ataques se utilizan para detectar los atacantes desconocidos, es decir, detecta un comportamiento que no se conocía antes. No es necesario escribir reglas para esta metodología. Detecta el tráfico malicioso basado en un patrón de tráfico de red normal. La desventaja de tal método es que genera alta tasa de alarma (Sharma, Kumar y Tasneem, 2018).

Según Clarke (2017), un IDS basado en anomalías comprende lo que se considera actividad normal (una línea de base) y luego considera que cualquier actividad fuera de esa actividad normal es actividad "sospechosa". El beneficio del sistema de anomalías basado en el comportamiento es que no necesita configurar un archivo de definición de actividad sospechosa conocida. El inconveniente de un sistema basado en el comportamiento es que cualquier cosa fuera de la norma se considera sospechosa, lo que da como resultado una gran cantidad de falsas alarmas (falsos positivos).

2.2.4.4 IDS BASADO EN ANALISIS HEURISTICO

El escaneo heurístico utiliza la lógica algorítmica del análisis estadístico del tráfico que pasa a través de la red. Sus tareas consumen mucha CPU y recursos, por lo que es una consideración importante al planificar su implementación. Los algoritmos basados en heurística pueden requerir un ajuste fino para adaptarse al tráfico de la red y minimizar laposibilidad de falsos positivos. Por ejemplo, una firma del sistema puede generar una alarma si se escanea un rango de puertos en un host o red en particular (Santos y Gregg, 2019).

Según Clarke (2017), el análisis heurístico es un tipo de análisis que identifica la actividad maliciosa en función de las reglas generadas por el proveedor que se almacenan en una base de datos y luego son utilizadas por el software de detección. Las reglas del proveedor se crean en base a experiencias pasadas con actividades maliciosas. l análisis heurístico es un enfoque popular con el software antivirus.

2.2.5 TIPOS DE SISTEMA DE DETECCIÓN DE INTRUSIONES (IDS)

2.2.5.1 SISTEMA DE DETECCIÓN DE INTRUSOS BASADO EN RED (NIDS).

Según Lane, Conklin, White y Williams (2019), un sistema basado en la red (NIDS) tendrá una "vista panorámica" de lo que está sucediendo en general en la red. Identifica los

intentos de intrusión al examinar el tráfico de la red, potencialmente mirando tanto el encabezado como el contenido de los paquetes que se transmiten. Puede monitorear el tráfico entrante y saliente. El monitoreo del tráfico saliente también puede proporcionar pistas de que un sistema en la red ha sido penetrado porque puede detectar signos de actividad, como la descarga inusual de archivos grandes o confidenciales.

Según Parisi (2019), la tarea típica de los sistemas de detección de intrusos en la red (NIDS) es identificar posibles patrones de ataque mediante el análisis del tráfico de la red; es decir, procesando paquetes de red en tránsito, tanto entrantes como salientes, y detectando patrones de ataque conocidos en el flujo de datos. Algunos ataques típicamente detectados por NIDS son: Adware (resulta en descargas de anuncios no solicitados), Spyware (información confidencial transmitida hacia hosts remotos), Amenazas persistentes avanzadas (APT) (aprovechan fallas específicas de la organización o servicios mal configurados), Botnets (aprovecha los recursos de la red de la organización transformando hosts en máquinas zombies, ejecutando instrucciones remotas).

Para Prowse (2017), NIDS es un tipo de IDS que intenta detectar actividades de red maliciosas, como son: Escaneos de puertos y ataques DoS, al monitorear constantemente el tráfico de la red. También puede ser instrumental en la detección de máquinas no autorizadas, incluidos los equipos de escritorio, computadoras portátiles y dispositivos móviles no autorizados, así como también puntos de acceso no autorizados, servidores DHCP y rastreadores de red.

El Sistema de Detección de Intrusiones (IDS) basados en red La mayoría de los sistemas comerciales de detección de intrusos están basados en la red. Estos IDS detectan ataques al capturar y analizar paquetes de la red. Escuchar en un segmento de red o conmutador, uno basado en red IDS puede monitorear el tráfico de red que afecta a múltiples hosts que están conectado al segmento de red, protegiendo así esos hosts. Los IDS basados en red a menudo consisten en un conjunto de sensores de propósito único o hosts colocados en varios puntos de una red. Estas unidades monitorean la red tráfico, realizando un análisis local de ese tráfico y reportando ataques a una consola de gestión central. Como los sensores se limitan a ejecutar el IDS, pueden protegerse más fácilmente contra ataques, para que sea más Es difícil para un atacante determinar su presencia y ubicación (Bace y Mell, 2015).

Un sistema de detección de intrusiones en la red (NIDS) es una técnica común que se utiliza para analizar el tráfico en todas las capas de interconexión de sistemas abiertos, detectando la presencia de tráfico normal o actividades sospechosas. Para ser efectivo, un NIDS debe ver toda la red y debe colocarse en un punto apropiado de la red. En una red basada en concentrador, el NIDS se puede colocar en el concentrador y puede ver todo el tráfico, pero esto no es posible en una red conmutada donde no hay concentrador. En una red conmutada, se utiliza la duplicación o expansión de puertos para permitir una vista completa, pero esto genera una sobrecarga. El propio NIDS se ve afectado por ataques DoS y DDoS, similares a los realizados contra las puertas de enlace IP. El NIDS no puede detectar el tráfico de red cifrado (Bul'ajoul, James y Pannu, 2015).

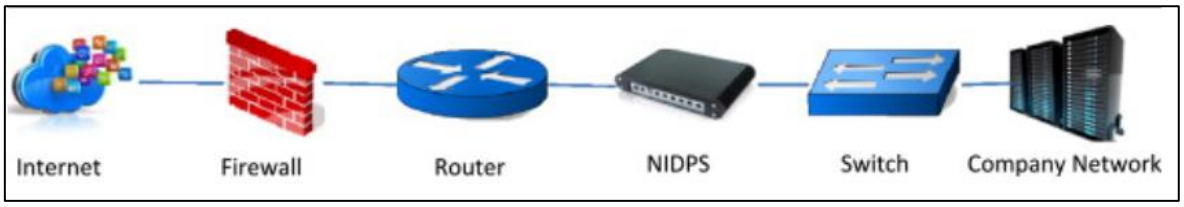


Figura 2.4. Un ejemplo de IDS basados en la red (Bul'ajoul, James y Pannu, 2015).

Para Vacca (2012), los NIDS auditan los paquetes de datos y registran cualquier paquete sospechoso en un archivo de registro, el nodo NIDS captura todos los paquetes de datos en la red según lo definido por el script de configuración.

Según Baker y Esler (2005) el NIDS monitorea un segmento de red completo. Normalmente, una tarjeta de interfaz de red de computadora (NIC) funciona en modo no promiscuo. En este modo de operación, solo los paquetes destinados a la dirección de control de acceso a medios (MAC) específica de la NIC (o paquetes de difusión) son reenviado la pila para su análisis. Los NIDS deben funcionar en modo promiscuo para supervisar el tráfico de red no destinado a su propia dirección MAC. NIDS puede espiar todas las comunicaciones en el segmento de red. Además, el NIDS debe estar conectado a un puerto span en su conmutador local, o una red toca duplicar el tráfico en el enlace que desea monitorear. Operación de La NIC del NIDS en modo promiscuo es necesaria para proteger su red.

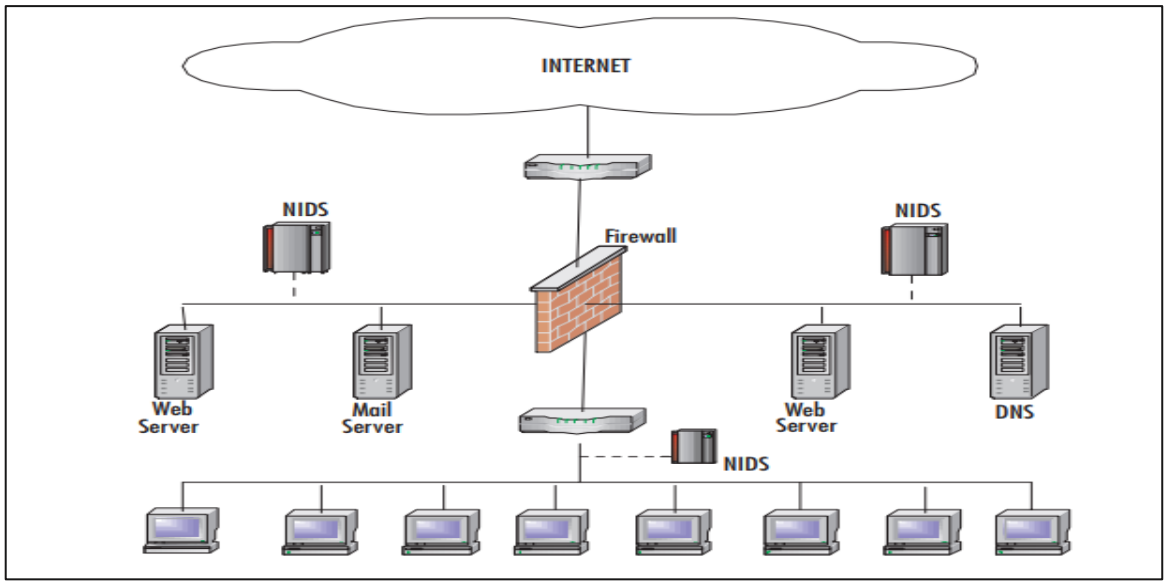


Figura 2.5. Red NIDS (Baker y Esler, 2005).

Según Rehman (2003), los NIDS son sistemas de detección de intrusos que capturan paquetes de datos que viajan por los medios de la red (cables, inalámbricos) y los relacionan con una base de datos de firmas. Dependiendo de si un paquete coincide con una firma de intrusos, se genera una alerta o el paquete se registra en un archivo o base de datos. Un uso importante de Snort es como NIDS.

Según Koziol (2003), con NIDS el analista de intrusiones tiene una visión gran angular de lo que está sucediendo dentro y alrededor de la red. El monitoreo de hosts o atacantes específicos se puede aumentar o disminuir con relativa facilidad. Puede ser más seguro y menos propenso a interrupciones que un HIDS. El NIDS debe ejecutarse en un único host reforzado que solo admita servicios relacionados con la detección de intrusiones, lo que dificulta su desactivación. Los NIDS pierden las desventajas de depender de la integridad y la disponibilidad del host monitoreado y, posteriormente, son menos propensos a interrupciones no observadas.

2.2.5.1.1 COMPONENTES DE UN IDS BASADO EN RED (NIDS)

Para Clarke (2017), el IDS basado en la red está compuesto por algunos componentes que trabajan juntos para identificar el tráfico sospechoso, como se muestra en la figura 2.6:

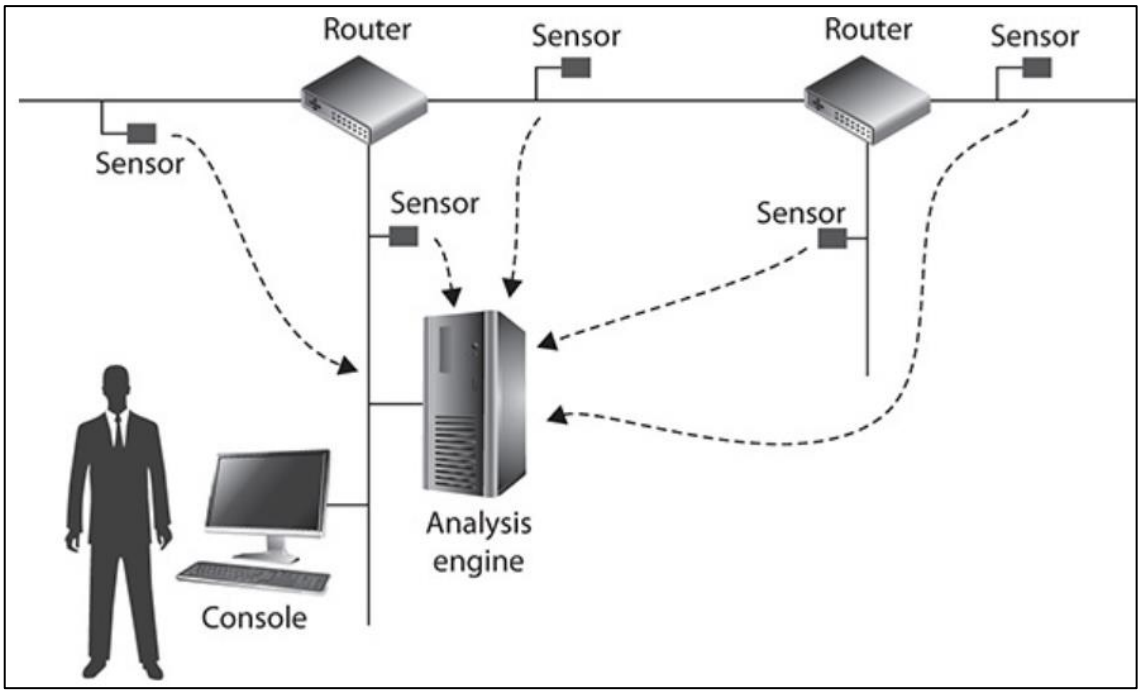


Figura 2.6. Componentes de un IDS basado en red (Clarke, 2017).

a) **Sensores:** Es una pieza de software o hardware que se coloca en cada segmento de red y es responsable de recolectar el tráfico de ese segmento y luego reenviar el tráfico al motor de análisis, o a un recolector si su infraestructura tiene uno. Un recolector es un componente que recolecta todo el tráfico de los sensores y lo envía al motor de análisis.
b) **Motor de análisis:** También conocido como motor de correlación, el motor de análisis es responsable de recibir los paquetes del sensor (o colector) y luego realizar el análisis de los paquetes para determinar si se consideran sospechosos.
c) **Consola:** Se conoce como la consola administrativa y es donde generalmente se envían las alertas y notificaciones. El administrador configura los NIDS desde la consola.

2.2.5.2 SISTEMA DE DETECCIÓN DE INTRUSOS BASADO EN HOST (HIDS).

Para Parisi (2019), la tarea de Host Intrusion Detection Systems (HIDS) es detectar posibles intrusiones que afectan a las máquinas host dentro de una organización, especialmente las máquinas que se consideran críticas. Con este fin, monitorea algunas de las métricas del sistema que se supone que son significativas para identificar posibles ataques.

Para Lane et al. (2019), un sistema basado en host se centra en lo que le está sucediendo a ese host en particular, tiene la posibilidad de detectar el ataque a nivel de host. Como tal, no notará lo que está sucediendo en la red en general.

Según Dulaney y Easttom (2017), los HIDS detectan actividad que indica una posible intrusión. Hay una variedad de tales sistemas, algunos comerciales y otros de código abierto. Estos sistemas tendrán algunos falsos positivos (tráfico legítimo etiquetado como ataque) y falsos negativos (ataques etiquetados como tráfico legítimo). La clave es interpretar lo que su HIDS le está diciendo correctamente para determinar si necesita alterar la configuración para obtener lecturas más precisas.

Los IDS basados en host funcionan con información recopilada dentro de un sistema informático individual. Este punto de vista permite a los IDS basados en host analizar actividades con gran confiabilidad y precisión, determinar exactamente qué procesos y usuarios están involucrados en un ataque particular al sistema operativo. Además, a diferencia de los IDS basados en red, los IDS basados en host pueden "ver" el resultado de un intento ataque, ya que pueden acceder directamente y monitorear los archivos de datos y el sistema procesos que suelen ser objeto de ataques (Bace y Mell, 2015).

Un Sistema de Detección de Detección a nivel de Host (HIDPS), es un agente de software que se puede instalar en una computadora en particular para monitorear y analizar eventos en ese host en particular para detectar cualquier comportamiento sospechoso. Los HIDPS son capaces de integrar análisis de código, monitorear llamadas al sistema, detectar desbordamientos de búfer, privilegios de uso indebido, privilegios abusivos, sistemas de archivos, listas de bibliotecas, aplicaciones, configuraciones del sistema, análisis de registros del sistema y muchos otros. Los HIDPS pueden hacer estas cosas porque están diseñados para operar con un host específico y con respecto a aplicaciones como servidores web, servidores de bases de datos, servidores de archivos, servidores de correo y servidores DNS. A menudo se integran en el software del servidor y se pueden implementar con relativa facilidad para comunicarse con otros componentes de red y sistemas operativos. Pueden inspeccionar el tráfico cifrado porque pueden analizar paquetes en los extremos de la aplicación. Sin embargo, los HIDPS tienen algunas desventajas (Bul'ajoul, James y Pannu, 2015).

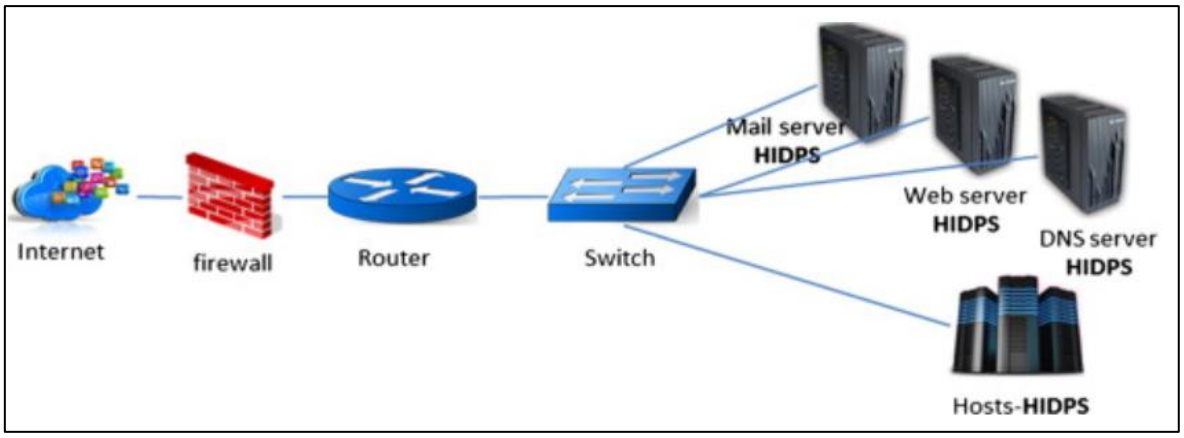


Figura 2.7. Un ejemplo de IDPS basado en host (Bul'ajoul, James y Pannu, 2015).

Según Baker y Esler (2005) el HIDS protege solo el sistema host en el que reside y su tarjeta de red funciona de manera predeterminada en modo no confidencial. El modo de operación no promiscuo puede ser una ventaja en algunos casos, porque no todas las NIC son capaces de modo promiscuo. Además, el modo promiscuo puede ser CPU intensiva para una máquina host lenta. Debido a su ubicación en el host para ser monitoreados, los HIDS tienen acceso a todo tipo de información local adicional con seguridad, otra ventaja de HIDS es la capacidad de adaptar el conjunto de reglas muy finamente para cada host individual.

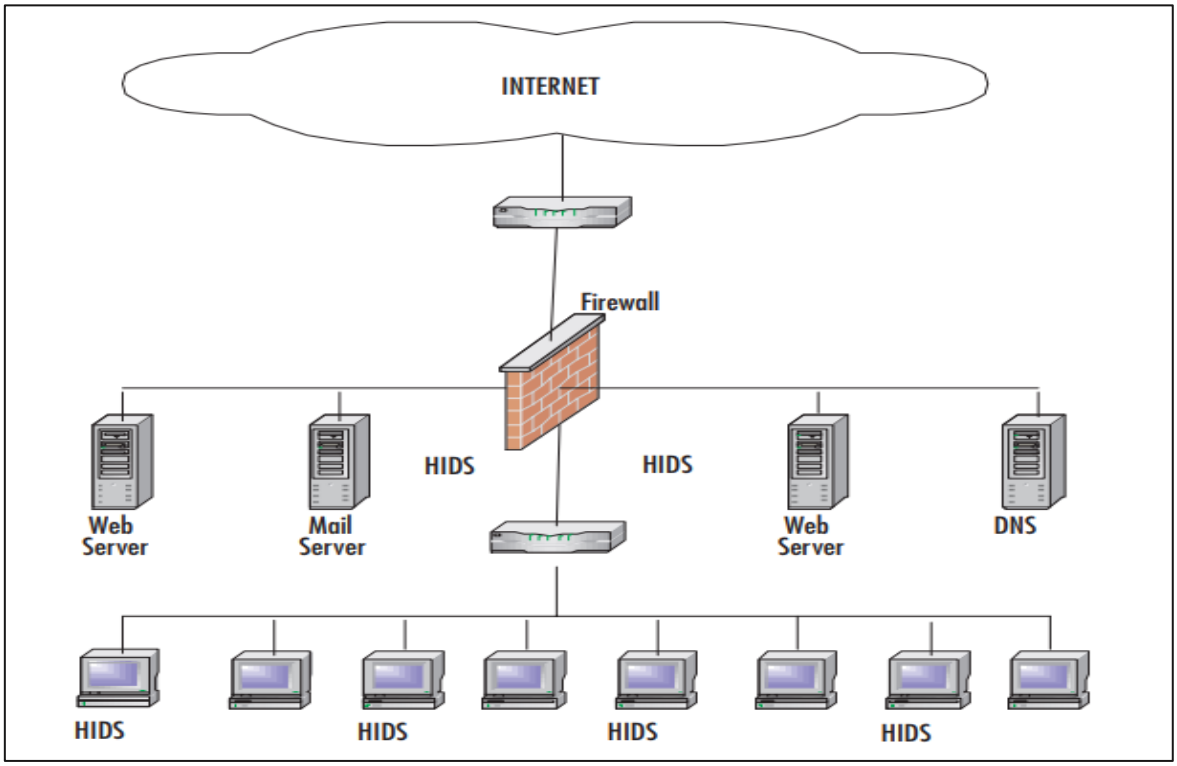


Figura 2.8. Una red mediana (Baker y Esler, 2005).

Según Koziol (2003), los HIDS supervisan los ataques a nivel del sistema operativo, la aplicación o el núcleo, tienen acceso a registros de auditoría, mensajes de error, derechos de servicio y aplicación, y cualquier recurso disponible para el host monitoreado. Los HIDS están en sintonía con el host en el que residen. Tienen un conocimiento profundo que está disponible solo para un IDS que realmente reside en la misma computadora que se está monitoreando. Por lo tanto, los HIDS pueden tener un conocimiento específico sobre el huésped y el tipo de actividad que es normal para él.

2.2.5.3 SISTEMA DE DETECCIÓN DE INTRUSOS DISTRIBUIDO (DIDS).

Según Baker y Esler (2005) el DIDS estándar funciona en una arquitectura Manager/Probe. Los sensores de detección de NIDS se ubican de forma remota e informan a una estación de administración centralizada. se cargan periódicamente a la estación de administración y se pueden almacenar en una central base de datos; se pueden descargar nuevas firmas de ataque a los sensores según sea necesario Las reglas para cada sensor se pueden adaptar para satisfacer sus necesidades individuales. se puede reenviar a un sistema de mensajería ubicado en la estación de administración y se usa para notificar al administrador de IDS.

La Figura 2.9 muestra un DIDS compuesto por cuatro sensores y una estación de administración centralizada. Los sensores NIDS 1 y NIDS 2 funcionan en sigilo promiscuo modo y están protegiendo los servidores públicos. Los sensores NIDS 3 y NIDS 4 protegen los sistemas host en la base informática confiable.

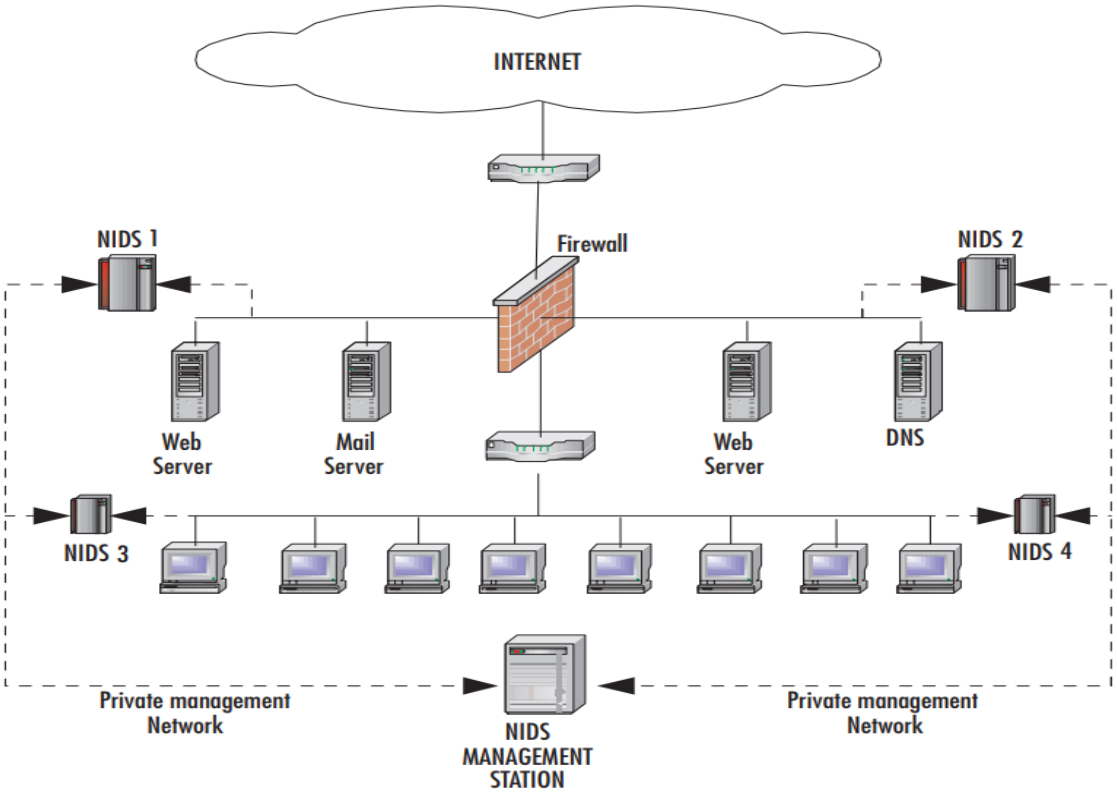


Figura 2.9. Red DIDS (Baker y Esler, 2005).

2.2.6. ¿DÓNDE COLOCAR UN IDS?

Para Arteaga (2020), un IDS puede ser colocado en 5 puntos diferentes de la red, como muestra la figura 2.10. De estos puntos el punto **B** (detrás de cortafuegos externo), ofrece varias ventajas al respecto como son:

1) Se monitorizan intrusiones que logran atravesar el firewall principal.

2) Detección de ataques a servidores que ofrecen servicios públicos.

3) En caso de no detectar ataques con éxito, puede reconocer algunas consecuencias de estos, como intentos de conexiones salientes, realizadas desde servidores comprometidos.

4) Identificación de los ataques y escaneos más comunes permite mejorar la configuración de los cortafuegos.

5) Monitoreo de la red LAN, ya que muchas amenazas son provocadas por usuarios internos al momento de ingresar a páginas web maliciosas o al insertar medios extraíbles contaminados en sus equipos informáticos.

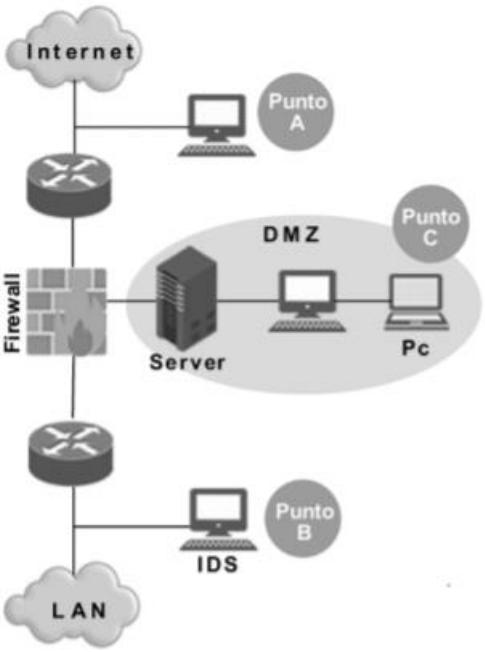


Figura 2.10. Colocación de un IDS (Arteaga, 2020).

Para Messier (2019), el lugar de colocación de un IDS de red es importante, puesto que vigila todo el tráfico de red que pasa por la interfaz de red. El mejor enfoque es colocar los IDS conectados en paralelo en lugar de en serie en el perímetro de la red para que se pueda observar todo el tráfico de red que ingresa, como se muestra en la Figura 2.11, donde muestra el IDS detrás del firewall, pero no directamente en el flujo de tráfico. El tráfico se desvía al IDS para que pueda detectarlo sin interponerse ni causar un fallo de red.

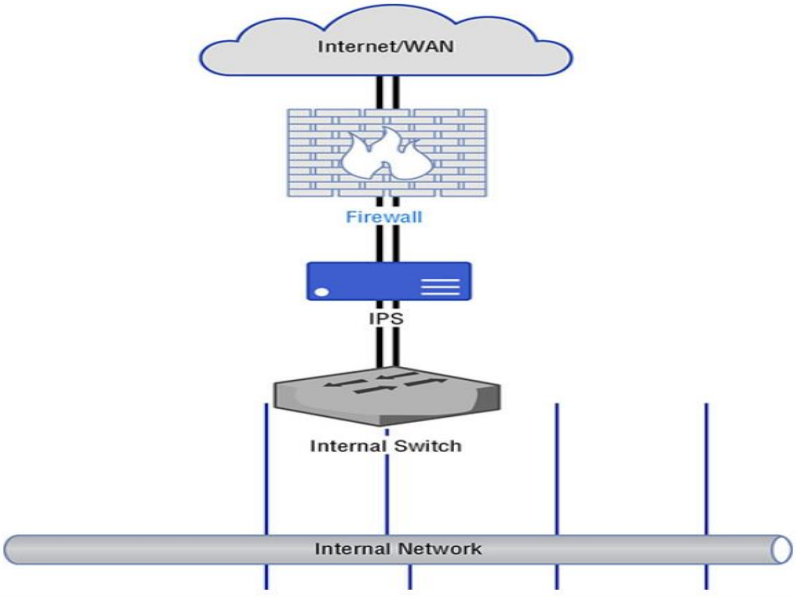


Figura 2.11. Diagrama de red que muestra la ubicación del IDS (Messier, 2019).

Para Blum y Bresnahan(2016), la clave de Snort es dónde coloca el servidor Snort. Si simplemente conecta su servidor Snort a un conmutador de red estándar como cualquier otro servidor, quedará algo decepcionado con los resultados que obtenga. Snort solo puede funcionar en los paquetes que ve, y el conmutador de red limitará esos paquetes a solo los paquetes destinados al servidor Snort. Para obtener todos los beneficios de Snort, conecte el servidor Snort a la red en una ubicación que vea todo el tráfico de red entrando y saliendo de su red local, como se muestra en la Figura 2.12.

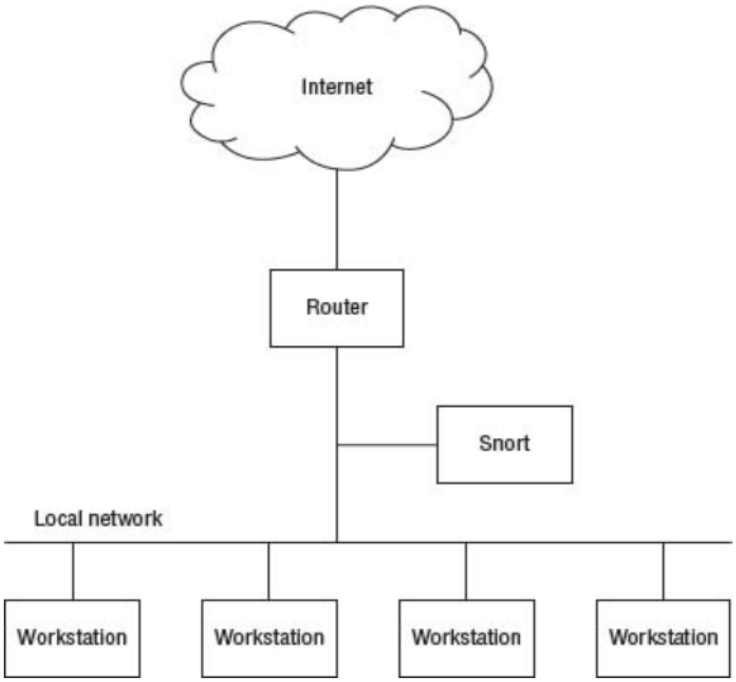


Figura 2.12. Colocación del servidor (Blum y Bresnahan, 2016)

Para Mira Alfaro (2002) existen principalmente tres zonas en las que podríamos poner un sensor, tal y como muestra la figura 2.13.

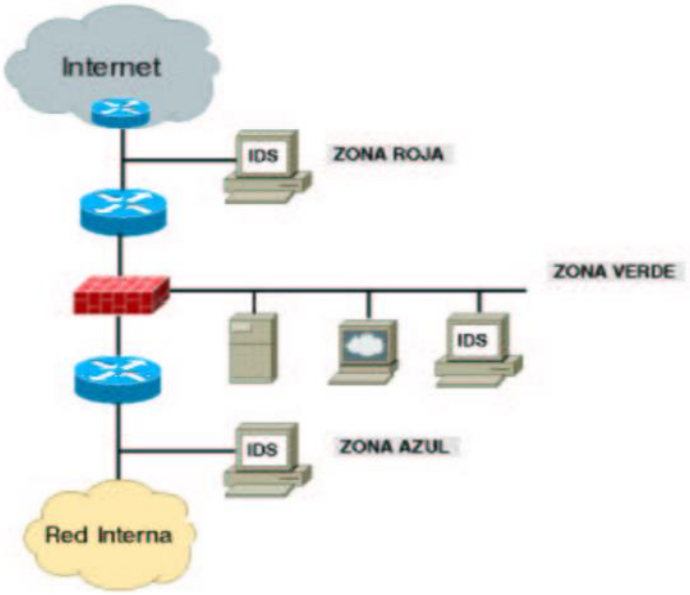


Figura 2.13. Localización de un IDS dentro de una organización (Mira Alfaro, 2002).

- **Zona roja:** Esta es de alto riesgo. En esta zona el IDS debe ser configurado para ser poco sensible, puesto que pueden existir muchos falsos positivos.
- **Zona verde:** En esta zona el IDS debería tener un poco más de sensibilidad ya que el firewall realiza un primer filtro por ende deben existir menos falsos positivos.
- **Zona azul:** Esta es la zona de confianza. El IDS debe tener una mayor sensibilidad y cualquier tipo de alarma que se genere debe ser inmediatamente revisada ya que en esta zona los falsos positivos deben ser muy pocos.

2.2.7. SNORT

Snort es una de las soluciones de prevención y detección de intrusos de código abierto y está disponible para descargar en snort.org. Snort hace que sea fácil para las entidades que comienzan dependiendo de la infraestructura implementada. También se puede escribir reglas personalizadas basadas en el entorno y el equipo que se necesita monitorear (Thompson, 2020).

Según Lane, Conklin, White y Williams (2019) Snort es un producto open-source que combina la inspección basada en firmas y anomalías. Aunque el programa es de código abierto, la base de datos de reglas en tiempo real más importante no lo es. Las reglas oficiales para Snort son generadas por el equipo de investigación de vulnerabilidad de Sourcefire y distribuidas por suscripción a sus clientes.

Para Goswami y Misra (2017) Snort funciona en redes IP realizando registros de paquetes en tiempo real, análisis de tráfico, análisis de protocolos y finalmente análisis de contenido. Por lo tanto, snort es adecuado también para su uso en sistemas de detección de intrusos. Cuando snort no se usa para la detección de intrusos, que es una tarea compleja, puede configurarse para usarse solo en modo de rastreo o de registro de paquetes.

Según Mandia, Luttgens y Pepe (2014) Snort se asemeja más al nivel de detalle al que puede estar acostumbrado a través de la supervisión de seguridad normal: alertas basadas en eventos. Cuando se configuran con conjuntos de reglas bien probados y ajustados, generarán eventos (o alertas) cuando se observen las condiciones predefinidas en la red monitoreada.

Para Costas Santos (2014) Snort es un IDS o Sistema de detección de intrusiones basado en red (NIDS). Implementa un motor de detección de ataques y barrido de puertos que permite registrar, alertar y responder ante cualquier anomalía previamente definida como patrones que corresponden a ataques, barridos, intentos aprovechar alguna vulnerabilidad, análisis de protocolos. Todo esto en tiempo real.

Para Baker, Beale y Caswell (2007) Snort tiene un sistema de detección de intrusos basado en red de código abierto (NIDS), pero también es un sniffer de paquetes rico en funciones y un capturador de paquetes, está basado en firmas que usa reglas para verificar si hay paquetes errantes en su red. Una regla es un conjunto de requisitos que desencadenaría una alerta. Admite el envío de alertas en tiempo real cuando se detecta un evento de intrusión.

Para Novak y Northcutt (2002) Snort es un NIDS basado en firmas que utiliza una combinación de reglas y preprocesadores para analizar el tráfico. Las reglas ofrecen un medio simple y flexible para crear firmas para examinar un solo paquete. El código del preprocesador permite un examen y manipulación más extensos de los datos que no se pueden hacer solo mediante reglas. Los preprocesadores pueden realizar una variedad de tareas, como la desfragmentación de IP, la detección de escaneo de puertos, la normalización del tráfico web y el reensamblado de flujo TCP, por nombrar algunos. Los preprocesadores le dan a Snort la capacidad de mirar y manipular flujos, en oposición al uso de reglas de vista de paquete único en un momento. Snort también es fácilmente configurable y flexible, lo que permite a los usuarios crear sus propias firmas y alterar la funcionalidad básica mediante el uso de complementos.

Para Scoot, Wolfe y Hayes (2004), es un IDS basado en la red que utiliza detección de firma; huele y examina la red paquetes de datos para contenido que coincide con ataques conocidos. Sus ventajas son:

a. **Snort es configurable:** Todo el funcionamiento interno de Snort, los archivos de configuración y las reglas están al descubierto para que pueda ajustar Snort a su arquitectura de red específica. No solo eso, sino que puedes crear tus propias reglas para nuevos ataques.
b. **Snort es gratis**: Snort se lanza bajo la GNU GPL, lo que significa que puedes úsalo gratis, sin importar si eres una empresa o simplemente un aficionado curioso.
c. **Snort se ejecuta en múltiples plataformas:** Snort no solo se ejecuta en todos los principales Sistemas operativos Unix y Unix-ish (incluido Linux), pero también se ejecuta en microsoft Windows.
d. **Snort se actualiza constantemente**: Lanzamientos de mantenimiento para Snort según sea necesario, generalmente una vez cada poco mes. Las reglas de Snort se actualizan regularmente con nuevas firmas de ataque y se pueden descargar desde www.snort.org/.

2.2.7.1. COMPONENTES DE SNORT

Según Rehman (2003), Snort está lógicamente dividido en múltiples componentes. Estos componentes trabajan juntos para detectar ataques particulares y generar resultados en un formato requerido del sistema de detección. Un IDS basado en Snort consta de los siguientes componentes principales:

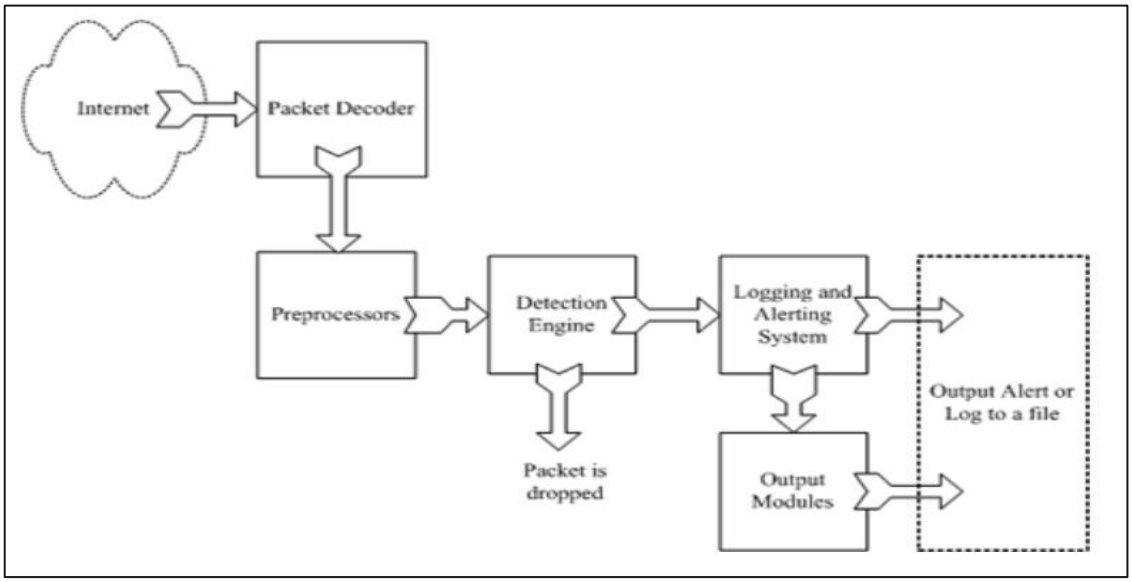


Figura 2.14. Componentes de Snort (Rehman, 2003).

A. PACKET DECODER (Decodificador de Paquetes)

El decodificador de paquetes recopila paquetes de diferentes redes, 2 interfaces y luego enviar para ser preprocesador o enviado al motor de detección. La interfaz de red puede ser Ethernet (Prakash y Kumar, 2012).

Scoot, Wolfe y Hayes (2004) indica que el decodificador de paquetes toma los datos de la capa 2 enviados de la librería de captura de paquetes y lo desmonta. Primero decodifica el marco de enlace de datos (Ethernet, TokenRing o 802.11), luego el protocolo IP, luego el paquete TCP o UDP. Cuando termine de decodificar, Snort tiene toda la información de los protocolos en todos los lugares correctos para su posterior procesamiento.

Para Rehman (2003), el decodificador de paquetes toma los paquetes de diferentes tipos de interfaces de red y prepara los paquetes para ser preprocesados o para ser enviados al motor de detección.

Tabla 2.1

Posibles decodificadores

Descifrador	Descripción
DecodeIptablesPkt	Este decodificador se utiliza para decodificar paquetes Iptables en modo en línea. Básicamente es un contenedor alrededor de la función DecodeIP ().
DecodeIpfwPkt	Este decodificador se usa para decodificar paquetes desde el firewall del Protocolo de Internet (IPFW).
DecodeEthPkt	Este decodificador verifica el campo *ether_type* del encabezado Ethernet y llama a los decodificadores de paquetes apropiados para desglosar aún más el paquete.
DecodeIEIE80211Pkt	Este decodificador examina los paquetes de la red de área local inalámbrica (WLAN) 802.11.
DecodeTRPkt	Este decodificador se usa para paquetes Token Ring. Verifica los campos de encabezado de acuerdo con la Solicitud de comentarios (RFC) 1042. Luego se pasa a varios otros decodificadores como *DecodeVlan ()* o *DecodeIP (),* dependiendo del campo de encabezado de tipo Ethernet.
DecodeFDDIPkt	Este decodificador se utiliza para decodificar paquetes de interfaz de datos distribuidos de fibra (FDDI).

Descifrador	Descripción
DecodeChdlcPkt	Este decodificador se utiliza para decodificar paquetes encapsulados de control de enlace de datos de alto nivel (HDLC). Prueba el tamaño del paquete y los diversos campos HDLC antes de pasar a la función *DecodeIP ()*.
DecodeSlipPkt	Este decodificador se usa para decodificar el tráfico SLIP. Se realiza una prueba para determinar si el tamaño del paquete es mayor o igual que el tamaño de un encabezado SLIP, antes de pasar el paquete a la función *ParseIP ()*.

Fuente: Rathaus, Ramirez, Caswell y Beale (2005, p.154).

B. PREPROCESSOR (Preprocesadores)

Funciona con Snort para modificar u organizar el paquete antes de llegar al motor de detección para aplicar alguna operación en el paquete. A veces también generan alertas si existen algunas anomalías encontradas en el paquete. Básicamente coincide con el patrón de toda la cadena, cambiando la secuencia o agregando algún intruso de valor extra puede engañar al Sistema de Detección de intrusiones (IDS), pero el preprocesador reorganiza la cadena del Sistema de Detección de intrusiones (IDS) y puede detectar alguna intrusión en la cadena (Prakash y Kumar, 2012).

Para Scoot, Wolfe y Hayes (2004) el preprocesador de Snort tiene varios complementos que pueden ser encendido o apagado El preprocesamiento opera en los paquetes decodificados, realizando una variedad de transformaciones que facilitan los datos para que Snort pueda digerir. Los preprocesadores pueden alertar, clasificar o descartar un paquete antes enviándolo al motor de detección más intensivo en CPU.

Para Rehman (2003), los preprocesadores son componentes o complementos que se pueden usar con Snort para organizar o modifique los paquetes de datos antes de que el motor de detección realice alguna operación para averiguar si el paquete está siendo utilizado por un intruso. Algunos preprocesadores también realizan detección por encontrar anomalías en los encabezados de los paquetes y generar alertas. Los preprocesadores son muy importantes para cualquier IDS para preparar paquetes de datos para ser analizados contra las reglas en el motor de detección. Los hackers usan diferentes técnicas para engañar a un IDS de diferentes maneras.

Tabla 2.2

Preprocesadores Snort

Nombre del preprocesador	Función
Flow	Este preprocesador ayuda a mantener un registro de flujo de estado de los paquetes que pasan por el motor Snort.
Frag2	Este preprocesador detecta y vuelve a ensamblar paquetes fragmentados que intentan evitar la detección. También detecta un ataque de denegación de servicio (DoS) utilizando paquetes fragmentados a una alta velocidad.
stream4	Este preprocesador vuelve a ensamblar los paquetes TCP y los inspecciona para detectar intentos de ataques de evasión de IDS desde herramientas como snort o *stick* utilizando ataques sin estado. También detecta escaneos de puertos, problemas de estado con una sesión y registra información de la sesión.
stream4_reassemble	Reorganiza los paquetes en sesiones significativas para el motor de reglas Snort y para los preprocesadores cargados después de Snort. También puede especificar el reensamblado del lado del cliente, del lado del servidor o de ambos lados de una conexión a través de todos los puertos o un conjunto de puertos seleccionados.
Http_inspect	Este nuevo preprocesador maneja todo el tráfico HTTP para ayudar a acelerarlo hacia el motor de reglas. Tambien sirve para varios propósitos, tales como: normalización del tráfico HTTP; Perfiles de tráfico HTTP y normalización, posiblemente para cada servidor web de su organización; y la capacidad de detectar el uso de proxy.
rpc_decode	Es un decodificador de aplicaciones. Escucha los paquetes de protocolo RPC en ciertos puertos, y luego decodifica el tráfico en esos puertos a ASCII para ser devuelto al motor de reglas Snort para su comparación.
telnet_decode	Es un decodificador de aplicaciones. Decodifica todo el tráfico en varios puertos, incluido 23/tcp, y luego lo devuelve al motor Snort.
bo_decode	Este preprocesador detecta cuándo el popular programa de caballo de Troya Back Orifice está en uso en su red. Este troyano muy popular tiene su propio protocolo que Snort puede detectar rápidamente y pasar al motor de reglas para una inspección detallada para determinar los comandos en uso.
Flow-portscan	Este es el único preprocesador que tiene que tener el preprocesador de flujo habilitado para funcionar. Toma datos de flujo y encuentra los escaneos de puertos en esos datos.
Perfmonitor	Este es un nuevo preprocesador que genera información estadística sobre la carga bajo la que se encuentra Snort, la carga del sensor y varias mediciones de rendimiento de la red.

Fuente: Orebaugh, Biles y Babbin (2009, p.169).

C. DETECTION ENGINE (Motor de detección)

Su trabajo principal es encontrar salidas de actividad de intrusión en paquetes con la ayuda de reglas de Snort,y si las encuentra se aplica la regla apropiada de lo contrario descarta el paquete. Se necesita diferentes tiempos para responder diferentes paquetes y también depende del poder de la máquina y el número de reglas definidas en el sistema (Prakash y Kumar, 2012).

Para Scoot, Wolfe y Hayes (2004) el motor de detección es el corazón de Snort. Se necesita información del decodificador de paquetes y preprocesadores y opera en él en las capas de transporte y aplicación (Capas 4 y 5 del modelo OSI), comparando lo que está en el paquete con la información en sus reglas complemento de detección. Estas reglas contienen firmas para ataques.

Para Rehman (2003), el motor de detección se aplica reglas a los paquetes, es la parte de tiempo crítico de Snort. Dependiendo de cuán poderosa sea su máquina y cuántas reglas haya definido, puede tomar diferentes cantidades de tiempo para responder a diferentes paquetes.

Según Rathaus, Ramirez, Caswell y Beale (2005), la mayor parte de la capacidad de Snort para detectar ataques está incorporada en las reglas que se utilizan para construir el motor de detección. Es dentro del motor de detección que la mayoría de las decisiones se toman con respecto a qué alertas (si las hay) se deben generar para un paquete en particular.

D. LOGGING AND ALERTING SYSTEM (Sistema de registro y alerta)

Cualquier motor de detección que encuentre en el paquete, puede generar una alerta o utilizarse para registrar la actividad. Todos los archivos de registro se guardan por defecto en la carpeta */var/log/snort* y usando el comando *–l*, la ubicación se puede cambiar (Prakash y Kumar, 2012).

Para Rehman (2003), dependiendo de lo que el motor de detección encuentre dentro de un paquete, el paquete puede usarse para registrar la actividad o generar una alerta. Los registros se guardan en archivos de texto simples, archivos de estilo tcpdump u otra forma. Todos los archivos de registro se almacenan en la carpeta /var/log/snort de forma predeterminada. Puede usar las opciones de la línea de comando –l para modificar la ubicación de generación de registros y alertas.

E. OUTPUT MODULES (Módulos de salida)

Los módulos de salida o complementos guardan la salida generada por el sistema de registro y alerta de Snort dependiendo de cómo el usuario quiere la operación diferente. Principalmente controla las diferentes salidas debido al sistema de registro y alerta (Prakash y Kumar, 2012).

Para Scoot, Wolfe y Hayes (2004) cuando se activa un preprocesador o regla, se genera una alerta y registrado (opcionalmente junto con el paquete ofensivo). Snort apoya una variedad de complementos de salida, incluidos sus propios formatos de registro basados en texto y binarios, varias bases de datos y syslog.

Para Rehman (2003), los módulos de salida o complementos pueden realizar diferentes operaciones dependiendo de cómo desea guardar la salida generada por el sistema de registro y alerta de Snort. Básicamente estos módulos controlan el tipo de salida generada por el sistema de registro y alerta. los módulos de salida pueden hacer cosas como las siguientes:

a) Simplemente inicie sesión en el archivo /var/log/snort/ alertas o algún otro archivo
b) Envío de trampas SNMP
c) Envío de mensajes a la instalación de syslog

2.2.7.2. REGLAS DE SNORT

Para Woland, Kampanakis y Santos (2016), una regla de Snort tiene dos secciones: un encabezado y un cuerpo. El encabezado contiene la acción, el protocolo, las direcciones IP y las máscaras de red de origen y destino, así como el puerto de origen y destino. El cuerpo contiene palabras clave que definen los criterios para activar una alerta. Las palabras clave del cuerpo contienen mensajes de eventos, patrones en la carga útil que deben coincidir y especificaciones de las partes del paquete que deben coincidir.

Según Rathaus, Ramirez, Caswell y Beale (2005), detecta uno o varios tipos de actividad de intrusión y genera alertas dependiendo de las veces que una regla haya coincidido con el contenido de un paquete. Los casos en que la línea que se evalúa en la función ***ParseRule()*** pertenece a una regla típica de Snort, (***Pass, Alert, Log),*** se completa una estructura local a la función ***ParseRule()*** y se agrega a la lista de reglas. El ***ParseRule()*** función define una instancia local de la ***RuleTreeNode*** estructura llamada ***proto_node***. Luego llena esta estructura, dependiendo del tipo de regla que se analiza.

Se puede usar una regla para generar un mensaje de alerta, registrar un mensaje o, en términos de Snort, pasar el paquete de datos, es decir, soltarlo en silencio. Pasar y soltar son opuestos entre sí. Las reglas de Snort están escritas en una sintaxis comprensible. La mayoría de las reglas están escritas en una sola línea. Sin embargo, también puede extender las reglas a varias líneas utilizando un carácter de barra diagonal inversa al final de las líneas. Las reglas generalmente se colocan en un archivo de configuración, generalmente snort.conf. Se puede usar varios archivos al incluirlos en un archivo de configuración principal (Rehman, 2003).

2.2.7.3. SINTAXIS DE UNA REGLA SNORT

Para Rehman (2003) una regla puede detectar uno o varios tipos de actividad de intrusión, todas las reglas de Snort tienen dos partes lógicas: encabezado de regla y opciones de regla. El encabezado de la regla contiene información sobre qué acción toma una regla. También contiene criterios para hacer coincidir una regla con los paquetes de datos. La parte de opciones generalmente contiene un mensaje de alerta e información sobre qué parte del paquete debe usarse para generar el mensaje de alerta. La parte de opciones contiene criterios adicionales para hacer coincidir una regla con los paquetes de datos.

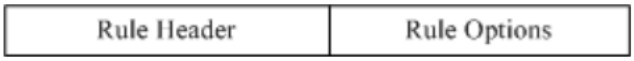


Figura 2.15. Estructura básica de las reglas de Snort. (Rehman, 2003).

El encabezado de la regla puede considerarse una breve descripción de la conexión de red. Cuatro parámetros definen una conexión de red única: IP de origen, puerto de origen, IP de destino y puerto de destino. El encabezado también incluye la dirección del recorrido del paquete, según lo definido por los símbolos ->o <>. Las opciones de regla definen lo que está involucrado en el paquete de red. Básicamente es un mensaje a Snort para inspeccionar el paquete en busca de valores coincidentes y determinar si se considera malicioso el paquete. Estas opciones se activan solo si los encabezados de las reglas coinciden con cierto contenido del paquete. Si hay una coincidencia, Snort generalmente escribe un mensaje de alerta en el archivo de alerta en el directorio de registro de Snort (Cox y Gerg, 2004).

Según De Haro Bermejo (2015), Snort posee una sintaxis propia que permite especificar hasta el más mínimo detalle las condiciones que han de cumplirse para que un paquete sea asociado a las acciones indicadas por cada una de las reglas.

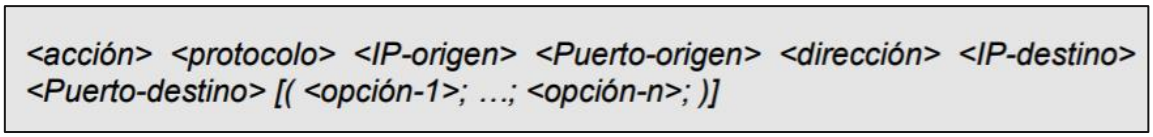


Figura 2.16. Estructura reglas Snort (De Haro Bermejo, 2015).

a. **Acción:** La acción definida en una regla de Snort puede ser elegida de entre las siguientes acciones:

Tabla 2.3

Posibles Acciones

alert	Genera una alerta usando el método elegido y luego registra el paquete en un fichero de log.
Log	Registra el paquete en un fichero de log
pass	Ignora el paquete.
activate	Genera una alerta y luego activa otra regla dinámica.
dynamic	Permanece desactivada hasta que se activa con una regla "activate", luego actúa como una regla "log".
drop	Bloquea y registra el paquete en un fichero de log.
reject	Bloquea el paquete, lo registra, y luego envía un TCP reset si es protocolo es TCP o un ICMP port unrecheable si el protocolo es UDP.
sdrop	Bloquea el paquete, pero no lo registra en un fichero de log.

Fuente: Adaptado de De Haro Bermejo (2015).

b. **Protocolo:** El siguiente campo en una regla es el protocolo. Hay cuatro protocolos que Snort analiza actualmente para un comportamiento sospechoso: TCP, UDP, ICMP y IP. En el futuro puede haber más, como ARP, IGRP, GRE, etc.

c. **Direcciones IP:** El siguiente campo hace referencia a la dirección IP de origen, y tras el operador de dirección, hace referencia a la IP de destino. La palabra clave "any" puede ser utilizada para definir cualquier dirección IP.

d. **Números de puerto:** Los números de puerto pueden especificarse de varias maneras, incluyendo algunos puestos, definiciones estáticas de puertos, rangos y por negación. La palabra clave "any" puede ser utilizada para definir cualquier número de puerto.

e. **Operador de dirección:** El operador de dirección indica la orientación, o dirección, del tráfico sobre el que se aplica la regla. Las opciones pueden ser "->" (origen -> destino) o "<>" (bidireccional).

f. **Opciones:** Todas las opciones de reglas de Snort se separan una de la otra usando el punto y coma (;). La palabra clave de la opción es separada de su valor por dos puntos (:). A continuación, se explican las cuatro categorías que existen:

Tabla 2.4

Categorías

general	Proporcionan información acerca de la regla, pero no tienen ningún efecto durante la detección.
payload	Buscan datos dentro de los paquetes.
non-payload	Buscan datos que no sean payloads.
post-detection	Descarta desencadenantes específicos que ocurren después de que una regla se ha disparado.

Fuente: Adaptado de De Haro Bermejo (2015).

Según Rathaus, Ramirez, Caswell y Beale (2005), hay muchas formas de actualizar sus reglas. La forma de hacerlo depende de si gestiona sus sensores con scripts o una interfaz gráfica de usuario o una interfaz web. Hay casi tantas formas de hacer esto como instalaciones de snort. En el nivel más básico, lo que necesita para administrar las reglas de snort es:

1. Una forma de extraer reglas de las fuentes en las que confía (snort.org, VRT, Bleeding Edge Threats).
2. Una herramienta para mostrarle las reglas nuevas y modificadas para su aprobación.
3. Una forma de incorporar sus reglas locales y volver a aplicar sus cambios locales a las reglas públicas.
4. Una forma de controlar qué reglas van a qué sensores.
5. Una forma de aplicar configuraciones específicas de snort a sensores específicos (vars, preprocesadores, interfaces, etc.).
6. Una forma de probar conjuntos de reglas antes de presionar a los sensores (para evitar matar un sensor con un error tipográfico).
7. Una forma de llevar esas reglas a cada sensor

2.2.7.4. FUNCIONAMIENTO DE SNORT

Snort tiene 3 usos principales:

a. **Un sniffer de paquetes**

Simplemente lee los paquetes fuera de la red y la muestra de forma continua. Una vez que finaliza el modo sniffer nos muestra una estadística del tráfico. Para iniciar el modo sniffer y visualizar en pantalla todo el tráfico TCP/IP se ejecuta. ***/snort -v*** (Snort.org, 2020).

Snort simplemente descarga los paquetes de red en la pantalla del terminal. (Blum y Bresnahan, 2016).

Para Rehman (2003) Snort captura y muestra paquetes de la red con diferentes niveles de detalle en la consola. La información que se muestra en la pantalla cuando ejecuta Snort en el modo de captura de paquetes es la que se muestra en la Figura 2.8, es una salida típica de un paquete TCP.

```
11/20-15:56:14.633891 192.168.1.2:22 -> 192.168.1.100:2474
TCP TTL:64 TOS:0x10 ID:57044 IpLen:20 DgmLen:184 DF
***AP*** Seq: 0xF5683D7A  Ack: 0x9DAEEE9C  Win: 0x6330  TcpLen: 20
```

Figura 2.17. Salida para un paquete TCP (Rehman, 2003).

Donde puede ver la información sobre el paquete donde indica:

1. Fecha y hora en que se capturó el paquete.
2. La dirección IP de origen es 192.168.1.2.
3. El número de puerto de origen es 22.
4. La dirección IP de destino es 192.168.1.100.
5. El puerto de destino es 2474.
6. El protocolo de capa de transporte utilizado en este paquete es TCP.
7. El tiempo de vida o el valor TTL en la parte del encabezado IP es 64.
8. El tipo de servicio o valor de TOS es 0x10.
9. La ID del paquete es 57044
10. La longitud del encabezado IP es 20.
11. La carga útil de IP tiene 184 bytes de longitud.
12. No fragmentar o el bit DF está configurado en el encabezado IP.
13. Dos banderas TCP A y P están encendidas.
14. El número de secuencia TCP es 0xF5683D7A.
15. El número de reconocimiento en el encabezado TCP es 0xDAEEE9C.
16. El campo de la ventana TCP es 0x6330.
17. La longitud del encabezado TCP es 20.

b. Un registrador de paquetes

Registra los paquetes en el disco. Para ejecutar snort en este modo debemos utilizar el parámetro ***–l /var/log/snort. Donde /var/log/snort*** es el directorio donde se registra el tráfico (Snort.org, 2020).

Snort registra todos los paquetes en un archivo de registro (Blum y Bresnahan, 2016).

c. Un NIDS

Realiza la detección y el análisis del tráfico de red. Este es el modo más complejo y configurable. Para que Snort funcione en modo IDS, debemos pasar el parámetro -c directorio hacia ***snort.conf.***, en el fichero de configuración snort.conf, se especifica la configuración deseada. Una vez realizada la configuración, el siguiente paso es hacer que snort se ejecute cada vez que arrancamos. Para ello, podemos realizarlo de distintas formas. Una de ellas es incluir el siguiente comando en el archivo de arranque: ***/etc/rc.d/rc.local*** (Snort.org, 2020).

Snort no almacena datos detallados de paquetes, solo informa sobre los intentos de intrusión detectados (Blum y Bresnahan, 2016).

Cuando Snort está en modo NIDS, aplica reglas en todos los paquetes capturados. Si un paquete coincide con una regla, solo entonces se registra o se genera una alerta. Si un paquete no coincide con ninguna regla, el paquete se descarta silenciosamente y no se crea ninguna entrada de registro. Cuando usa Snort en modo de detección de intrusos, generalmente proporciona un archivo de configuración en la línea de comandos. Este archivo de configuración contiene reglas de Snort o referencias a otros archivos que contienen reglas de Snort (Rehman, 2003).

2.2.8. UBUNTU 16.04

Una nueva instalación de Ubuntu está razonablemente completa. Las aplicaciones y herramientas que están presentes forman un sistema informático integral que se puede utilizar para todo el uso diario básico de la computadora. Pero los repositorios de software de Ubuntu albergan miles de paquetes de software más que pueden transformar su computadora en una herramienta poderosa para aumentar su productividad. Las posibilidades son casi infinitas (Haines, 2017).

Para Parziale, Novak, Acosta y Bader (2016), el software libre es parte de un fenómeno más amplio, que es un cambio hacia el reconocimiento del valor del trabajo compartido. Históricamente, las cosas compartidas tenían muy mala fama. La reputación era que la gente siempre abusaba de las cosas compartidas y, en el mundo físico, algo que se comparte y se abusa se vuelve inútil. En el mundo digital, creo que tenemos el efecto inverso, donde algo que se comparte puede volverse más valioso que algo que se guarda de cerca.

2.2.9. OPEN SOURCE

El código abierto es una forma de licenciar software. Significa que el software se puede distribuir libremente y contiene el código fuente. Esto significa que los usuarios pueden hacer copias, dárselas a amigos e incluso obtener una copia del código fuente. La idea detrás del código abierto es animar a los usuarios a examinar el código fuente de un producto y, si es posible, mejorarlo. La creencia es que, a través de la revisión y las mejoras realizadas por tantas personas, un producto alcanzará un mayor nivel de calidad más rápido que los comerciales (Easttom, 2014).

Según Gonzáles-Barahona (2011), el concepto de software libre es fundamentalmente legal: es un software con el que se pueden hacer cierto tipo de cosas, porque su autor da permiso para ello. tiene una visión bastante diferente sobre lo que los usuarios deberían poder hacer. Por eso, si recibes un programa libre, el autor te está permitiendo que:

a) Lo uses como mejor te parezca.
b) Puedas estudiar cómo funciona, y modificarlo si quieres.
c) Lo redistribuyas a quien quieras.
d) Distribuyas copias modificadas, si quieres.

Para **OPENBIZ** (2009), **Open Source Initiative** utiliza la Definición de Open Source para determinar si una licencia de software de computadora puede o no considerarse software abierto. La definición se basó en las directrices de software libre de Debian, fue escrita y adaptada primeramente por Bruce Perens. Bajo la Definición Open Source. Es similar pero no igual a la definición de licencia de Software Libre, las licencias deben cumplir diez condiciones para ser consideradas licencias de software abierto:

1. Libre redistribución: el software debe poder ser regalado o vendido libremente.

2. Código fuente: el código fuente debe estar incluido u obtenerse libremente.

3. Trabajos derivados: la redistribución de modificaciones debe estar permitida.

4. Integridad del código fuente del autor: las licencias pueden requerir que las modificaciones sean redistribuidas solo como parches.

5. Sin discriminación de personas o grupos: nadie puede dejarse fuera.

6. Sin discriminación de áreas de iniciativa: los usuarios comerciales no pueden ser excluidos.

7. Distribución de la licencia: deben aplicarse los mismos derechos a todo el que reciba el programa

8. La licencia no debe ser específica de un producto: el programa no puede licenciarse solo como parte de una distribución mayor.

9. La licencia no debe restringir otro software: la licencia no puede obligar a que algún otro software que sea distribuido con el software abierto deba también ser de código abierto.

10. La licencia debe ser tecnológicamente neutral: no debe requerirse la aceptación de la licencia por medio de un acceso por clic de ratón o de otra forma específica del medio de soporte del software.

2.2.10. ORACLE VM VIRTUALBOX

Para Dash (2013), Oracle VM VirtualBox es un paquete de software de virtualización multiplataforma de código abierto. El software de virtualización de escritorio le brinda la capacidad de instalar y ejecutar múltiples sistemas operativos en una computadora en un entorno virtual sin perturbar el sistema operativo host. VirtualBox crea un entorno virtual completamente aislado y, por lo tanto, evita que virus, malware o cualquier otro tipo de amenazas se propaguen desde la máquina del huésped al host.

Para Romero (2010), VirtualBox es gratuito y esto reduce sus costos iniciales para un centro de datos ágil. VirtualBox transformará la infraestructura de TI en un centro de datos delgado en una plataforma Windows XP / 7 o Ubuntu Linux. Aunque VirtualBox ha crecido a pasos agigantados, no hay suficiente documentación para guiarlo a través de sus características e implementación.

Para Jeff, Jeff, Gary y Bob (2017), VM VirtualBox es un motor de virtualización multiplataforma de alto rendimiento, se puede implementar en hardware de escritorio o servidor. Como hipervisor alojado, VirtualBox extiende el sistema operativo existente instalado en el hardware en lugar de reemplazarlo. VirtualBox incluye un hipervisor para la plataforma host, una interfaz de programación de aplicaciones (API) y un kit de desarrollo de software (SDK) para administrar máquinas virtuales invitadas (VM), una herramienta de línea de comandos para administrar invitados localmente, un servicio web para la administración remota de invitados, una herramienta gráfica de estilo asistente para administrar invitados, una consola gráfica para mostrar aplicaciones de invitados en el host local y un servidor de Protocolo de escritorio remoto (RDP) incorporado que proporciona acceso completo a un invitado desde un cliente remoto.

2.2.11. RED JERARQUICO

En un modelo jerárquico implica dividir la red en capas independientes. Cada capa (o nivel) en la jerarquía proporciona funciones específicas que definen su función dentro de la red general. Esto ayuda al diseñador y al arquitecto de red a optimizar y seleccionar las características, el hardware y el software de red adecuados para llevar a cabo las funciones específicas de esa capa de red. Los modelos jerárquicos se aplican al diseño de LAN y WAN. (Wikipedia, s.f).

Este modelo de red que se administra y expande con facilidad es el modelo jerárquico que se compone de 3 capas:

a) **Capa de acceso**: Permite el acceso a los ordenadores y dispositivos.
b) **Capa de distribución**: Conecta el acceso con el núcleo. En las redes pequeñas esta capa no existe dando lugar al modelo colapsado.
c) **Capa de núcleo:** La capa central es responsable de transportar grandes cantidades de tráfico de manera confiable y rápido

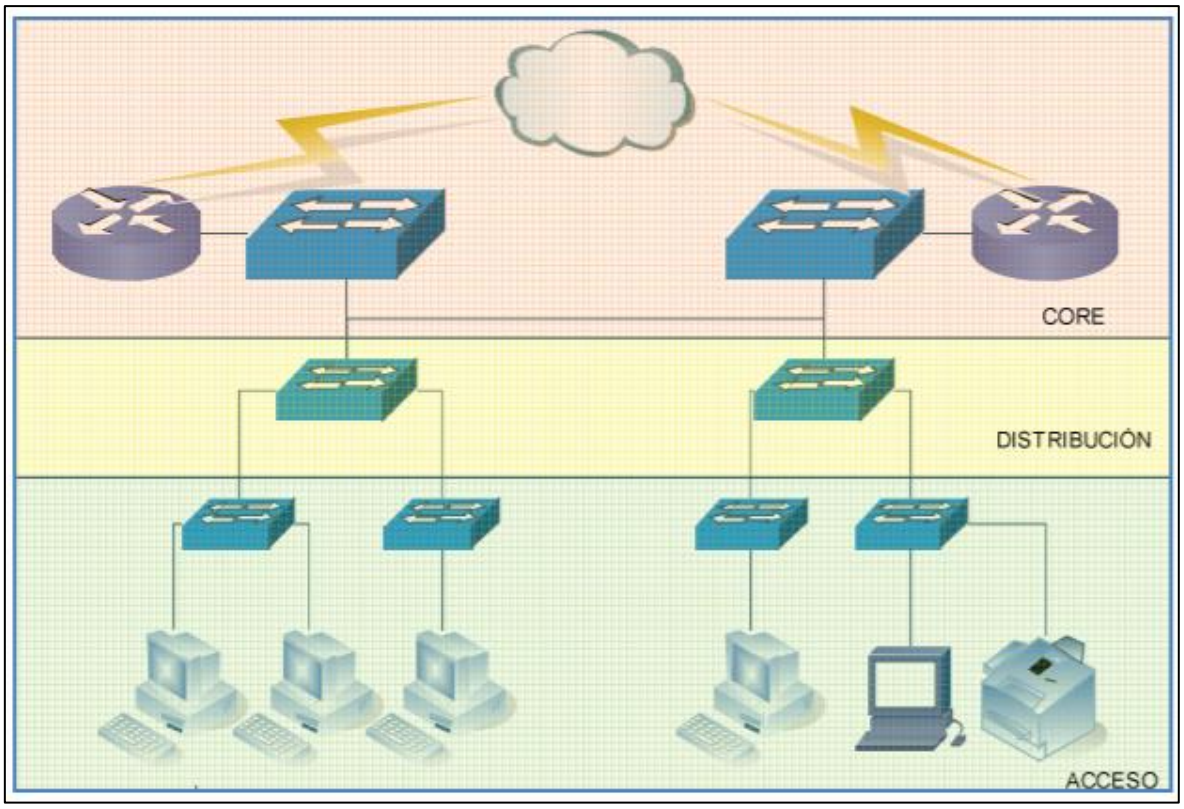


Figura 2.18. Red LAN Jerarquico (Tituaña, 2013).

La red LAN diseño jerárquico para desglosarlo en grupos modulares o capas. Este desglose del diseño en capas permite a cada capa implementar funciones específicas, lo que simplifica el diseño de red y, por lo tanto, la implementación y administración de la red. En arquitecturas de red mallada o plana, los cambios tienden a afectar a una gran cantidad de sistemas. El diseño jerárquico permite restringir los cambios operativos a un subgrupo de la red, lo que facilita la administración y mejora la recuperabilidad (CISCO, 2014).

Un diseño de red LAN jerárquico incluye las siguientes tres capas:
Capa de acceso: ofrece a los terminales y usuarios acceso directo a la red.
Capa de distribución: une las capas de acceso y ofrece conectividad a los servicios.
Capa central: ofrece conectividad entre las capas de distribución para entornos de LAN grandes.

Tabla 2.5

Beneficios de la Red Jerárquica.

CARACTERÍSTICA	DESCRIPCIÓN
Escalabilidad	Las redes jerárquicas pueden expandirse con facilidad.

Redundancia	La redundancia a nivel del núcleo y de la distribución aseguran la disponibilidad de la ruta.
Rendimiento	El agregado del enlace entre los niveles y núcleo de alto rendimiento y switches de nivel de distribución permite casi la velocidad del cable en toda la red.
Seguridad	La seguridad del puerto en el nivel de acceso y las políticas en el nivel de distribución hacen que la red sea más segura.
Facilidad de administración	La consistencia entre los switches hace que la administración sea más simple.
Facilidad de mantenimiento	El modularidad del diseño jerárquico permite que la red escale son volverse demasiado complicada.

Fuente: Adaptado de Tech Club Tajamar (2016).

2.2.12. PYMES

Las Pymes son pequeñas y medianas empresas, con alta predominancia en el mercado de comercio, quedando prácticamente excluidas del mercado industrial por las grandes inversiones necesarias y por las limitaciones que impone la legislación en cuanto al volumen de negocio y de personal (Wikipedia, s.f).

Para Arbulú (2006), la legislación peruana define como la unidad económica constituida por una persona natural o jurídica, bajo cualquier forma de organización o gestión empresarial contemplada en la legislación vigente, que tiene como objeto desarrollar actividades de extracción, transformación, producción, comercialización de bienes o prestación de servicios.

2.2.13. POBLACIÓN

Para Tomas (2009), la población "es el conjunto de todos los individuos que cumplen ciertas propiedades y de quienes ciertos datos. Podemos entender que una población abarco todo el conjunto de elementos de los cuales podemos obtener información, entendiendo que todos ellos han de poder ser identificados".

Según Tamayo y Tamayo (1997), La población se define como la totalidad del fenómeno a estudiar donde las unidades de población poseen una característica común la cual se estudia y da origen a los datos de la investigación.

Hernández (2001), menciona “La población o muestra se puede definir como un conjunto de unidades o ítems que comparten algunas notas o peculiaridades que se desean estudiar” (p.127).

Bernal (2006) define que la población es el conjunto total de individuos, objetos o medidas que poseen algunas características comunes observables en un lugar y en un momento determinado. Cuando se vaya a llevar a cabo alguna investigación debe de tenerse en cuenta algunas características esenciales al seleccionarse la población bajo estudio. Entre éstas tenemos:

a. Homogeneidad, que todos los miembros de la población tengan las mismas características según las variables que se vayan a considerar en el estudio o investigación.
b. Tiempo, se refiere al período de tiempo donde se ubicaría la población de interés. Determinar si el estudio es del momento presente o si se va a estudiar a una población de cinco años atrás o si se van a entrevistar personas de diferentes generaciones.
c. Espacio, se refiere al lugar donde se ubica la población de interés. Un estudio no puede ser muy abarcador y por falta de tiempo y recursos hay que limitarlo a un área o comunidad en específico.
d. Cantidad, se refiere al tamaño de la población. El tamaño de la población es sumamente importante porque ello determina o afecta al tamaño de la muestra que se vaya a seleccionar, además que la falta de recursos y tiempo también nos limita la extensión de la población que se vaya a investigar.

CAPITULO III
METODOLOGÍA DE LA INVESTIGACIÓN

3.1. TIPO Y NIVEL DE INVESTIGACIÓN

A. TIPO DE INVESTIGACIÓN

De acuerdo al autor Carrasco (2006), "Investigación aplicada se distingue por tener propósitos prácticos inmediatos definidos, es decir, se investiga para actuar, transformar, modificar o producir cambios en un determinado sector de la realidad. Para realizar investigaciones aplicadas es muy importante contar con el aporte de las teorías científicas, que son producidas por la investigación básica y sustantiva" (p.44). Por esta consideración el tipo de **investigación es aplicada.**

B. NIVEL DE INVESTIGACIÓN

De acuerdo al autor Hemández Sampieri Et. Al (2010), las investigaciones descriptivas vienen a ser "los estudios descriptivos buscan especificar las propiedades, las características y los perfiles importantes de personas, grupos, comunidades o cualquier otro fenómeno que se someta a análisis." De acuerdo al autor (2006), una de las funciones principales de la investigación descriptiva es la capacidad para seleccionar las características fundamentales del objeto de estudio y su descripción detallada de las partes, categorías o clases de dicho objeto"; y agrega "La investigación descriptiva es uno de los tipos o procedimientos investigativos más populares y utilizados por los principiantes en la actividad investigativa. Los trabajos de grado, en los pregrados y en muchas maestrías, son estudios de carácter eminentemente descriptivo. En tales estudios se muestran, narran, reseñan o identifican hechos, situaciones, rasgos característicos de un objeto de estudio, o se diseñan productos, modelos, prototipos, guías, etcétera (p.112).

De acuerdo al autor Carrasco (2006), señala que, la investigación descriptiva se soporta principalmente en técnicas como la encuesta, la entrevista, la observación y revisión documental. Este tipo de investigación estudia, analiza, describe y especifica situaciones y propiedades de personas, grupos, comunidades o cualquier otro fenómeno u objeto que sea sometido al análisis. Por esta consideración el nivel de **investigación es descriptivo.**

3.2. DISEÑO DE INVESTIGACIÓN

De acuerdo con el autor Hemández Sampieri Et. Al (2010), se puede definir la investigación no experimental, "como aquella investigación que se realiza sin manipular deliberadamente las variables, se trata de estudios donde no hacemos variar en forma intencional las variables independientes para ver su efecto sobre otras variables. Lo que hacemos en la investigación no experimental es observar fenómenos tal como se dan en su contexto natural, para posteriormente analizarlos" (p.191).

Según Carrasco (2008), el diseño de investigación transversal descriptivo, se empela para analizar y conocer las características, cualidades internas y externas, propiedades y rasgos esenciales de los hechos y fenómenos de un hecho realizado a momento determinado del tiempo. De acuerdo a Hernández et al 2010, una investigación de diseño transversal es "cuando se recolectan datos en un solo momento o en un tiempo único y su propósito es describir variables y analizar los hechos tal como se dan". Los instrumentos de recolección de datos, son usados durante el proceso de forma única. En esta investigación se tenía que analizar procesos, conocer características, estudiar rasgos y entender funcionalidades para encontrar datos necesarios para la construcción del aplicativo de inteligencia de negocios; por lo tanto, el diseño de investigación es **no experimental de tipo transversal descriptivo.**

3.3. HIPÓTESIS DE LA INVESTIGACIÓN

"No en todas las investigaciones cuantitativas se planean hipótesis. El hecho de formulemos o no hipótesis depende de un factor esencial: el alcance inicial del estudio. Las investigaciones cuantitativas que formulan hipótesis son aquellas cuyo planteamiento define que su alcance será correlacional o explicativo, o en las que tienen un alcance descriptivo, pero que intentan pronosticar una cifra o un hecho" (Hernández, baptista y Collado, 2014, p.104).

"Las investigaciones de tipo descriptivo no requieren formular hipótesis; es suficiente plantear algunas preguntas de investigación que, como ya se anotó, surgen del planteamiento del problema, de los objetivos y, por supuesto, del marco teórico que soporta el estudio" (Bernal, 2010, P.136). La investigación que se desarrolló es de tipo descriptivo, por lo que no se pretende pronosticar hallar o verificar lo planteado en los objetivos, se optó por no plantear hipótesis.

3.4. POBLACIÓN Y MUESTRA

A. POBLACIÓN

Estuvo compuesta por todos los sistemas de seguridad a nivel de paquetes de la arquitectura TCP/IP para detección de intrusos basado en red.

B. MUESTRA

Sistema de detección de intrusos Snort Open Source.

3.5. DEFINICIÓN CONCEPTUAL DE LAS VARIABLES

VARIABLE DE ESTUDIO 1

a. **Snort Open Source como sistema de detección de intrusos:** Es un IDS basado en la red que utiliza detección de firma; detecta ataques y barrido de puertos que permite registrar, alertar y responder ante cualquier anomalía previamente definida.

Dimensiones

a. **Interfaz de red:** Realiza la captura de paquetes y la detección, análisis del tráfico de red, modo más complejo y configurable.
b. **Método de captura:** Captura y muestra paquetes de la red con diferentes niveles de detalle en la consola.
c. **Motor de reglas y filtrado de eventos:** Detecta uno o varios tipos de actividad de intrusión y genera alertas dependiendo de las veces que una regla haya coincidido con el contenido de un paquete.

VARIABLES ESTUDIO 2

a. **Seguridad de la infraestructura de red en entornos libres:** Es la práctica de prevenir y proteger contra la intrusión no autorizada en redes.
b. **Pymes del Perú:** Son pequeñas y medianas empresas, con alta predominancia en el mercado de comercio, tienen la capacidad de convertirse en el motor del desarrollo empresarial.

3.6. DEFINICIÓN OPERACIONAL DE LAS VARIABLES

VARIABLE DE ESTUDIO 1

a. Snort Open Source

Dimensiones

a. Interfaz de red.
b. Método de captura.
c. Motor de reglas y filtrado de eventos.

VARIABLE DE ESTUDIO 2

a. Seguridad de la infraestructura de red.

Dimensión

a. Pymes del Perú.

3.7. TÉCNICAS E INSTRUMENTOS

A. TÉCNICAS

Análisis documentario.

CAPITULO IV

IMPLEMENTACION DEL SISTEMA DE DECTECCIÓN DE INTRUSOS (IDS) EN LA EMPRESA MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C.

4.1. ANTECEDENTES DE LA EMPRESA

La empresa MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C. inicia su operación hace más de 8 años fundada en la ciudad de Ayacucho en 2011 e inició sus actividades económicas brindando un servicio de transporte de carga a nivel nacional y venta al por mayor de materias primas agropecuarias. La empresa establecida es una Sociedad Anónima Cerrada, formada por tres socios fundadores. Estos tres miembros integran la junta directiva y tienen el mismo poder de decisión. Sin embargo, se ha denominado un gerente general quien es el representante a nivel funcional.

Posee una moderna flota y gran variedad de camiones que sus clientes requieren en soluciones de transporte, cuenta con operadores altamente capacitados en diferentes materias relacionadas al transporte de carga nacional permite ofrecer a sus clientes la seguridad de que su carga será transportada por profesionales aptos. Hoy en día la empresa MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C, es una organización confiable que busca brindar el mejor servicio, asegurando la calidad de su trabajo con el único objetivo de la total satisfacción de sus clientes. Su visión a futuro es consolidarse como la empresa de transporte más confiable y segura.

4.1.1. ESTRUCTURA ORGANIZACIONAL

La empresa MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C, está conformada por 5 áreas principales: Gerencia general, ventas, operaciones, contabilidad.

a) **Gerencia General:** Como se ha indicado, se denominó un gerente general, este cargo es representativo ya que los tres socios son los que deciden en la empresa previa coordinación. La persona que ocupa este cargo es responsable de velar por el correcto funcionamiento de la empresa y lograr los objetivos propuestos.
b) **Ventas:** Esta área está encargada de las funciones comerciales, ya que como se indicó también se dedica al rubro de compra y venta de materias primas agropecuarias. Las principales actividades que se realizan en esta área es la de buscar proveedores de las diferentes materias primas, posteriormente se realiza seguimiento a los pedidos

con ello a las materias primas, si cumple con la calidad establecida el proveedor pasa a la lista de proveedores de la empresa. Sin embargo, si no cumpliese con la calidad se realiza un plan de acción de mejora, y nuevamente se evalúa si cumple con la calidad establecida, si cumple se registra a la lista de proveedores caso contrario se rechaza y se busca otros proveedores.

Una vez que se cuente con la materia prima son vendidos a los diferentes mercados nacionales, siendo los principales los mercados mayoristas del departamento de Lima.

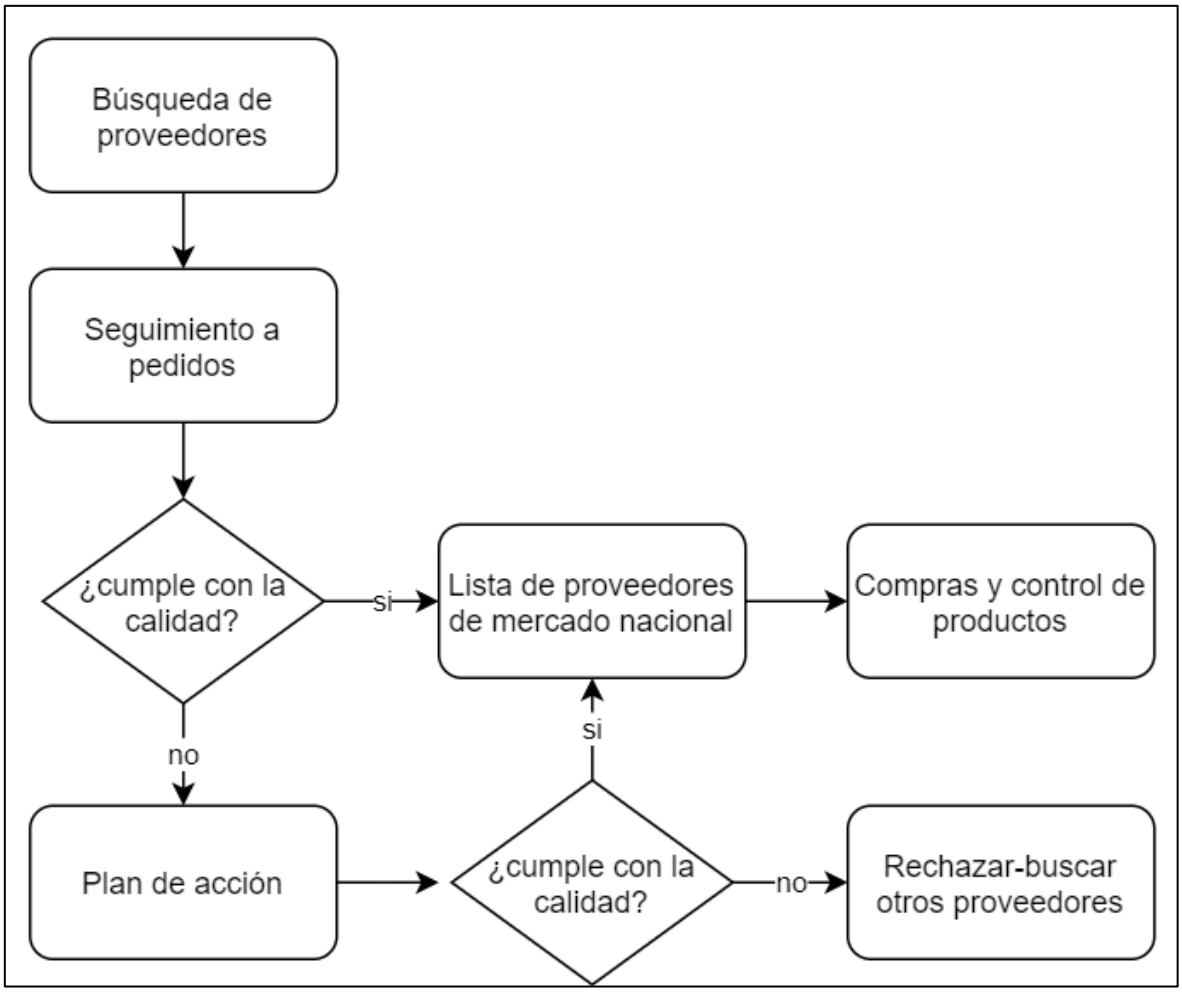


Figura 4.1. Proceso de búsqueda de proveedores y compra MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C

c) Almacén: En esta se almacena las materias primas que se compra, para su posterior venta a los diferentes mercados mayoristas a nivel local y nacional. Las materias primas que se almacena son cereales (cebada, trigo, maíz, haba, maíz morado, tara, quinua, papa seca, linaza, tarwi) y tubérculos (papa, olluco).

d) **Operaciones:** El área de operaciones es crucial para el funcionamiento de la empresa en el rubro de transporte. La principal función que se realiza en esta área se encarga del mantenimiento correctivo y preventivo de las unidades de transporte de carga.

Contabilidad: Esta área es muy importante en toda la empresa ya que se realizan todos los procedimientos relacionados con información financiera que debe registrar su exactitud. Registrando y controlando los gastos e ingresos y las diversas operaciones económicas que realiza la empresa en sus diferentes actividades.

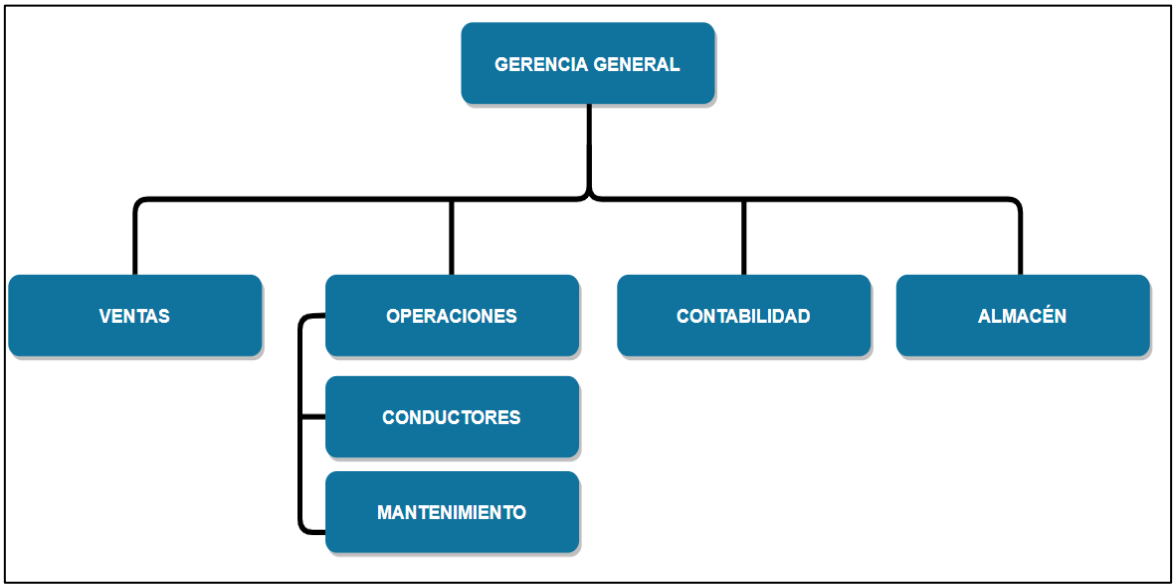


Figura 4.2. Organigrama MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C

4.1.2 SITUACIÓN ACTUAL DE INFRAESTRUTURA DE RED

La empresa MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C cuenta con los siguientes dispositivos:

Tabla 4.1

Dispositivos y características de Multinegocios & Transp. D.S Huamani S.A.C

DISPOSITIVO	CARACTERÍSTICA
Router Cisco EPC3825	El Cisco EPC3825 es un router con WiFi y compatible con DOCSIS 3.0, soporta 30, 50 y 100 megas. Además, tiene 8 canales de bajada hasta 440 Mbps y 4 de subida, hasta 120 Mbps. Cuenta con cuatro puertos Ethernet 10/100/1000 y WiFi 802.11n de hasta 300 Mbps (wiki.bandaancha, s.f).
PC-01(contabilidad).	Marca: Dell 4ta Generación.

	Modelo: Optiplex 9020. Procesador: Intel Core i7. Disco duro: 500 GB. RAM 4 GB. Sistema Operativo: Windows 10.
PC-02 (Gerencia)	Marca: Lenovo Modelo: Thinkcentre M73 Procesador: Intel Core i5 Disco duro: 500 GB RAM 4 GB Sistema Operativo: Windows 8
Impresora Ecotank L4160	Modelo L4160. Conexión Wifi. Escanea y fotocopiadora doble cara. Resolución de escáner 48 bits y 1200 x 2400 dpi.

La estructura de red de la empresa Multinegocios & Transp. D.S Huamani S.A.C, como se muestra en la Figura 4.3, el internet llega a través del router, esta red cuenta con 2 ordenadores conectados al router a través de cableado UTP, asimismo la impresora está conectada a través del Wifi.

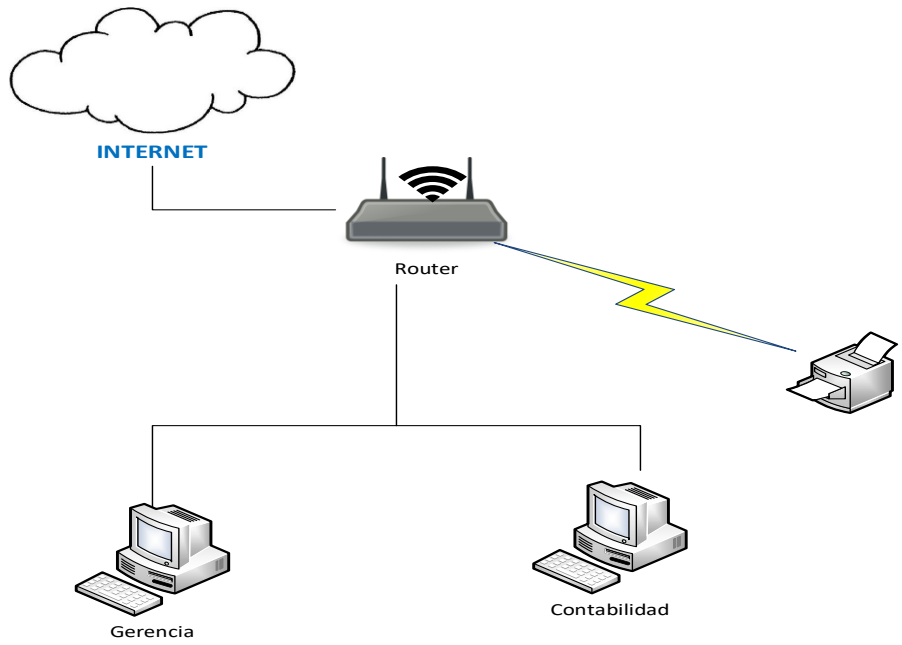


Figura 4.3. Topología red de la empresa Multinegocios & Transp. D.S Huamani S.A.C

4.2. DISEÑO DE LA ESTRUCTURA DE RED BASADA EN SISTEMA DE DETECCIÓN DE INTRUSOS A NIVEL DE RED (NIDS)

De acuerdo al capítulo II en la sección 2.2.10, la estructura jerárquica de la red está dividida en 3 capas:

A. CAPA DE ACCESO

La red esta creada de acuerdo a las diferentes áreas de la empresa: Gerencia, contabilidad, almacén y ventas. Las cuales forman parte de la capa de acceso, como se muestra en la figura 4.4.

Todas las áreas de la empresa cuentan con la misma topología y estructura de red, se cuenta con 5 ordenadores, 1 impresora, 1 switch y 1 access point. La topología que muestra en cada una de las áreas es la topología estrella, donde la conexión entre los diferentes dispositivos finales y el switch es de punto a punto. El access point puede tener conexión con los dispositivos por medios no guiados, para interconectar los 05 ordenadores con el switch se utiliza el cableado UTP y conectores RJ-45, mientras que para los móviles la conexión es través del wifi y el access point.

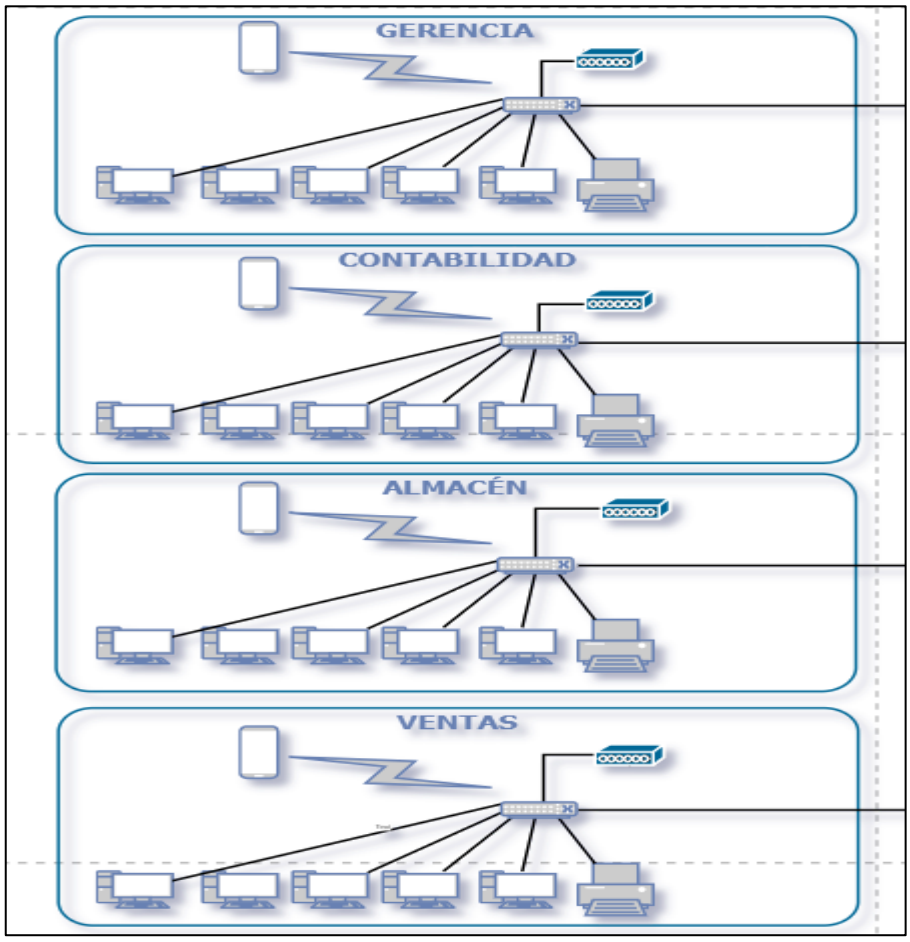


Figura 4.4. Estructura de red a nivel de la capa de acceso.

Tabla 4.2

Dispositivos de entrada y salida de red a nivel de la capa de acceso.

DISPOSITIVO DE ENTRADA	MEDIO	DISPOSITIVO DE SALIDA
PC 01	Cable UTP (guiado)	Switch
PC 02	Cable UTP (guiado)	Switch
PC 03	Cable UTP (guiado)	Switch
PC 04	Cable UTP (guiado)	Switch
PC 05	Cable UTP (guiado)	Switch
Impresora	Cable UTP (guiado)	Switch
Móvil	Wifi (no guiado)	Access Ponit

B. CAPA DE DISTRIBUCIÓN

En esta capa se conecta la capa acceso con la capa núcleo. La capa de distribución está conformada por 2 switch, como se muestra en la figura 4.5, lo que realiza es agregar los datos recibidos de la capa de acceso antes de que se transmitan a la capa núcleo para el enrutamiento hacia su destino final, se utiliza el cableado UTP y conectores RJ-45.

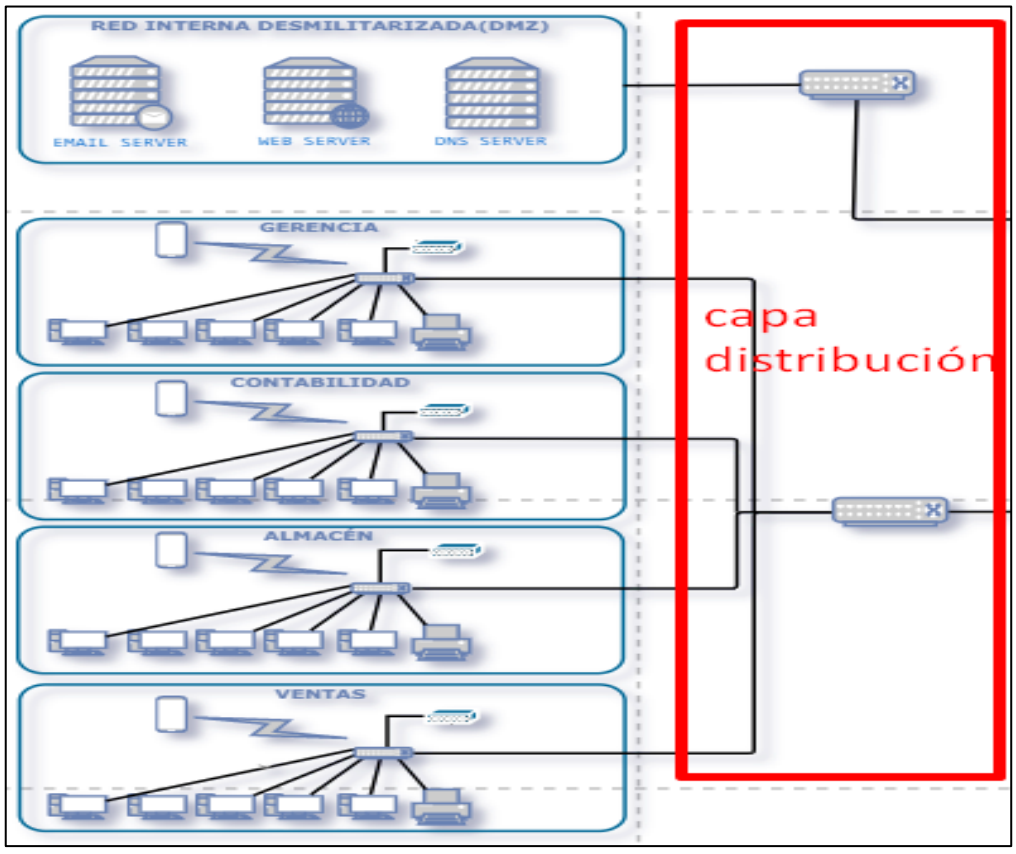


Figura 4.5. Estructura de red a nivel de la capa de distribución.

C. CAPA DE NÚCLEO

La capa núcleo está conformada por 1 switch, como se muestra en la figura 4.6, esta capa es fundamental para la interconectividad entre los elementos de la capa de distribución y la capa núcleo. La capa núcleo complementa el tráfico de todos los dispositivos de la capa de distribución, y reenvía grandes cantidades de datos rápidamente, para el enrutamiento hacia su destino se utiliza el cableado UTP y conectores RJ-45.

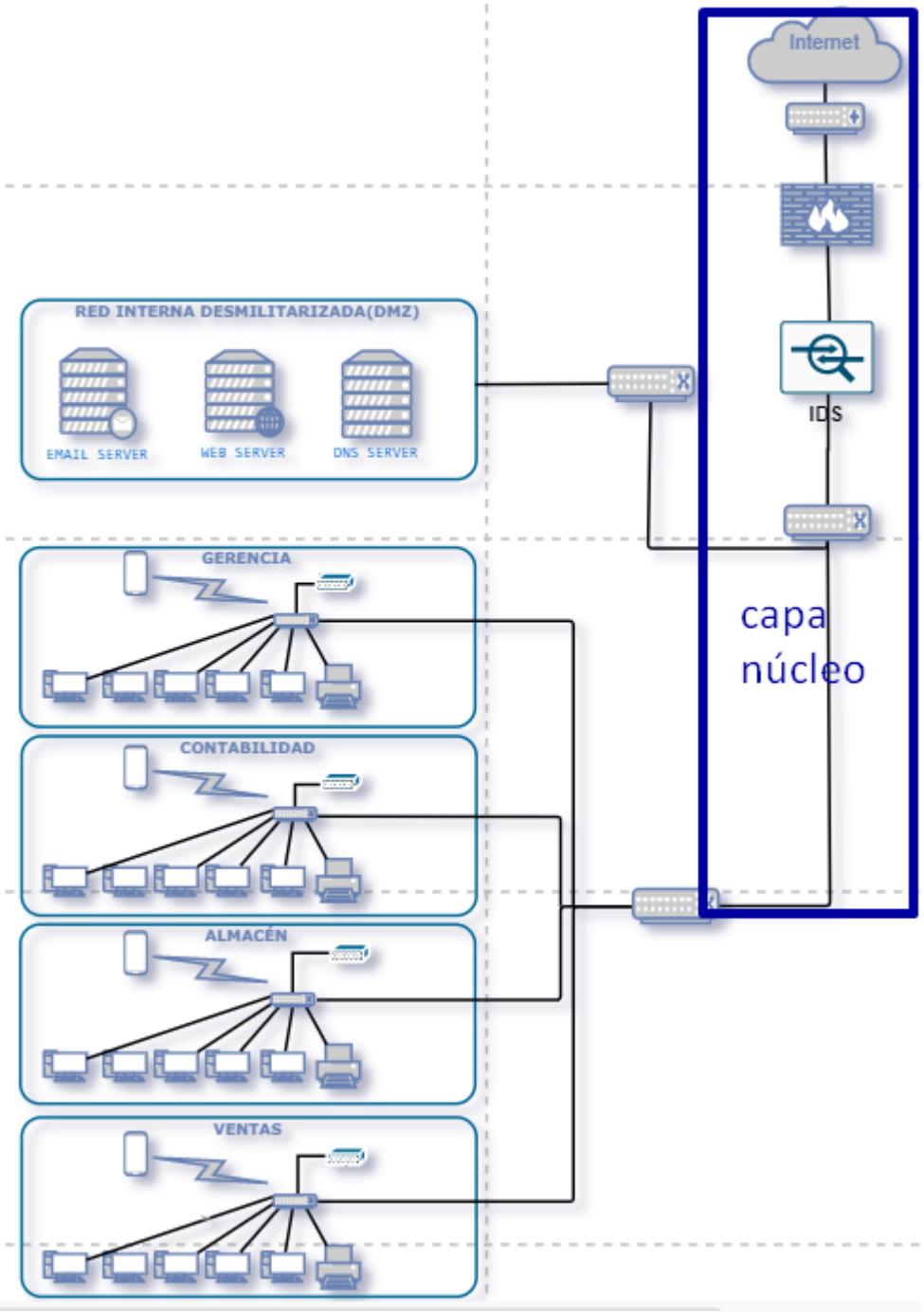


Figura 4.6. Estructura de red a nivel de la capa núcleo.

4.3. UBICACIÓN DEL SISTEMA DE DETECCIÓN DE INTRUSOS EN LA ESTRUCTURA DE RED

En este punto, se muestra en la figura 4.7 la estructura jerárquica de red LAN, basada en un Sistema de Detección a nivel de red para la Pyme.

De acuerdo al capítulo II en la sección 2.2.5, se optó por colocar el Sistema de Detección de Intrusiones (IDS) detrás del firewall, debido a que desde este punto estratégico para monitorizar las amenazas externas que lograron atravesar el firewall. Asimismo, identifica las intrusiones internas debido a que muchas amenazas son provocadas por usuarios internos al momento de ingresar a páginas web maliciosas o al ingresar dispositivos al ordenador.

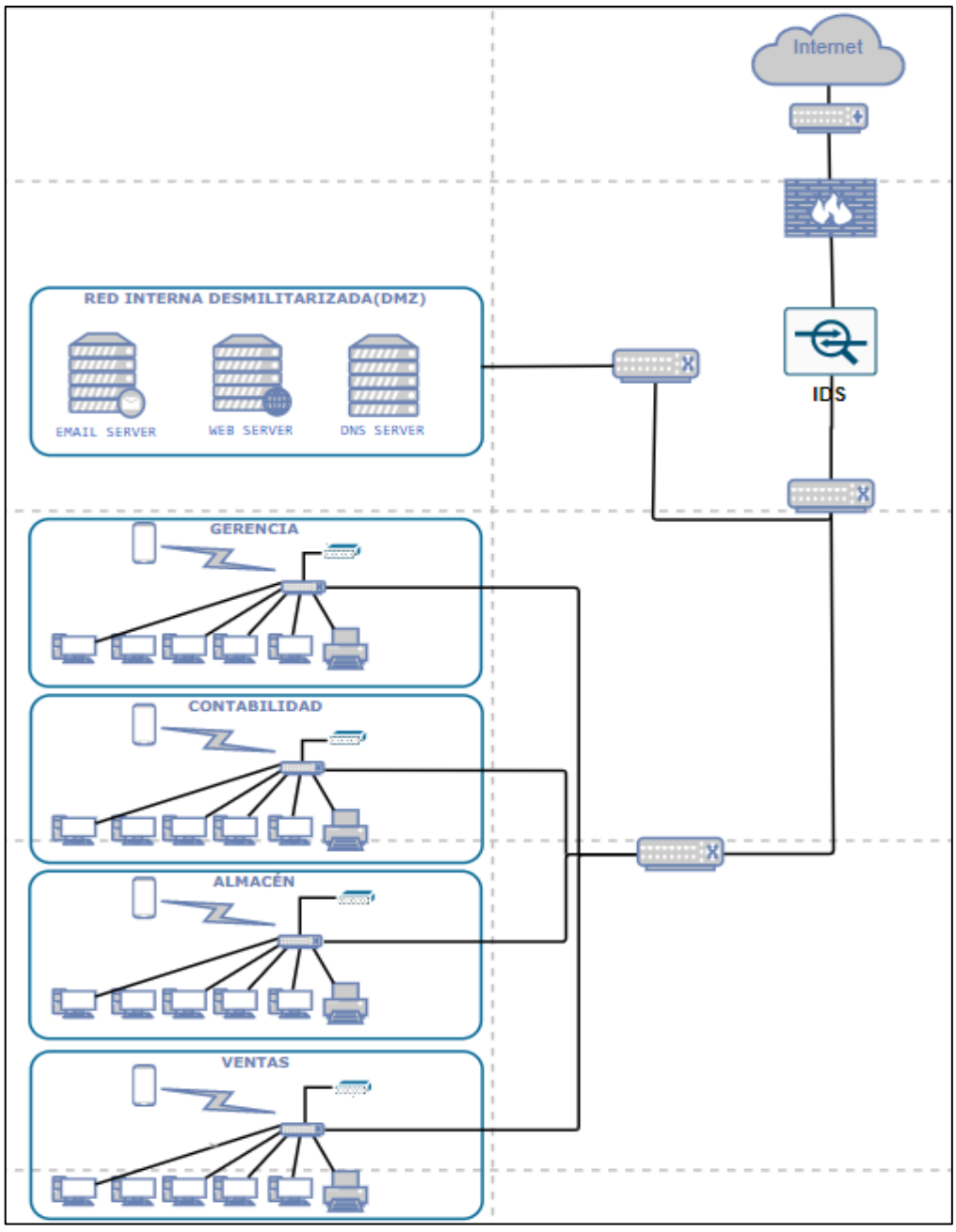


Figura 4.7. Estructura de red basada en sistema de detección de intrusiones a nivel de red.

4.4 CONFIGURACIÓN DEL SISTEMA DETECCIÓN DE INTRUSOS(IDS) SNORT

Antes de empezar con la configuración del Snort, tenemos que actualizar el sistema e instalar **openssh-server** (para que podamos administrar el sistema de forma remota). Seguidamente se reinicia para que se apliquen todas las actualizaciones.

```
sudo apt-get update
sudo apt-get dist-upgrade -y
sudo apt-get install -y openssh-server
sudo reboot
```

4.4.1 CONFIGURACIÓN DE LA TARJETA DE RED

A partir de Ubuntu 15.10, las interfaces de red ya no siguen el estándar **eth0**, para lo cual tendremos que verificar con el comando *ifconfig* el nombre de la red, en este caso el nombre es **enp0s3**.

```
root@snort:~# ifconfig
enp0s3    Link encap:Ethernet  direcciónHW 08:00:27:8d:8b:e3
          Direc. inet:10.0.2.24  Difus.:10.0.2.255  Másc:255.255.255.0
          Dirección inet6: fe80::a00:27ff:fe8d:8be3/64 Alcance:Enlace
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:1
          Paquetes RX:21 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:28 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1000
          Bytes RX:5899 (5.8 KB)  TX bytes:3468 (3.4 KB)

lo        Link encap:Bucle local
          Direc. inet:127.0.0.1  Másc:255.0.0.0
          Dirección inet6: ::1/128 Alcance:Anfitrión
          ACTIVO BUCLE FUNCIONANDO  MTU:65536  Métrica:1
          Paquetes RX:160 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:160 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:1
          Bytes RX:11840 (11.8 KB)  TX bytes:11840 (11.8 KB)

root@snort:~# _
```

Figura 4.8. Verificación de configuración de interfaz

Tenemos que deshabilitar **LRO** y **GRO** para cualquier interfaz que escuche Snort, utilizaremos *ethtool* comando en el archivo de configuración de la interfaz de red. Usamos **vi** para editar el archivo de interfaces de red:

```
sudo vi /etc/network/interfaces
```

Agregamos las siguientes dos líneas para cada interfaz de red, asegurándose de cambiar **enp0s3** según el nombre de la interfaz de red que tenga, ya que los nombres de su interfaz pueden ser diferentes.

```
post-up ethtool -K enpos3 gro off
post-up ethtool -K enpos3 lro off
```

```
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto enp0s3
iface enp0s3 inet dhcp
post-up ethtool -K enp0s3 gro off
post-up ethtool -K enp0s3 lro off
~
~
```

Figura 4.9. Configuración de interfaz

Verificamos que **LRO** y **GRO** estén deshabilitados, si no lo estuvieran tenemos que reiniciar con el comando ***sudo reboot*** y verificar**.**

```
sudo ifconfig enpos3 down && sudo ifconfig enpos3 up
ethtool -k enpos3 | grep receive-offload
```

```
root@snort:~# sudo ifconfig enp0s3 down && sudo ifconfig enp0s3 up
root@snort:~# ethtool -k enp0s3 | grep receive-offload
generic-receive-offload: off
large-receive-offload: off [fixed]
root@snort:~#
```

Figura 4.10. LRO y GRO deshabilitados.

4.4.2 INSTALACIÓN DE LOS REQUISITOS PREVIOS DE SNORT

Snort tiene cuatro requisitos previos principales:

PCAP	(libpcap-dev)	disponible desde el repositorio de Ubuntu
PCRE	(libpcre3-dev)	disponible desde el repositorio de Ubuntu
Libdnet	(libdumbnet-dev)	disponible desde el repositorio de Ubuntu
DAQ	(http://www.snort.org/downloads/)	

Primero instalamos todas las herramientas necesarias para crear software. Los elementos esenciales de construcción que hace esto por nosotros:

```
sudo apt-get install -y build-essential
```

Una vez que nuestras herramientas de compilación están instaladas, instalamos todos los requisitos previos de Snort que están disponibles en los repositorios de Ubuntu.

```
sudo apt-get install -y libpcap-dev libpcre3-dev libdumbnet-dev
```

Snort **DAQ** (biblioteca de adquisición de datos) tiene algunos requisitos previos que deben instalarse:

```
sudo apt-get install -y bison flex
```

Creamos una carpeta llamada ***snort_src*** para mantenerlos a todos en un solo lugar.

```
mkdir ~/snort_src
cd ~/snort_src
```

Descargue e instale la última versión de DAQ desde el sitio web de Snort. Los siguientes pasos usan **wget** para descargar la versión 2.0.7 de DAQ, que es la última versión al momento.

```
cd ~/snort_src
wget https://snort.org/downloads/snort/daq-2.0.7.tar.gz
tar -xvzf daq-2.0.7.tar.gz
cd daq-2.0.7
./configure
make
sudo make install
```

Cuando se ejecuta ***./configure***, debería ver la siguiente salida que muestra qué módulos se están configurando y cuáles estarán disponibles cuando compile DAQ:

Figura 4.11. Configuración DAQ (Modulo de adquisición de datos).

4.4.3 INSTALACIÓN DE SNORT

Para instalar Snort en Ubuntu, hay un requisito previo adicional que debe instalarse: **zlibg** que es una biblioteca de compresión.

Hay cuatro bibliotecas opcionales que mejoran la funcionalidad: ***liblzma-dev*** tres de los cuales proporcionan descompresión de archivos swf (adobe flash), openssl, y libssl-dev que proporcionan firmas de archivos SHA y MD5:

```
sudo apt-get install -y zlib1g-dev liblzma-dev openssl libssl-dev
sudo apt-get install -y libnghttp2-dev
```

Una vez que se hayan instalado todos los requisitos previos, descargamos Snort 2.9.16.1, para compilarlo y luego instalarlo.

```
cd ~/snort_src
wget https://snort.org/downloads/snort/snort-2.9.16.1.tar.gz
tar -xvzf snort-2.9.16.1.tar.gz
cd snort-2.9.16.1
./configure --enable-sourcefire
make
sudo make install
```

Ahora ejecutaremos el siguiente comando para actualizar las bibliotecas compartidas, si omitimos este paso obtendremos un error cuando intentemos ejecutar Snort:

```
sudo ldconfig
```

Colocaremos un enlace simbólico al binario Snort en /usr/sbin:

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

Probaremos Snort ejecutando el binario como usuario normal, pasándole el ***– V***, que le dice a Snort que se verifique a sí mismo y cualquier archivo de configuración que le pase.

```
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.16.1 GRE (Build 140)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2020 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.7.4
           Using PCRE version: 8.38 2015-11-23
           Using ZLIB version: 1.2.8

root@snort:~# _
```

Figura 4.12. Información sobre Snort 2.9.16.1

4.4.4 CONFIGURACIÓN DE SNORT PARA EJECUTAR EN MODO NIDS

Para ejecutar Snort crearemos una cuenta sin privilegios. También crearemos una serie de archivos y directorios requeridos por Snort, y estableceremos permisos en esos archivos. Snort tendrá los siguientes directorios: Configuraciones y archivos de reglas en ***/etc/snort.*** Las alertas serán escritas a ***/var/log/snort***. Las reglas compiladas **(.so rules)** se almacenarán en ***/usr/local/lib/snort_dynamicrules.***

```
# Crea usuario Snort y grupo:
sudo groupadd snort
sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
# Crea directorios de Snort:
sudo mkdir /etc/snort
sudo mkdir /etc/snort/rules
sudo mkdir /etc/snort/rules/iplists
sudo mkdir /etc/snort/preproc_rules
sudo mkdir /usr/local/lib/snort_dynamicrules
sudo mkdir /etc/snort/so_rules
# Crea algunos archivos que almacenan reglas y listas de IP
sudo touch /etc/snort/rules/iplists/black_list.rules
sudo touch /etc/snort/rules/iplists/white_list.rules
sudo touch /etc/snort/rules/local.rules
sudo touch /etc/snort/sid-msg.map
# Crea nuestros directorios de registro:
sudo mkdir /var/log/snort
sudo mkdir /var/log/snort/archived_logs
# Ajusta permisos:
sudo chmod -R 5775 /etc/snort
sudo chmod -R 5775 /var/log/snort
sudo chmod -R 5775 /var/log/snort/archived_logs
sudo chmod -R 5775 /etc/snort/so_rules
sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
```

Cuando queramos cambiar la propiedad de los archivos que creamos anteriormente para asegurarnos de que Snort pueda acceder a los archivos que usa:

```
sudo chown -R snort:snort /etc/snort
sudo chown -R snort:snort /var/log/snort
sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules
```

Snort necesita algunos archivos de configuración los cuales son:

1) classification.config
2) file magic.conf
3) reference.config
4) snort.conf
5) threshold.conf
6) attribute table.dtd
7) gen-msg.map
8) unicode.map

Para poder copiar los archivos de configuración y los preprocesadores dinámicos, ejecutaremos los siguientes comandos:

```
cd ~/snort_src/snort-2.9.16.1/etc/
sudo cp *.conf* /etc/snort
sudo cp *.map /etc/snort
sudo cp *.dtd /etc/snort
cd  ~/snort_src/snort-2.9.16.1/src/dynamic-preprocessors/build/usr/local/lib/snort_dynamicpreprocessor/
cd  ~/snort_src/snort-2.9.16.1/src/dynamic-preprocessors/build/usr/local/lib/snort_dynamicpreprocessor/
```

Ahora tenemos el siguiente diseño del directorio y las ubicaciones de los archivos:

Archivo Binario de Snort:	*/usr/local/bin/snort*
Archivo de configuración de Snort:	*/etc/snort/snort.conf*
Directorio de datos de registro de Snort:	*/var/log/snort*
Directorios de reglas de Snort:	*/etc/snort/rules*
	/etc/snort/so_rules
	/etc/snort/preproc_rules
	/usr/local/lib/snort_dynamicrules
Snort directorios de la lista de IP:	*/etc/snort/rules/iplists*
Preprocesadores dinámicos de Snort:	*/usr/local/lib/snort dynamicpreprocessor/*

Nuestro listado de directorios de Snort se ve así:

Figura 4.13. Distribución de directorios de Snort.

Cambiaremos algunas configuraciones ingresando en el archivo ***snort.conf***:

```
sudo vi /etc/snort/snort.conf
```

Cambiaremos las siguientes líneas para cumplir con su entorno:

HOME_NET debe coincidir con la red interna. En este caso es **HOME_NET** es 10.0.0.0 con una máscara de subred de 24 bits 255.255.255.0:

```
ipvar HOME_NET 10.0.0.0/24
```

No debemos cambiar **EXTERNAL_NET** a **$HOME_NET**, porque podríamos perder alertas de Snort.

```
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules


var WHITE_LIST_PATH /etc/snort/rules/iplists
var WHITE_LIST_PATH /etc/snort/rules/iplists
```

Habilitaremos el archivo de las reglas locales, donde podemos agregar reglas sobre las que Snort puede alertar. Tenemos que descomentar la línea:

```
include $RULE_PATH/local.rules
```

Una vez que el archivo de configuración esté listo, haremos que Snort verifique que es un archivo válido y que todos los archivos necesarios a los que hace referencia son correctos. Nos tiene que mostrar que ha sido ejecutado exitosamente una vez ejecutado el comando siguiente.

```
snort -T -c /etc/snort/snort.conf -i etho
```

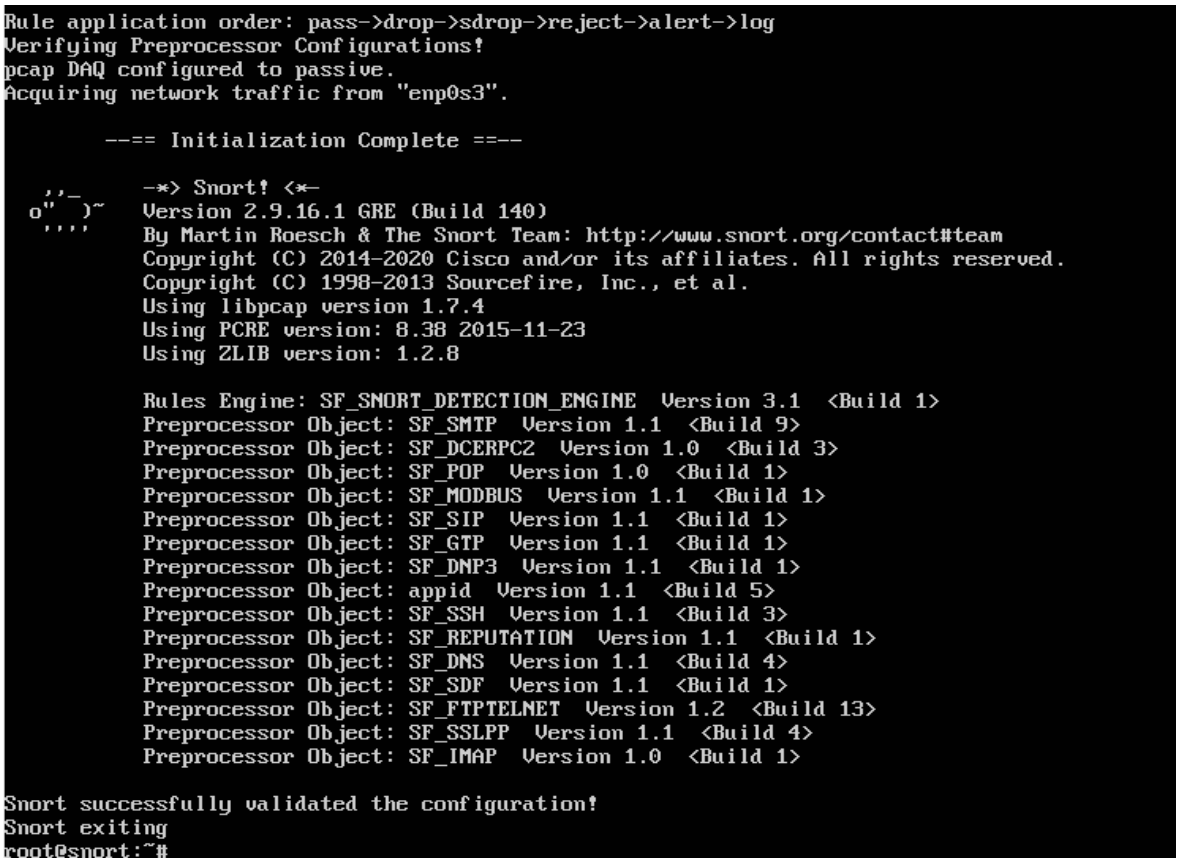


Figura 4.14. Ejecución de Snort 2.9.16.1

4.4.5 CONFIGURACIÓN Y TEST DE LAS REGLAS SNORT

De acuerdo al capítulo II en la sección 2.2.6.3, de donde podemos decir que, la variable **HOME_NET** define las direcciones de red a monitorear. Asimismo, la variable **EXTERNAL_NET** define qué hosts externos monitorear. El valor predeterminado de **any** permite a monitorear todas las direcciones de red local.

En esta etapa, se define las reglas en el motor de detección. Estas reglas serán colocadas en el directorio de ***local.rules*.**

Antes de configurar las reglas, tenemos que ejecutar Snort en modo Sistema de Detección de Intrusiones a nivel de red (NIDS) tal como se muestra en la Figura 4.15.

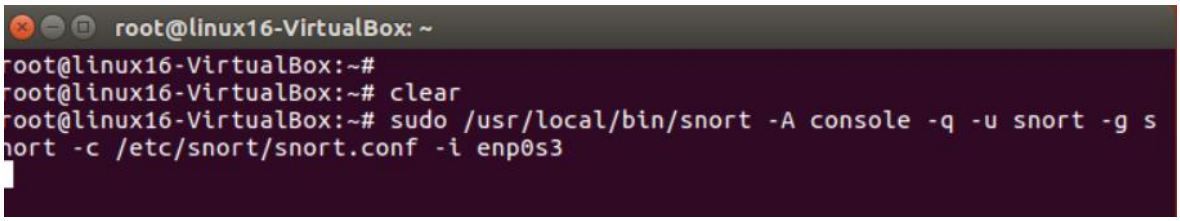


Figura 4.15. Snort en modo NIDS

1. ICMP detectada

Esta regla genera una alerta, con un comportamiento sospechoso en el protocolo ICMP desde cualquier dirección IP de origen y puerto. Teniendo como dirección de IP destino $HOME_NET ingresando por cualquiera de sus puertos. El mensaje que emite la alerta es "Prueba ICMP detectada".

```
alert icmp any any -> $HOME_NET any (msg:"Prueba ICMP detectada"; GID:1;
sid:10000001; rev:001; classtype:icmp-event;)
```

Verificamos si al realizar ***ping*** desde cualquiera de los hosts, hacia cualquiera de los puertos nos emite una alerta detectada.

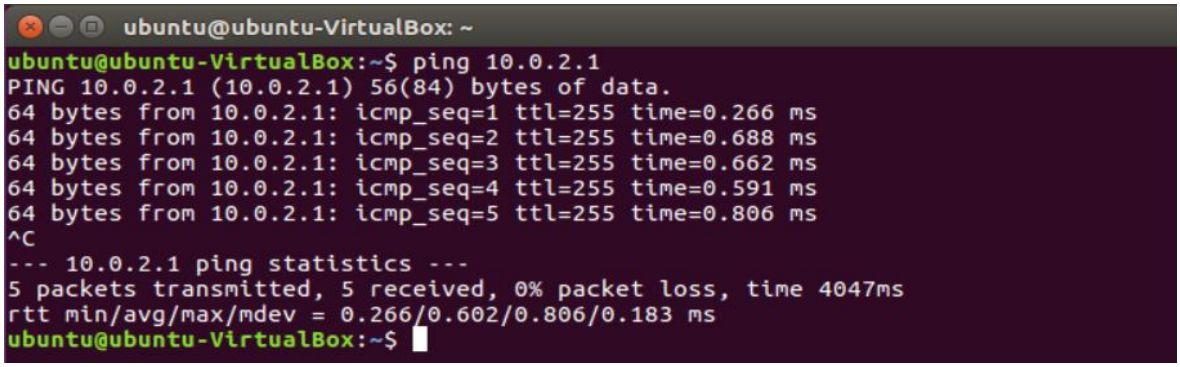


Figura 4.16. Ejecución de regla Nº 1.

En la figura b nos indica desde que dirección IP se realizó el ping, fecha y hora de la ejecución y tiempo de vulnerabilidad ICMP.

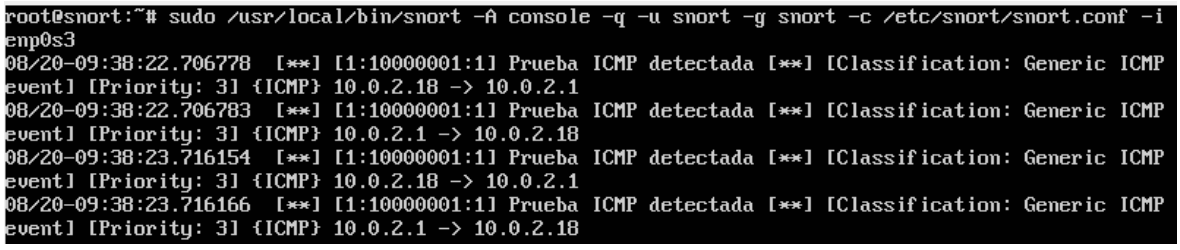


Figura 4.17. Alerta de regla Nº 1.

2. Escaneo de puertos TCP

La regla genera una alerta, con un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Teniendo como dirección de IP destino $HOME_NET, ingresando por cualquiera de los puertos. El mensaje que emite la alerta es "Escaneo de puertos TCP detectada".

```
alert tcp any any -> $HOME_NET any (msg: "Escaneo de puertos TCP detectada"; GID:1; sid:10000002; rev:001; classtype:web-application-attack; detection_fi ter:track by_src, count 30, seconds 60;)
```

Se realiza la prueba de la regla utilizando el comando ***nmap*** que analizará el escaneo cuando alguien intente escanear su red para identificar el host de red activos.

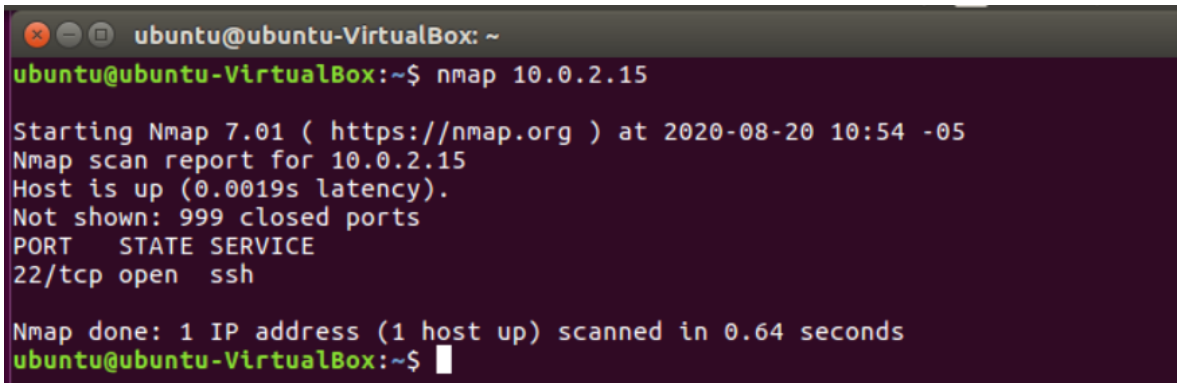


Figura 4.18. Ejecución de regla Nº 2.

En la figura 4.19 nos muestra la alerta que se registra al ejecutar el comando ***nmap***, la dirección de IP desde donde se realizó el escaneo es ***10.0.2.18,*** la clasificación de esta vulnerabilidad es ***Ataque desde una Aplicación web.***

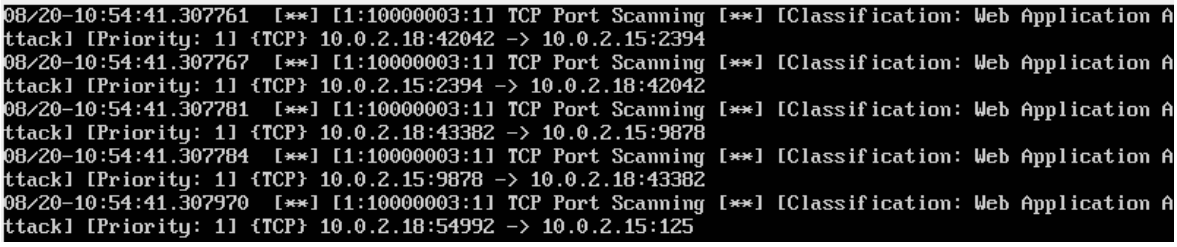


Figura 4.19. Alerta de regla Nº 2.

3. Intento de conexión FTP

La regla genera una alerta, con un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino $HOME_NET y puerto de destino 21. El mensaje que emite la alerta es "Intento de conexión FTP".

```
alert tcp any any -> $HOME_NET 21 (msg:"Intento de conexión FTP sid:10000003; rev:001;)
```

Realizamos la prueba de la regla donde ejecutamos el comando ***ftp*** y verificamos la conectividad del servidor ftp.

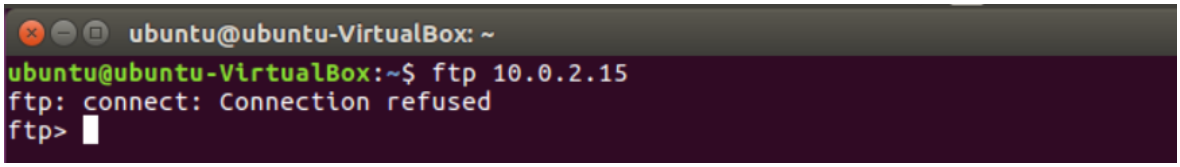


Figura 4.20. Ejecución de regla Nº 3.

La regla implementada se verifica correctamente, la figura 4.21 indica que la dirección ***ip_10.0.2.18*** se trata de conectar del puerto 21.

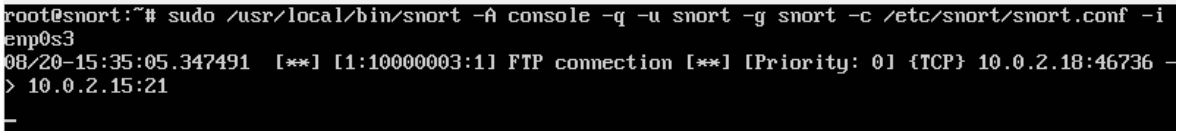


Figura 4.21. Alerta de regla Nº 3.

4. Intento de conexión TELNET

La regla genera una alerta, con un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino $HOME_NET y puerto de destino 80. El mensaje que emite la alerta es “Intento de conexión TELNET”.

```
alert tcp any any -> $HOME_NET 80 (msg:" Intento de conexión TELNET"; sid:10000004; rev:001;)
```

Para verificar la utilidad de la regla anterior se ejecuta el comando ***telnet 10.0.0.0 80*** para comprobar si el puerto de un ordenador remoto está abierto o cerrado.

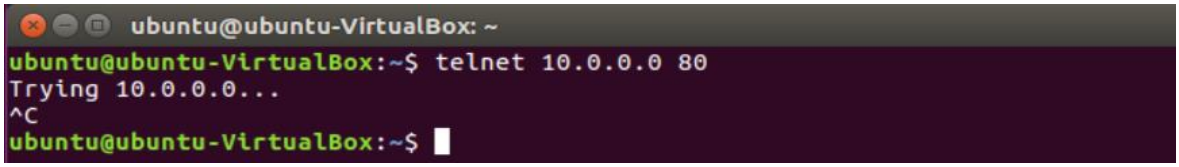


Figura 4.22. Ejecución de regla Nº 4.

La regla implementada se verifica correctamente, la figura 4.23 indica que la dirección ***ip_10.0.2.18*** se trata de conectar a través del puerto 80.

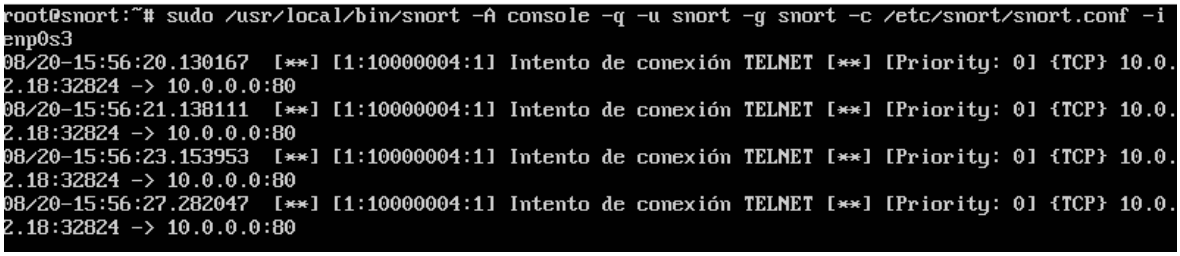


Figura 4.23. Alerta de regla Nº 4.

5. Posible ataque DoS TCP

La regla genera un alerta ataque de denegación de servicio distribuido, con un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino $HOME_NET y puerto de destino 80. El mensaje que emite la alerta es “Posible TCP DoS”.

```
Alert tcp any any -> $HOME_NET 80 (flags: S; msg “Possible TCP DoS;Flow:stateless;
threshold: type both, track by_src, count 70, seconds 10; sid:10000005;rev:001;)
```

Para verificar la regla implementada se utilizó la herramienta ***hping3*** que le permite enviar paquetes manipulados. Además, le permite controlar el tamaño, la cantidad y la fragmentación de los paquetes para sobrecargar el objetivo y evitar o atacar los firewalls.

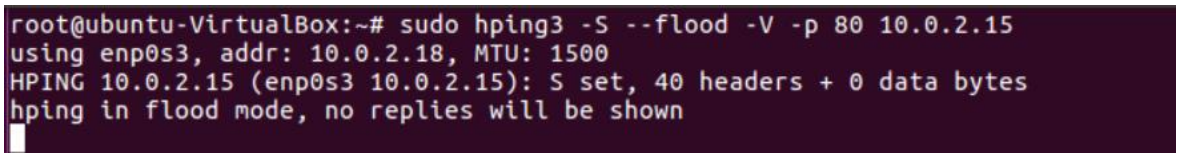


Figura 4.24. Ejecución de regla Nº 5.

En la figura 4.25 podemos observar que la regla funciona de manera efectiva, emitiendo una alerta de posible ataque de DoS (negación de servicios).

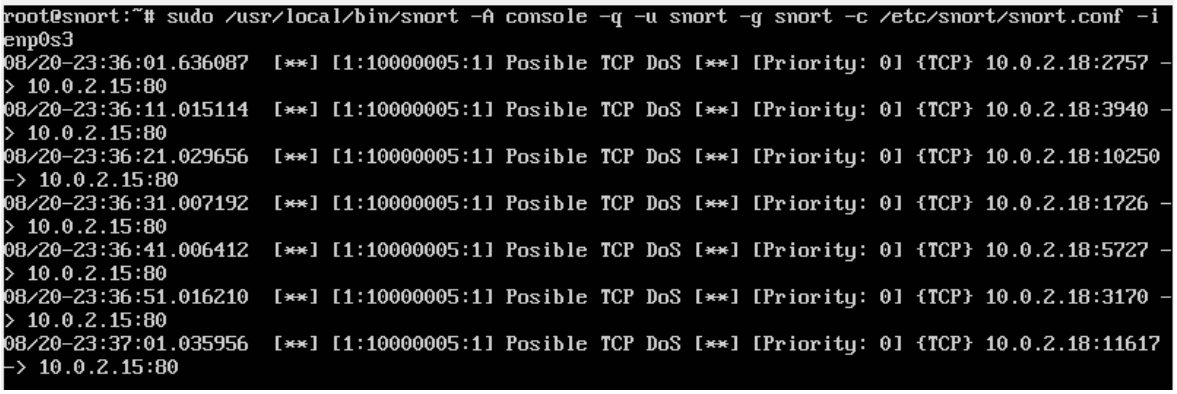


Figura 4.25. Alerta de regla Nº 5.

6. Conexión SSH detectada

La regla genera una alerta de una conexión no permitida de SSH, esto hace que las personas se conecten a una computadora local y remota. Esta regla tiene un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino 10.0.2.15 y puerto de destino 22. El mensaje que emite la alerta es "Conexión SSH detectada".

```
Alert tcp any any -> 10.0.2.15 22 (msg: "Conexión SSH detectetada"; sid:10000006;)
```

Para verificar la regla anterior utilizaremos la herramienta **PuTTY** desde donde podemos conectarnos a servidores remotos, ingresando el nombre o dirección del host y puerto.

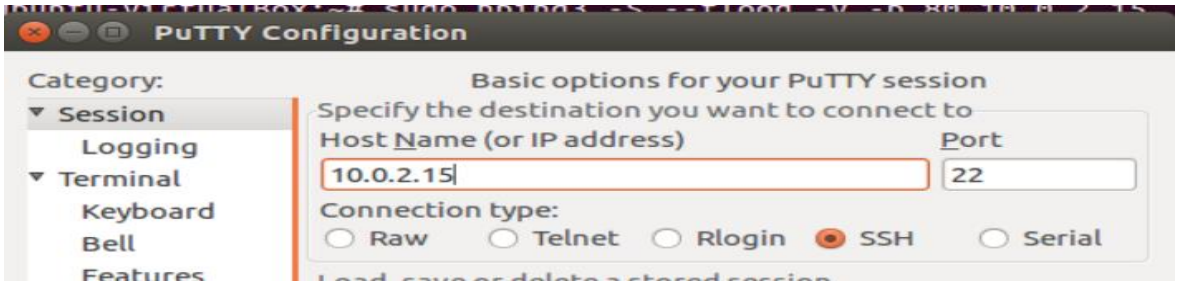


Figura 4.26. Ejecución de regla Nº 6.

En la figura 4.27 podemos observar que la regla funciona de manera efectiva, emitiendo una alerta de intento de conexión SSH.

```
root@snort:~# sudo /usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i
enp0s3
08/21-08:45:26.867061  [**] [1:10000006:0] Intento de conexion SSH [**] [Priority: 0] {TCP} 10.0.2.1
8:34957 -> 10.0.2.15:22
08/21-08:45:26.867410  [**] [1:10000006:0] Intento de conexion SSH [**] [Priority: 0] {TCP} 10.0.2.1
8:34957 -> 10.0.2.15:22
08/21-08:45:26.871454  [**] [1:10000006:0] Intento de conexion SSH [**] [Priority: 0] {TCP} 10.0.2.1
8:34957 -> 10.0.2.15:22
08/21-08:45:26.873573  [**] [1:10000006:0] Intento de conexion SSH [**] [Priority: 0] {TCP} 10.0.2.1
8:34957 -> 10.0.2.15:22
```

Figura 4.27. Alerta de regla Nº 6.

7. Alerta de ingreso a Facebook

La regla genera una alerta de una conexión a Facebook, esta regla tiene un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino $HOME_NET y puerto de destino any. El mensaje que emite la alerta es "Alguien se encuentra ingresando a Facebook".

```
Alert tcp any any -> $HOME_NET any (content: "www.facebook.com"; msg:" Alguien
se encuentra ingresando a Facebook"; sid:10000007; rev:001;)
```

Para verificar que la regla implementada funciona de manera efectiva, se ingresa a la página de la red social Facebook.

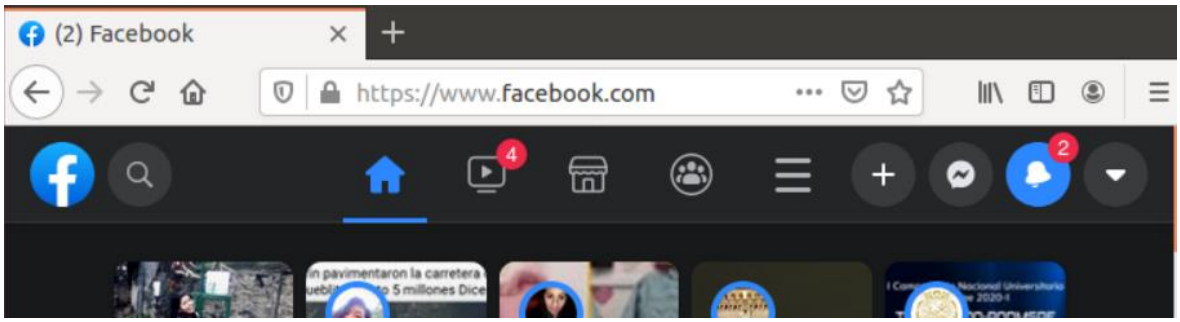


Figura 4.28. Ejecución de regla Nº 7.

De acuerdo a la Figura 4.29, podemos observar que la alerta funciona de manera correcta. Emite un mensaje de alerta.

```
root@snort:~# sudo /usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i
enp0s3
08/21-12:13:51.365555  [**] [1:10000007:1] Alguien se encuentra ingresando a Facebook [**] [Priority
: 0] {TCP} 10.0.2.18:59442 -> 157.240.197.35:443
08/21-12:13:52.183175  [**] [1:10000007:1] Alguien se encuentra ingresando a Facebook [**] [Priority
: 0] {TCP} 10.0.2.18:59446 -> 157.240.197.35:443
```

Figura 4.29. Alerta de regla Nº 7.

8. Alerta de ingreso a Youtube

La regla genera una alerta de una conexión a Youtube, esta regla tiene un comportamiento sospechoso en el protocolo TCP desde cualquier dirección IP de origen y puerto. Tiene como dirección de IP destino $HOME_NET y puerto de destino any. El mensaje que emite la alerta es “Alguien se encuentra ingresando a youtube”.

```
Alert tcp any any -> $HOME_NET any (content: “www.youtube.com”; msg:” Alguien se encuentra ingresando a youtube”; sid:10000008; rev:001;)
```

Para verificar que la regla implementada funciona de manera efectiva, se ingresa a la página de Youtube.

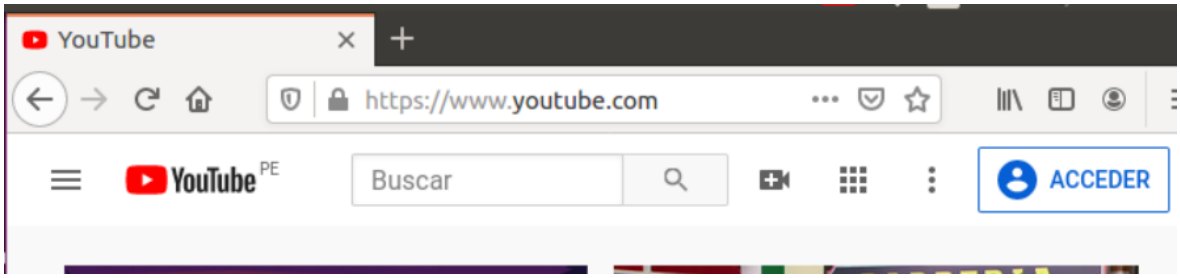


Figura 4.30. Ejecución de regla Nº 8.

De acuerdo a la Figura 4.31, podemos observar que la alerta funciona de manera correcta. Emite un mensaje de alerta.

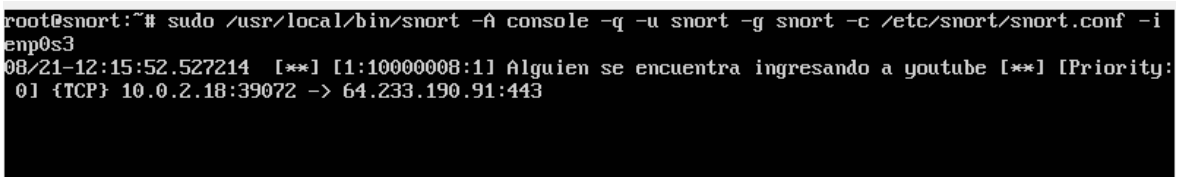


Figura 4.31. Alerta de regla Nº 8.

En la tabla 4.3 se puede observar la estructura general de la cabecera de la regla y la regla que emite:

Tabla 4.3

Reglas para motor de detección

PROTOCOLO	DIRECCION ORIGEN	DIRECCION DESTINO	PUERTO ORIGEN	PUERTO DESTINO	ALERTA
IMCP	any	HOME_NET	any	any	Prueba ICMP detectada
TCP	any	HOME_NET	any	any	Escaneo de puertos TCP detectada
TCP	any	HOME_NET	any	21	Intento de conexión FTP
TCP	any	HOME_NET	any	80	Intento de conexión TELNET
TCP	any	HOME_NET	any	80	Posible ataque DoS TCP
TCP	any	10.0.2.15	any	22	Conexión SSH detectada
TCP	any	HOME_NET	any	any	Alerta de ingreso a Facebook
TCP	any	HOME_NET	any	any	Alerta de ingreso a Youtube

Fuente: Elaboración propia

CAPITULO V
CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

De acuerdo al numeral al Capítulo IV, se puede afirmar que a lo largo de la investigación se logró implementar la y mostrar la utilidad del sistema de detección de instrucciones Snort, con respecto a la seguridad de la infraestructura a nivel de red, puesto en práctica en una Pyme. Al monitorear el tráfico y realizar las pruebas necesarias nos muestra el resultado como solución de una buena seguridad de la red.

De acuerdo al numeral 4.2, del Capítulo IV, se implementó el diseño de la estructura de red basada en Sistema de Detección de Intrusos (IDS), identificando el lugar más adecuado desde donde se logró detectar con éxito el tráfico de la red de la pyme y así monitorear el tráfico autorizado y no autorizado. Asimismo, permitirá estar alerta frente a cualquier tipo de incidente que se pudiera presentar.

De acuerdo al numeral 4.4.2, del Capítulo, se logró realizar la implementación de Snort como un método de captura de paquetes, para que al momento en que circula un paquete por la red, este es capturado por el módulo DAQ, que lo reenvía posteriormente a Snort, este lo analizara con el Sistema de Detección de Intrusiones a nivel de red.

De acuerdo al numeral 4.4.5, del Capítulo IV, se logró implementar reglas en el motor de reglas de Snort, y se realizó el filtrado de los eventos y vulnerabilidades, donde estas reglas alertan los posibles intentos ataques como denegación de servicios, ingresos a páginas no autorizadas, escaneo de puertos entre otros. Se logró verificar que dichas reglas implementadas cumplen con su propósito puesto que se realizaron las pruebas necesarias para garantizar que la misma cuenta con la funcionalidad necesaria y su funcionamiento es el esperado.

5.2 RECOMENDACIONES

En el ámbito de Snort como Sistema de detección de Intrusos se recomienda un estudio posterior, donde se debería habilitar la comunicación entre un motor de detección y otro para que de esta manera el conocimiento obtenido por los motores de detección se comparta entre todos estos. Esto permitiría tomar medidas más oportunas ante los ataques basados en las reglas de sistemas de detección de intrusos.

La evaluación del desempeño o de los sistemas de detección de intrusos IDS bajo ataques es una tarea difícil porque su diferentes IDS que se desarrollan y reglas que se actualizan. El número de ataques es también aumentando. Por lo tanto, se debe realizar un experimento para evaluar el rendimiento de los sistemas de detección de intrusiones Open Source. Entonces, podemos elegir la herramienta IDS efectiva para luchar contra los ataques maliciosos.

Cuando la tasa de transmisión de paquetes "maliciosos" es muy alta, el ataque casi con certeza es detectado por alguna regla implementada. Por lo tanto, el hecho de que el rendimiento de Snort cae cuando la tasa de transferencias es alta es importante de igual manera que se genere los falsos positivos y falsos negativos, se considera que estos problemas necesitan más investigación.

Una oportunidad de investigación es utilizar las técnicas inteligentes de Snort Open Source para desarrollar nuevas reglas para la prevención más precisa de ataques para contrarrestar el crecimiento en diversidad y astucia generando algoritmos complejos de detección, prevención y corrección.

BIBLIOGRAFÍA

Alva, M. (05 diciembre, 2019). Diario Gestión. Empresas sufrieron 3,000 millones de intentos de ciberataques. Recuperado de https://gestion.pe/economia/empresas-sufrieron-3000-millones-de-intentos-de-ciberataques-noticia/?ref=gesr.

Arbulú, J. (2006). Características E Importancia de la Pyme en Nuestra Economía. *La PYME en el Perú*. pp.32-37. Recuperado de http://cendoc.esan.edu.pe/fulltext/e-journals/PAD/7/arbulu.pdf.

Arteaga, J. E. (2019). Evaluation of the functionalities of the intrusion detection systems based on the network of open source platforms using the anomaly detection technique. *Latin-American Journal of Computing (LAJC),* 7(1), pp.53. Recuperado de https://lajc.epn.edu.ec/Volumenes/LAJC_vol7no1.pdf.

Bace, R. y Mell, P. (2015). NIST Special Publication on Intrusion Detection System. *Intrusion Detection Systems,* 3, pp.1-51. Recuperado de http://www.iwar.org.uk/comsec/resources/ids/sp800-31.pdf.

Baker, A., Beale, J. y Caswell, B. (2007). *Snort Intrusion Detection and Prevention Toolkit*. Burlington: SYNGRESS.

Baker, A.R., Esler, J. (2005). *Snort® IDS and IPS Toolkit*. Burlington: SYNGRESS.

Bardales, E. (25 septiembre, 2014). Diario Gestión. Una de cada cinco pymes es víctima de delitos cibernéticos, según Microsoft. Recuperado de https://gestion.pe/tecnologia/cinco-pymes-victima-delitos-ciberneticos-microsoft-73780-noticia/?ref=gesr.

Bernal, A. (2010). *Metodología de la investigación*. Mexico: Pearson Educación.

Bernal, C. (2006). *Metodología de la Investigación para Administración, Economía, Humanidades y Ciencias Sociales.* México D.F, México: Pearson Educación

Blum, R. y Bresnahan, C. (2016). *LPIC-2: Linux Professional Institute Certification Study Guide, 2nd Edition*. Recuperado de https://learning.oreilly.com/library/view/lpic-2-linux-professional/9781119150794/c11.xhtml.

Bul'ajoul, W., James, A. y Pannu, M. (2015). Journal of Computer and System Sciences. *A k-anonymous approach to privacy preserving collaborative filtering*, 81, pp.1000-1011. Recuperado de https://www.sciencedirect.com/science/article/pii/S0022000014001767.

Charkrabarti, S., Chakraborty, M. y Mukhopadhyay, I. (2010). International Conference and Workshop on Emerging Trends in Technology. *Study of Snort-Based IDS.* 1, pp.43-47. https://www.researchgate.net/publication/220902217_Study_of_snort-based_IDS.

CERTSI. (2017). Cert de Seguridad e Industria. *Design and Configuration of IPS, IDS and SIEM in Industrial Control Systems*, 1, pp.19-56. Recuperado de https://www.incibe-cert.es/sites/default/files/contenidos/guias/doc/certsi_design_configtion_ips_ids_siem_in_ics.pdf.

CISCO. (01 abril, 2014). Diseño de red LAN cableada [Documento informativo]. Recuperado de https://www.cisco.com/c/dam/r/es/la/internet-of-everything-ioe/assets/pdfs/en-05_campus-wireless_wp_cte_es-xl_42333.pdf

Cisco EPC3825 (s.f). En *wiki.bandaancha*. Recuperado el 05 de julio de 2020 de https://wiki.bandaancha.st/Cisco_EPC3825.

Clarke, G. E. (2017). *CompTIA Security+ Certification Study Guide, Third Edition (Exam SY0-501), 3rd Edition*. Estados Unidos: McGraw-Hill.

Costas Santos, J. (2014). *Seguridad y Alta Disponibilidad.* España: RA-MA.

Cox, K. y Gerg, C. (2004). *Managing Security with Snort & IDS Tools*: Recuperado de https://learning.oreilly.com/library/view/managing-security-with/0596006616/pr02.html.

Coyla Jarita, Y. (2019). *Implementación de un sistema de detección y prevención de intrusos (IDS/IPS), basado en la norma ISO 27001, para el monitoreo perimetral de la seguridad informática, en la red de la Universidad Peruana Unión – Filial Juliaca* (Tesis Licenciatura, Universidad Peruana Unión). Recuperado de https://repositorio.upeu.edu.pe/handle/UPEU/2002.

Dash, P. (2013). *Introducción a Oracle VM VirtualBox*. Reino Unido: Packt Publishing.

De Haro Bermejo, F. (2015). *Detección de intrusiones con Snort* (Tesis de Posgrado, Universitat Oberta de Catalunya). Recuperado de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/43100/5/fdeharobTFM0615memoria.pdf.

Diario El Peruano (09 marzo, 2020). Diario El Peruano. Perú entre países más inseguros en línea en América Latina. Recuperado de https://elperuano.pe/noticia-peru-entre-paises-mas-inseguros-linea-america-latina-90806.aspx.

Diseño jerárquico de la red. (s.f). En *Wikipedia*. Recuperado el 05 de Agosto de 2020 de https://es.wikipedia.org/wiki/Dise%C3%B1o_jer%C3%A1rquico_de_la_red#Modelos_del_dise%C3%B1o_jer%C3%A1rquico.

Dulaney, E. y Easttom, C. (2017). *CompTIA Security+ Study Guide, 7th Edition*. Recuperado de https://learning.oreilly.com/library/view/comptia-security-study/9781119416876/fcopyright.xhtml.

Easttom, C. (2014). *Network Defense and Countermeasures: Principles and Practices, Second Edition*. Estados Unidos: Pearson IT Certification.

Farro Flores, C. (2019). "Uno de los activos más importantes del negocio es la información". *Ciberdelincuencia: Amenaza Latente*, 38, pp.11. Recuperado de https://www.bascperu.org/pdf/principales/REVISTA-38_opt.pdf.

Francois Carpentier, J. (2016). *La seguridad informática en la PYME: Situación actual y mejores prácticas*. Barcelona: Ediciones ENI.

González Barahona, J.M. (2011). El concepto de software libre: *Revista Tradumàtica Tecnologies de la Traducció*, 9, pp. 5-11. Recuperado de https://www.oreilly.com/openbook/osfreesoft/book/index.html.

Gordon, D. (2019). *Networking Fundamentals.* Reino Unido: Packt.

Goswami, S. y Misra, S. (2017). *Network Routing*. Reino Unido: Wiley.

Haines, N. (2017). *Beginning Ubuntu for Windows and Mac Users: Start your Journey into Free and Open Source Software*. New York: Apress.

Hernández, B. (2001). *Técnicas estadísticas de investigación social*. España: Diaz de Santos S.A.

Hernández Sampieri, R., Fernández, C. y Baptista, P. (2014) *Metodología de la investigación* (6ta Ed.). México, D.F., México: McGraw Hill Interamericana.

Interfaz de red TCP/IP. (s.f). En IBM Knowledge Center. Recuperado el 22 de Junio 2020 de https://www.ibm.com/support/knowledgecenter/es/ssw_aix_72/network/tcpip_interfaces.html.

Izquierdo Cabrera, J. y Tafur Callirgos, T. E. (2017). *Mecanismos De Seguridad Para Contrarrestar Ataques Informáticos En Servidores Web Y Base De Datos* (Tesis Licenciatura, Universidad Señor De Sipán). Recuperado de http://repositorio.uss.edu.pe/handle/uss/4062.

Jeff, V., Jeff, S., Gary, C. y Bob, N. (2017). *Oracle® Solaris 11 System Virtualization Essentials, segunda edición*. Recuperado de https://learning.oreilly.com/library/view/oracle-solaris-11/9780134309828/copy.html

Kaspersky (2020). Ciber Amenaza Mapa En Tiempo Real. *Estadística histórica mundial*. Recuperado de https://cybermap.kaspersky.com/es/stats/#country=4&type=ids&period=w.

Koziol, J. (2003). *Intrusion Detection with Snort*. Estados Unidos: Sams.

Lane, N., Conklin, W. A., White, G. B. y Williams, D. (2019). *CASP+ CompTIA Advanced Security Practitioner Certification All-in-One Exam Guide, Second Edition (Exam CAS-003)*. Estados Unidos: McGraw-Hill.

Llopis Polvoreda, J. (2017). Sistema De Monitorización Del IDS Snort (Tesis Pregrado, Universitat Politècnica de València). Recuperado de https://riunet.upv.es/bitstream/handle/10251/88474/LLOPIS%20%20Sistema%20de%20monitorizaci%C3%B3n%20del%20IDS%20Snort.pdf?sequence=1.

Mandia, K., Luttgens, J. y Pepe, M. (2014). *Incident Response & Computer Forensics, Third Edition, 3rd Edition*. Estados Unidos: McGraw-Hill.

Messier, R. (2019). CEH v10 *Certified Ethical Hacker Study Guide*. Recuperado de https://learning.oreilly.com/library/view/ceh-v10certified/9781119533191/ftoc.xhtml.

Mira Alfaro, E. J. (2002). *Implantación de un Sistema de Detección de Intrusos en la Universidad de Valencia* (Tesis Licenciatura, Universidad de Valencia). Recuperado de http://rediris.es/cert/doc/pdf/ids-uv.pdf.

Novak, J. y Northcutt S. (2002). *Network Intrusion Detection, Third Edition*. Estados Unidos: Sams.

Rathaus, N., Ramirez, G., Caswell, B. y Beale, J. (2005). *Nessus, Snort, and Ethereal Power Tools: Customizing Open Source Security Applications*. Recuperado de https://learning.oreilly.com/library/view/nessus-snortand/9781597490207/ch07.html.

Rehman, R. (2003). Intrusion Detection With SNORT, Apache, MySQL, PHP, And ACID. Estados Unidos: Pearson Technology Group.

Romero, A. (2010). *VirtualBox 3.1: Beginner's Guide*. Recuperado de http://csciun1.mala.bc.ca:8080/~pwalsh/teaching/251/Tasks/vbsc.pdf.

Prakash, O. y Kumar, V. (2012). International Journal of Computer Applications & Information Technology. *Signature Based Intrusion Detection System Using SNORT*, Vol. I, pp. 35-41. Recuperado de https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.462.1508&rep=rep1&type=pdf.

OPENBIZ. (11 septiembre, 2009). OPEN SOURCE [Documento informativo]. Recuperado de http://www.openbiz.com.ar/Open%20Source.pdf.

Orebaugh, A., Biles, S. y Babbin, J. (2009). *Snort Cookbook Solutions and Examples for Snort Administrators*. Recuperado de https://learning.oreilly.com/library/view/snort-cookbook/0596007914/ch01.html.

Parisi, A. (2019). *Hands-On Artificial Intelligence for Cybersecurity*. Reino Unido: Packt Publishing.

Parziale, L., Acosta, P., Bader, F. y Novak, P. (2016). *The Virtualization Cookbook for IBM z Systems Volume 4: Ubuntu Server 16.04*. Recuperado de https://learning.oreilly.com/library/view/the-virtualization-cookbook/9780738442006/8354itsoad.xhtml#ww773577.

Prowse, D. L. (2017). *CompTIA® Security+ SY0-501 Cert Guide*. Estados Unidos: PEARSON.

Pymes. (s.f). En *Wikipedia*. Recuperado el 27 de mayo de 2020 de https://es.wikipedia.org/wiki/Peque%C3%B1a_y_mediana_empresa.

Santos, O y Gregg, M. (2019). *Certified Ethical Hacker (CEH) Version 10 Cert Guide, 3rd Edition.* Estados Unidos: Pearson IT Certification.

Scoot, C., Wolfe, P. y Hayes, B. (2004). *SNORT*. Canadá: Dummies.

Sharma, S., Kumar, A. y Tasneem, A. (2018). International Journal of Computer Applications. *Intrusion Detection Prevention System using SNORT*, 181, p.21-24. Recuperado https://www.researchgate.net/publication/329716671_Intrusion_Detection_Prevention_System_using_SNORT.

Smith, J. y Sanders, C. (2013). *Applied Network Security Monitoring*. United States of America: Syngress.

Snort.org. (2020). *SNORT R User's Manual 2.9.16*. Recuperado de https://www.snort.org/documents.

Stevens, W. R. y Fall, K. R. (2011). *TCP/IP Illustrated, Volume 1: The Protocols*. Estados Unidos: Addison-Wesley Professional

Tamayo M. y Tamayo A. (1997). El Proceso de la Investigación Científica. (3ª ed.). México: Limusa.

Tech Club Tajamar. (06 de Julio, 2016). *Diseño jerárquico de redes.* [Documento informativo]. Recuperado de https://techclub.tajamar.es/descripcion-del-modelo-jerarquico-de-la-red/

Thompson, E. C. (2020). *Designing a HIPAA-Compliant Security Operations Center: A Guide to Detecting and Responding to Healthcare Breaches and Events*. Estados Unidos: Apress.

Tituaña Haro, N. F. (2013). *Rediseño de la red LAN de la Secretaria Nacional de Telecomunicaciones, matriz Quito para soportar servicios de voz, datos y video* (Tesis Licenciatura, Escuela Politécnica Nacional). Recuperado de http://bibdigital.epn.edu.ec/handle/15000/6344.

Tomas, J. (2009). *Fundamentos de bioestadística y análisis de datos para enfermería.* España: Bellaterra.

Vacca, J. R. (2012). *Computer and Information Security Handbook, 2nd Edition*. Recuperado de https://learning.oreilly.com/library/view/computer-and-information/9780123943972/xhtml/Title_page.html.

Woland, A., Kampanakis, P. y Santos, O. (2016). *Cisco Next-Generation Security Solutions: All-in-one Cisco ASA Firepower Services, NGIPS, and AMP*. Recuperado de https://learning.oreilly.com/library/view/cisco-next-generation-security/9780134213071/ch12.html#ch12.

Yauri Lozano, E. (2017). *SnorUNI: Aplicación móvil para Sistemas de Detección y Prevención de Intrusiones basados en Snort* (Tesis Licenciatura, Universidad Nacional de Ingeniería). Recuperado de http://cybertesis.uni.edu.pe/handle/uni/5651.

ANEXO A

INSTALACIÓN DE UBUNTU SERVER 16.04.7

Una vez descargada desde la página oficial de Linux Ubuntu, se procede a instalar nuestro servidor y escogemos el idioma en que instalaremos, en este caso, español presionando *Enter*.

Figura 6.1. Seleccionar idioma.

Seleccionamos Instalar Ubuntu Server presionando *Enter*.

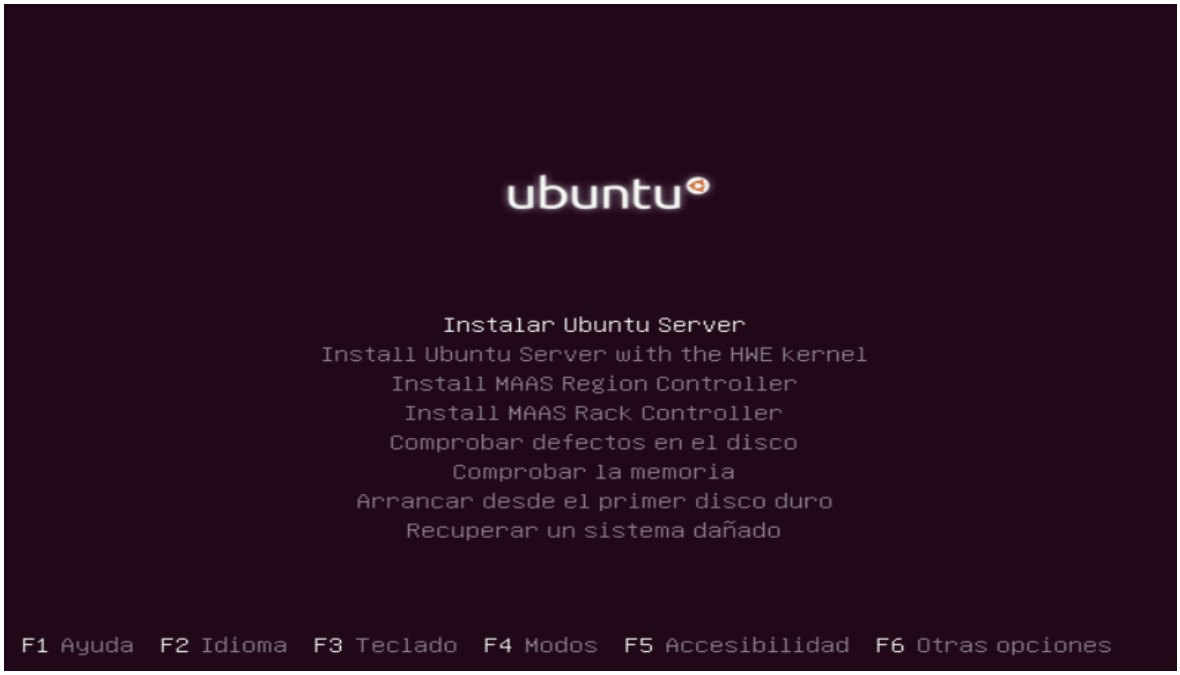


Figura 6.2. Instalar Ubuntu server.

Escogemos nuestra ubicación y presionamos *Enter*.

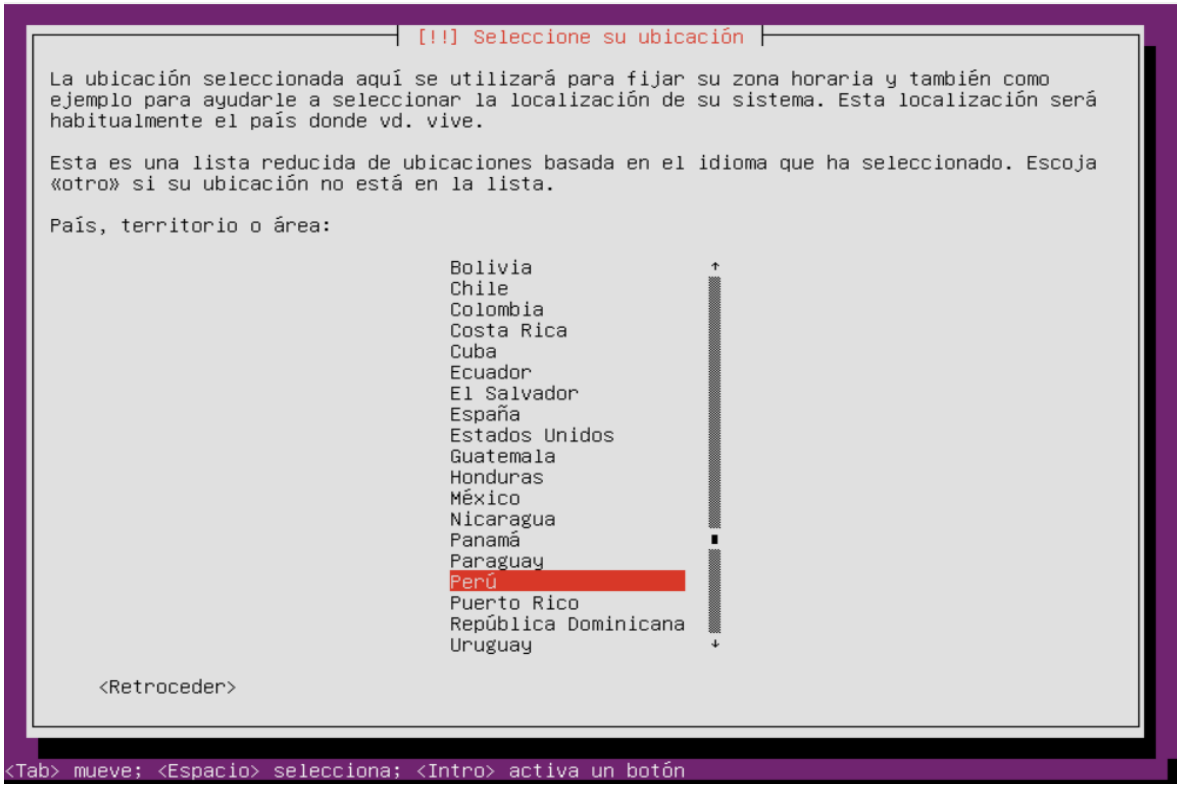


Figura 6.3. Seleccionar País.

Escogemos la distribución del teclado que utilizamos luego presionamos *Enter*.

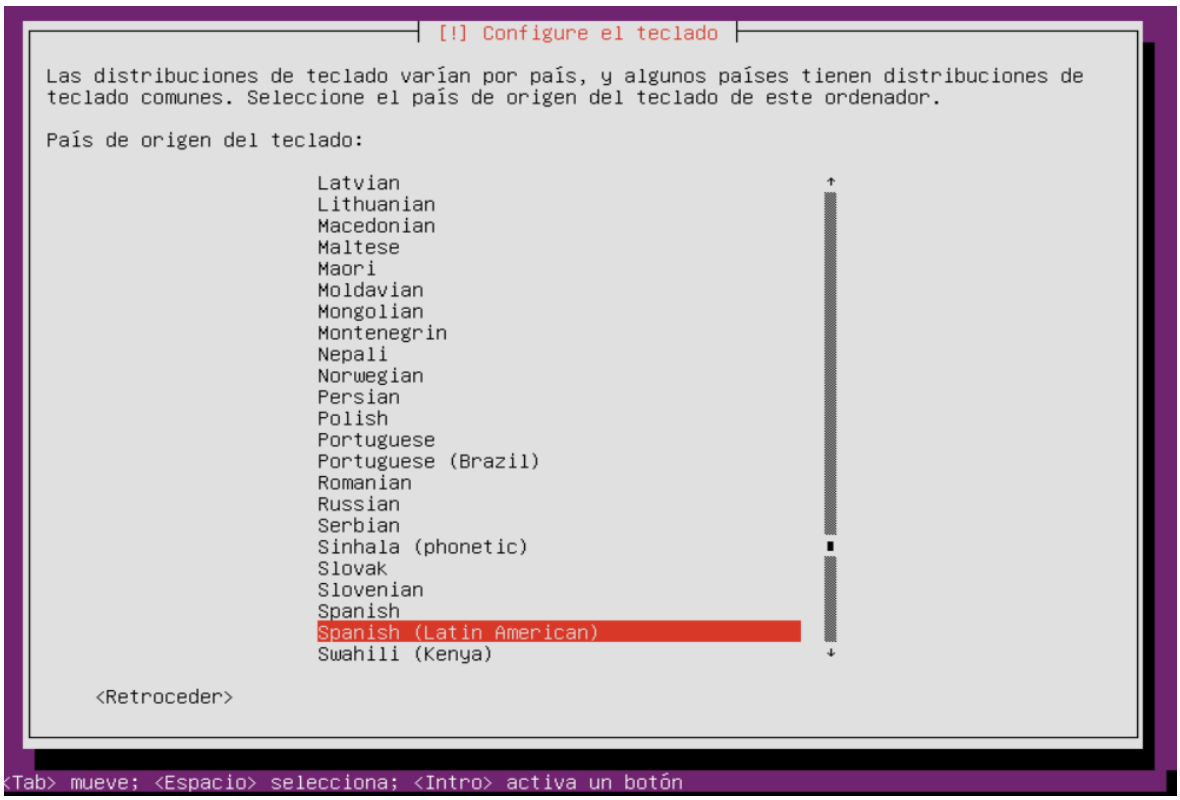


Figura 6.4. Seleccionar teclado de ordenador.

Nos piden que confirmemos nuestra distribución de teclado, seguidamente presionamos *Enter*.

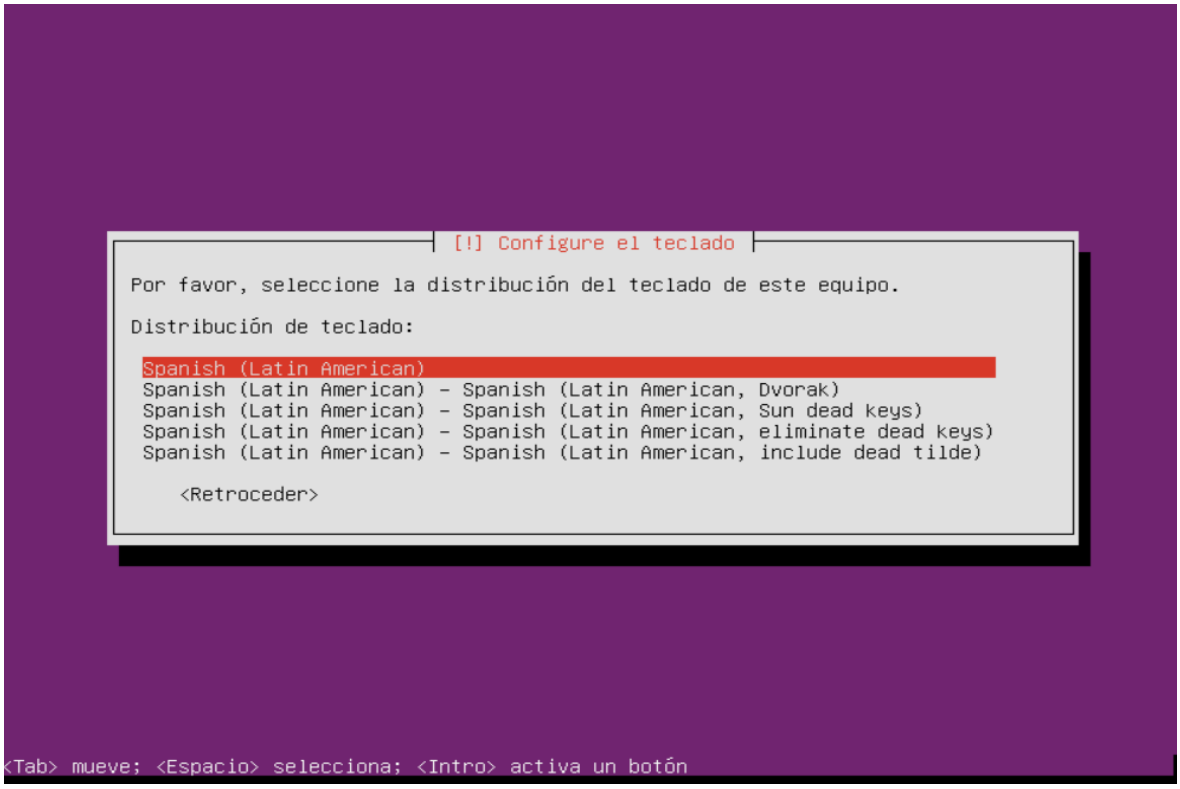


Figura 6.5. Seleccionar distribución teclado.

Introducimos el nombre de cómo queremos que se llame la maquina con los parámetros que nos indica en la siguiente imagen, seguidamente presionamos e*nter* sobre el botón *continuar*.

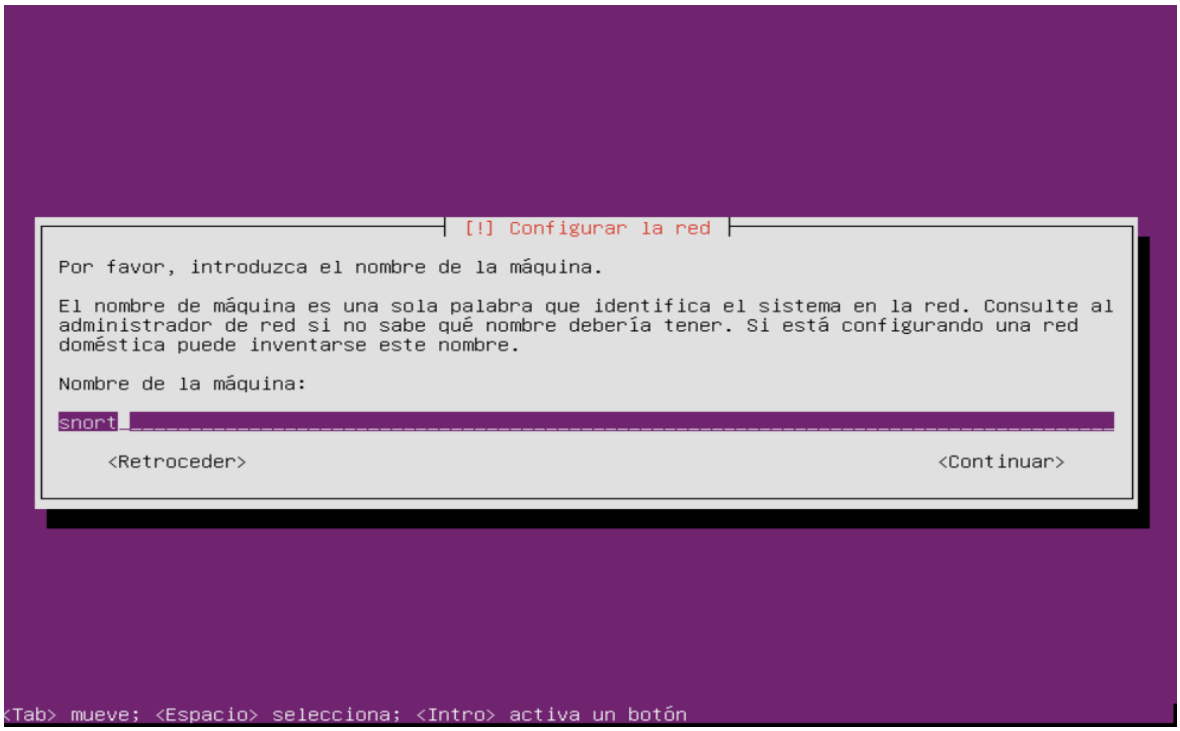


Figura 6.6. Configurar Red.

Creamos una contraseña con las recomendaciones de utilizar letras, números y signos de puntuación. Luego presionamos *enter* sobre el botón *continuar*.

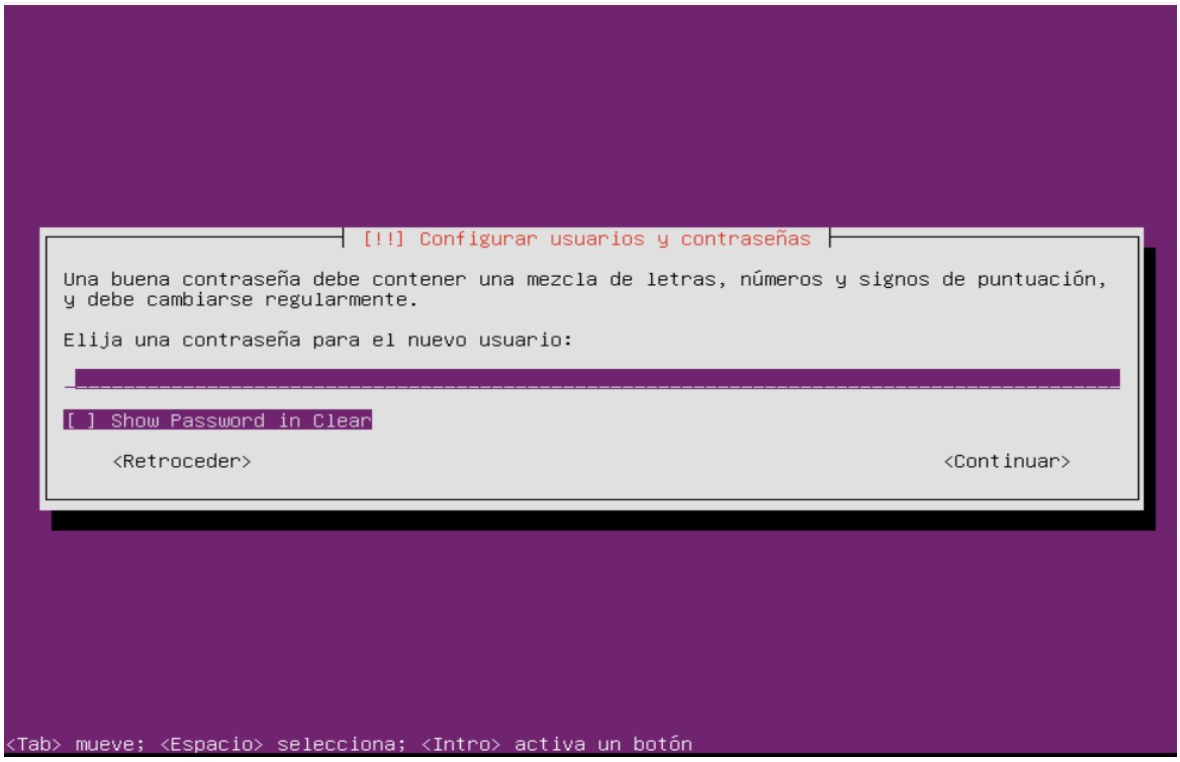


Figura 6.7. Configurar Usuario.

En esta opción lo recomendable es dar *enter* en el botón *No*.

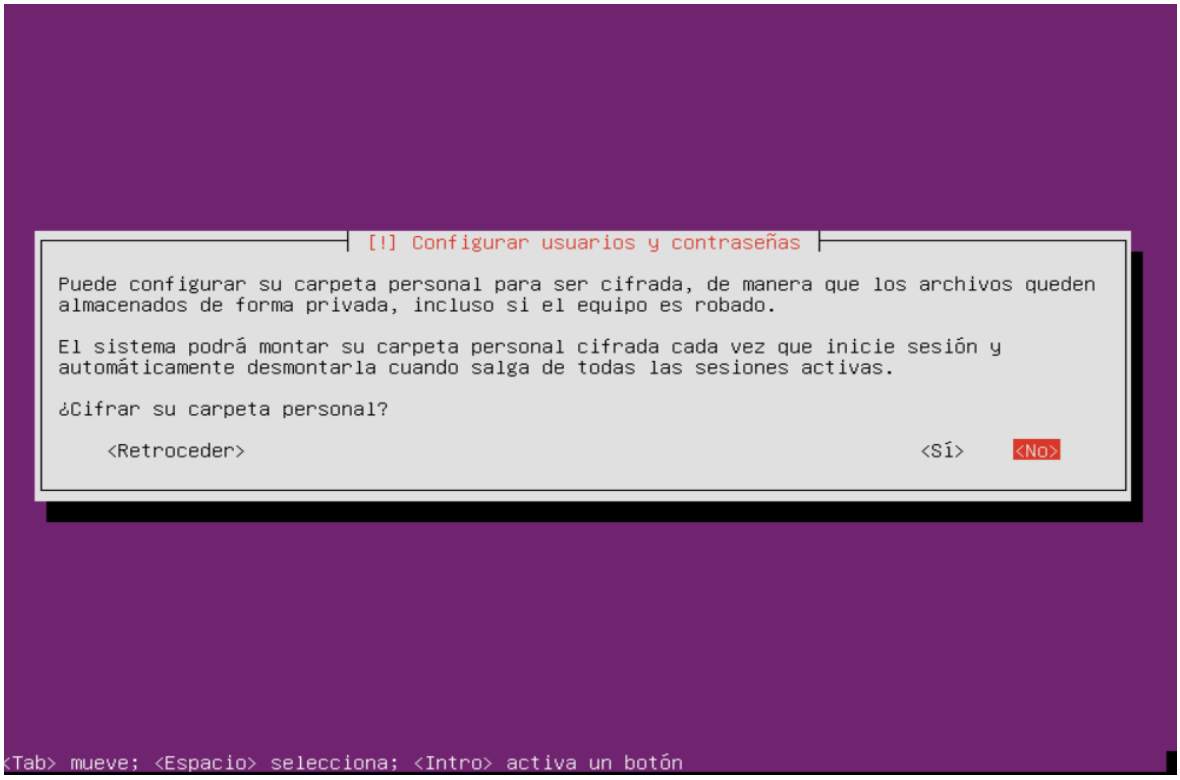


Figura 6.7. Configurar Usuario y Contraseña.

En este paso para particionar el disco. Elegimos la segunda opción, Guiado con LVM y presionamos *enter.*

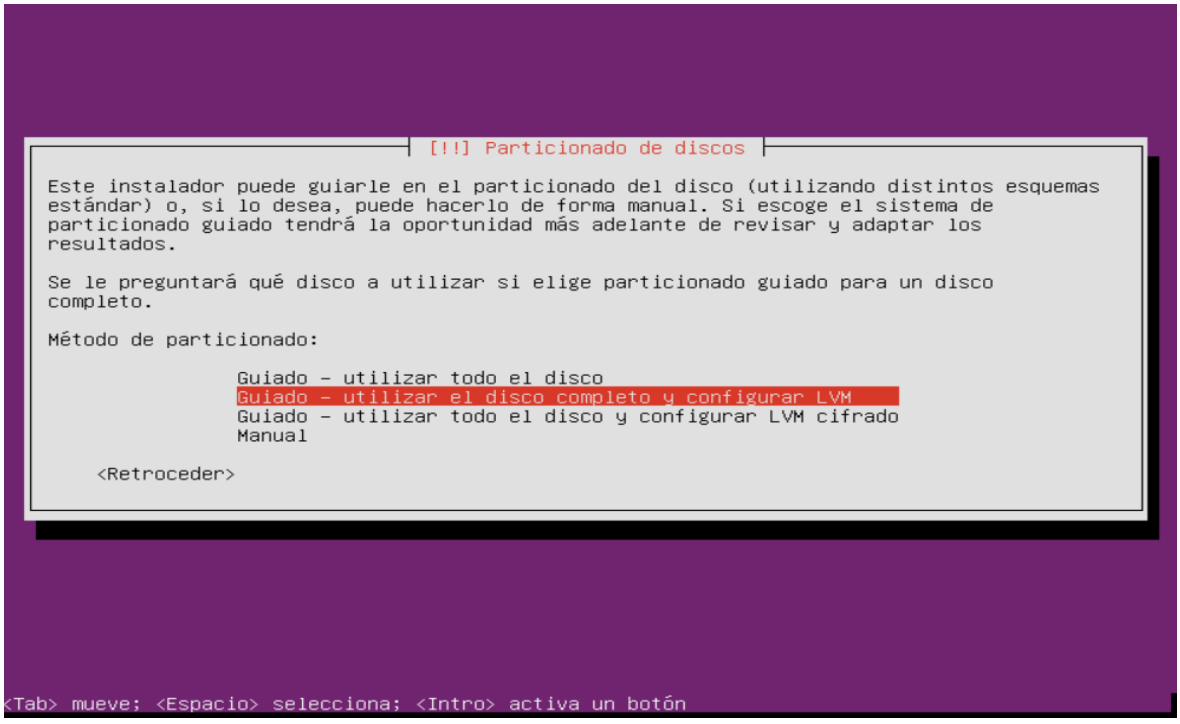


Figura 6.8. Particionando Discos.

Nos mostraran discos detectados en el equipo, escogemos cual se usar y presionamos *enter*.

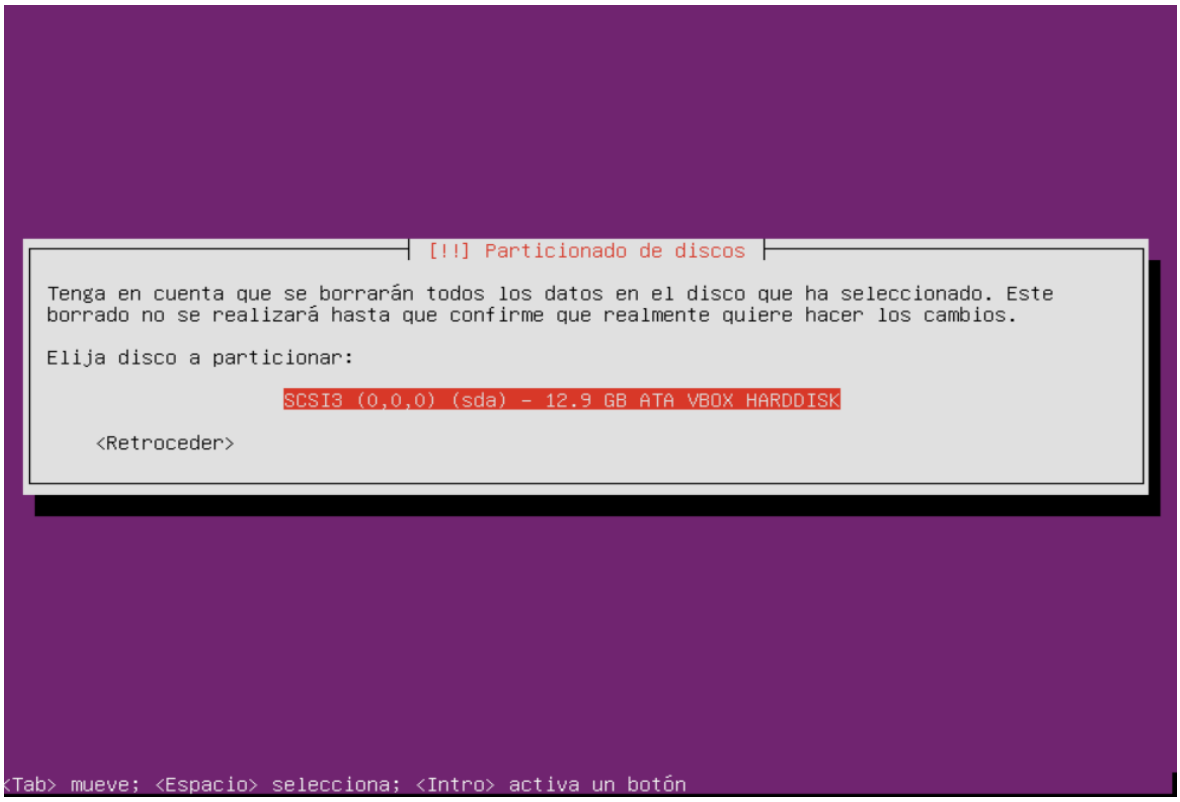


Figura 6.9. Selección de Disco.

Nos indica el volumen de disco a usar en este caso 12.1GB, presionamos *enter* en la opción continuar.

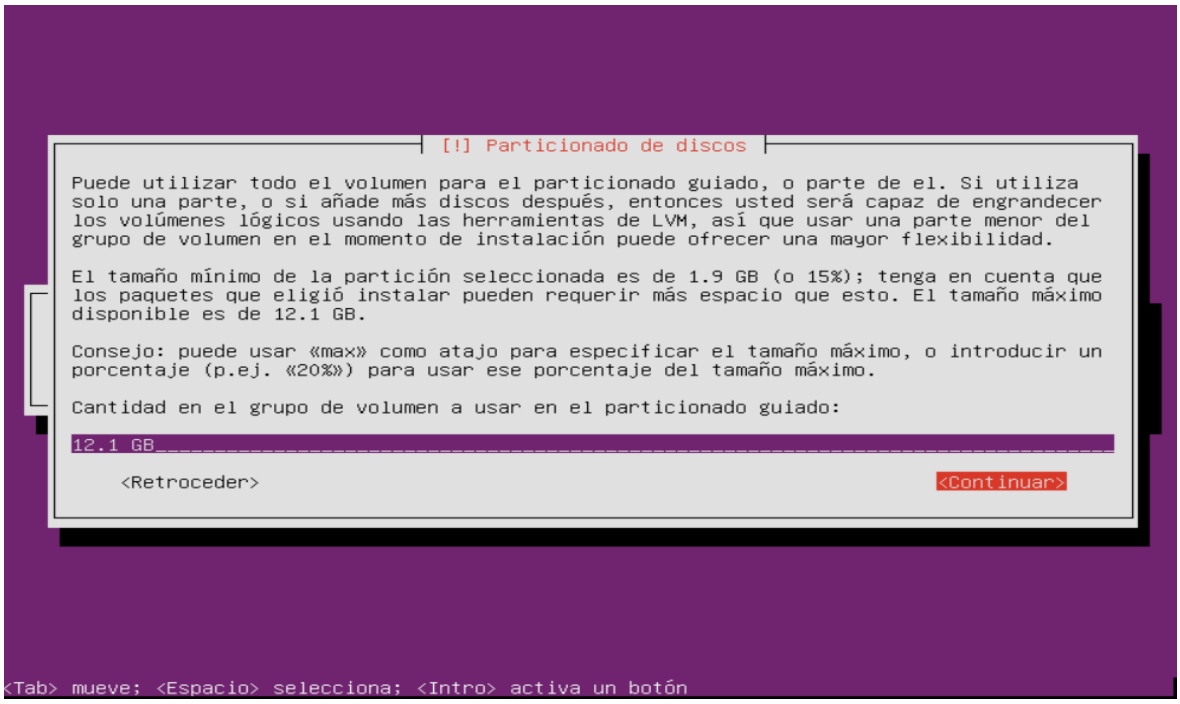


Figura 6.10. Partición de volumen de disco.

En esta opción se particionan los el disco con detalle de la siguiente imagen, presionamos *enter* en la opción *Si*.

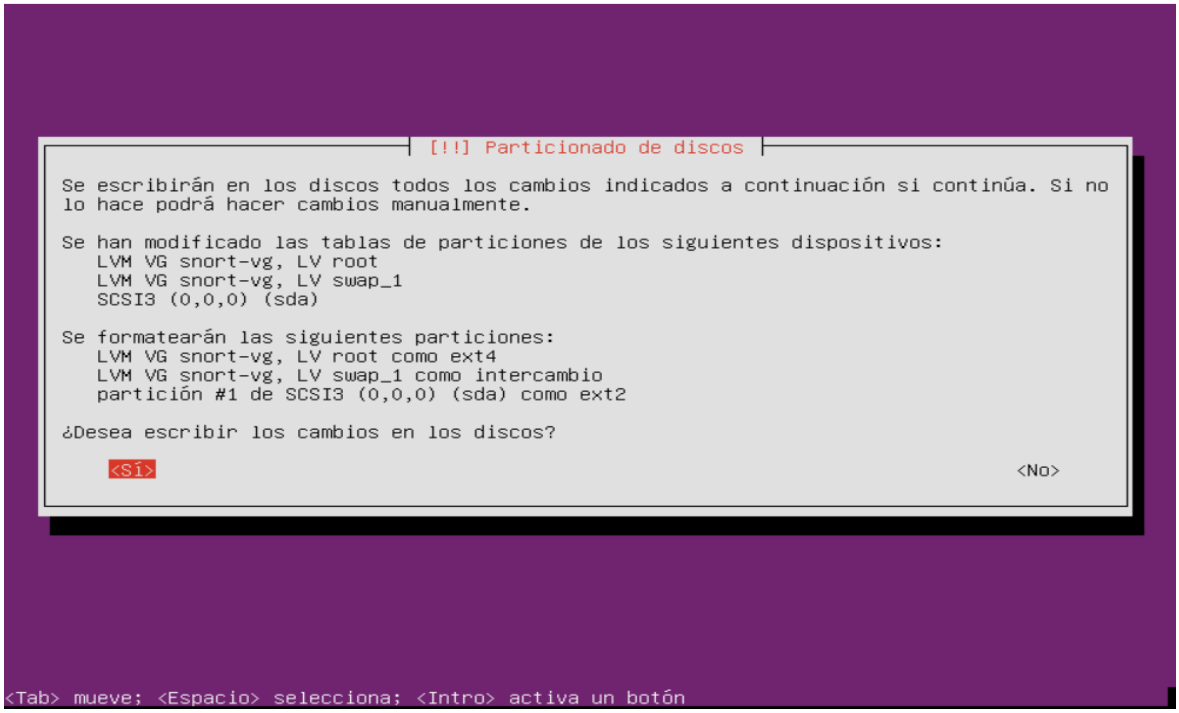


Figura 6.11. Escribir cambios en el disco.

Elegimos la opción sin configuraciones automáticas, para que la versión 16.04 de Ubuntu Server siga siendo la misma.

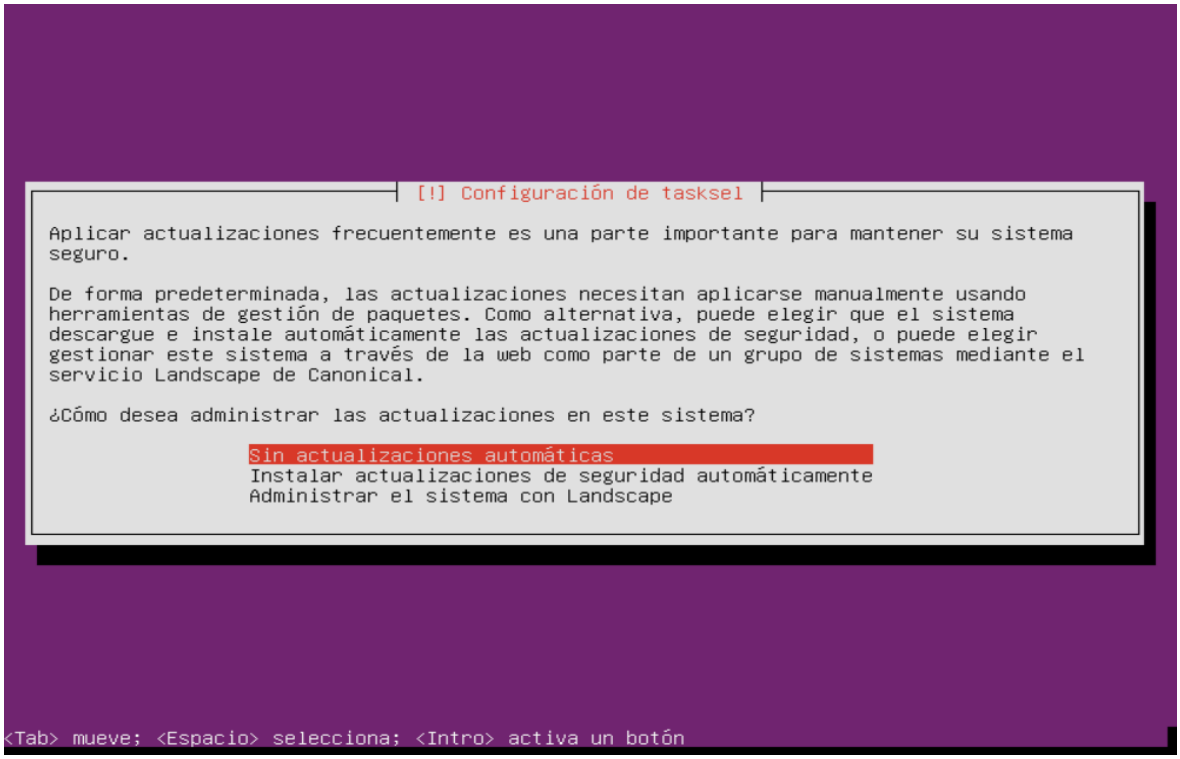


Figura 6.12. Configuración de taskel.

En esta opción instalaremos el cargador de arranque GRUB, para ello damos *enter* en *Si*.

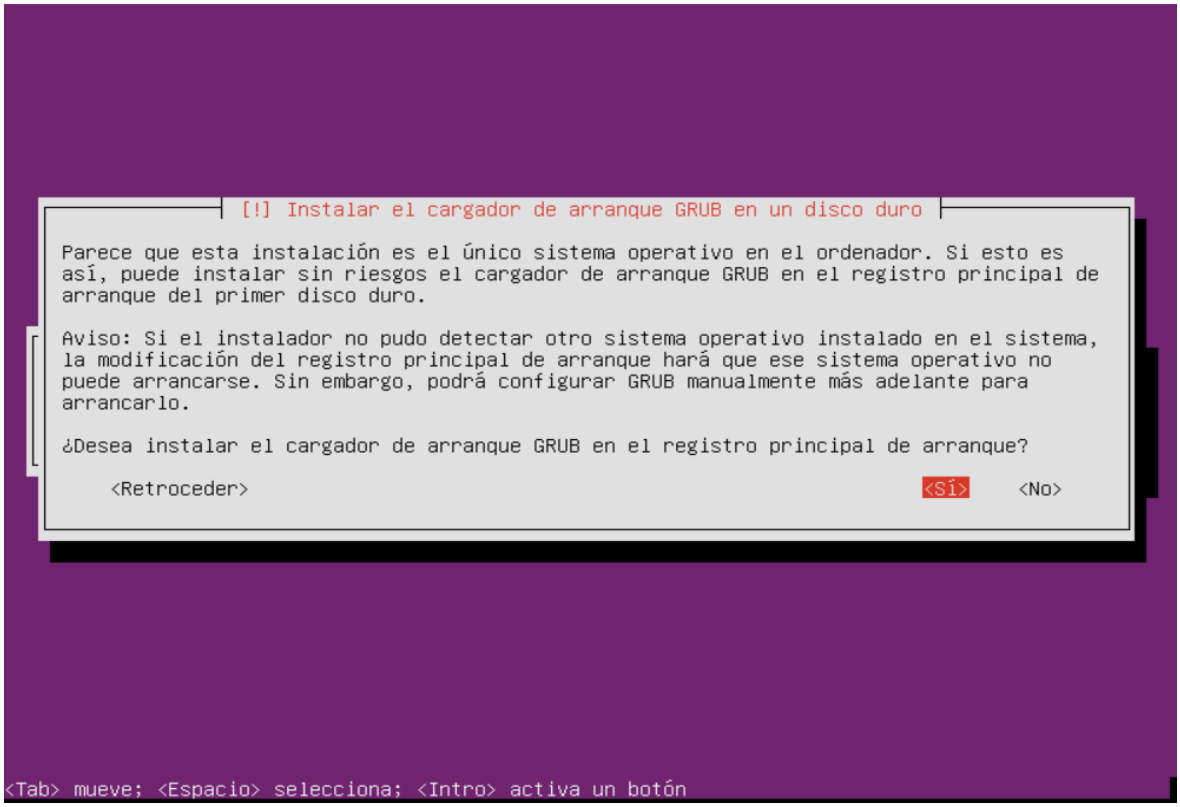


Figura 6.13. Instalar el cargador de arranque GRUP.

Ya finalizamos la instalación de Linux Ubuntu Server 16.04. Reiniciamos el equipo para poder utilizarlo.

Figura 6.14. Finalización de instalación.

ANEXO B

OBTENCIÓN DE REGLAS DE USUARIO REGISTRADO

También se puede registrarse en el sitio web de Snort. El registro da acceso para usar su código Oink para descargar las reglas de usuario registrado. Puede encontrar el código en los detalles de la cuenta de usuario de Snort.

Figura 6.15. Registrar Usuario

Ingresamos el Email con el que deseamos registrarnos e ingresamos la contraseña y la confirmación de la misma.

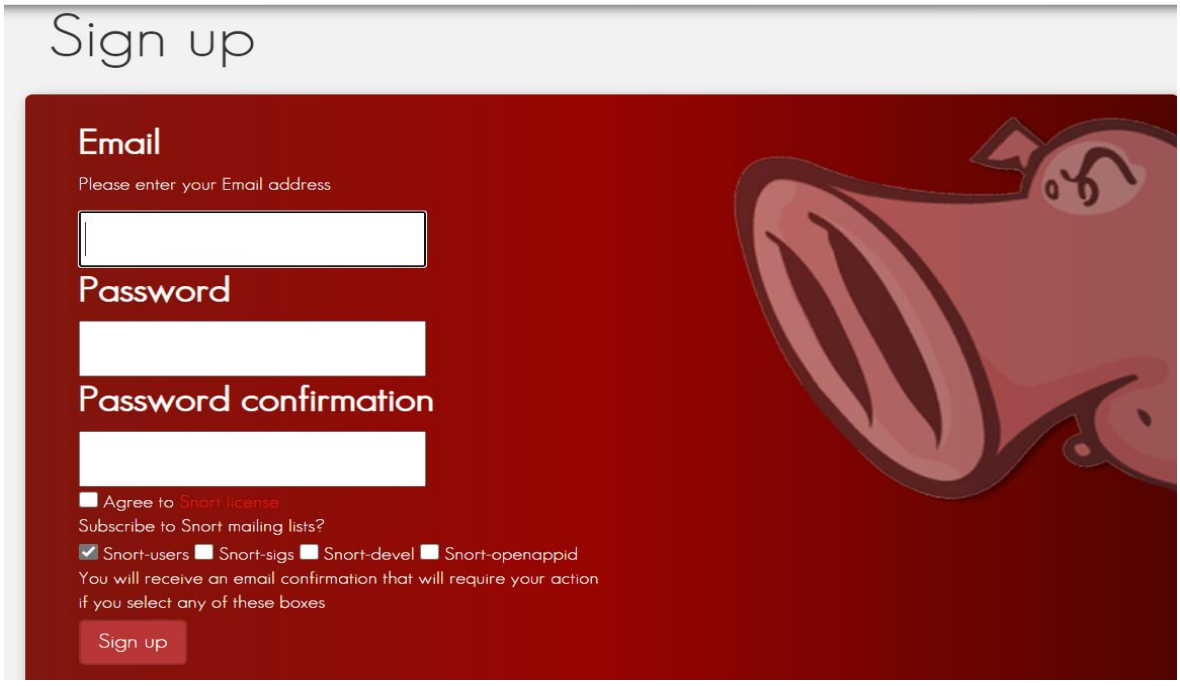


Figura 6.16. Ingresar datos de usuario

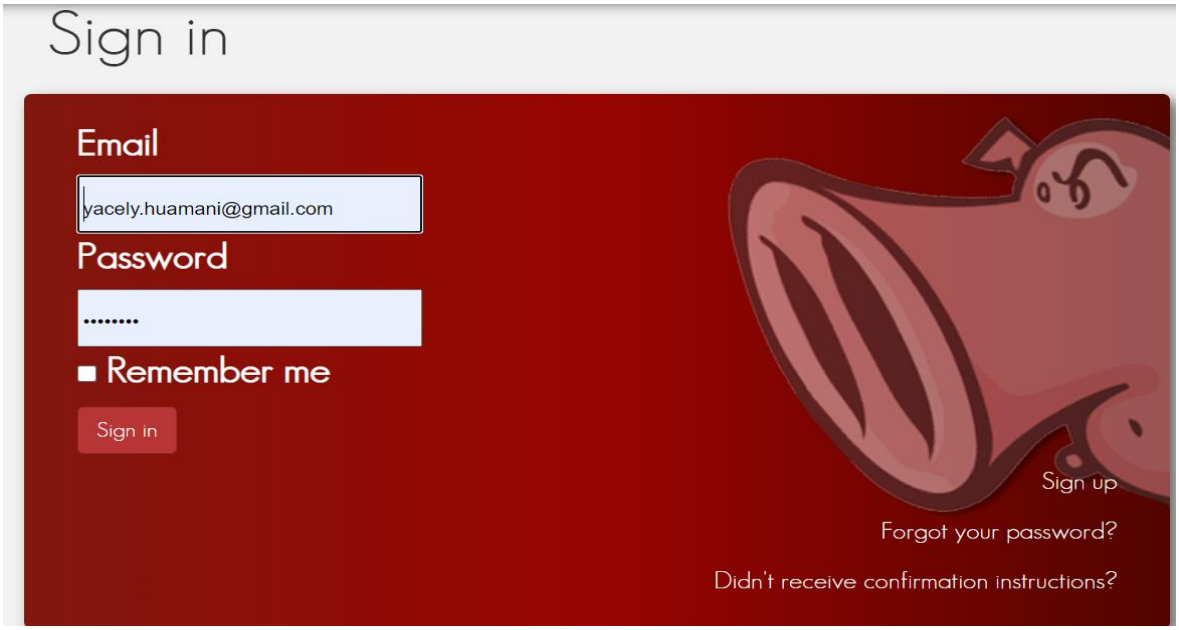


Figura 6.17. Inicio de sesión

Da el acceso para usar su código Oink para descargar las reglas de usuario registrado.

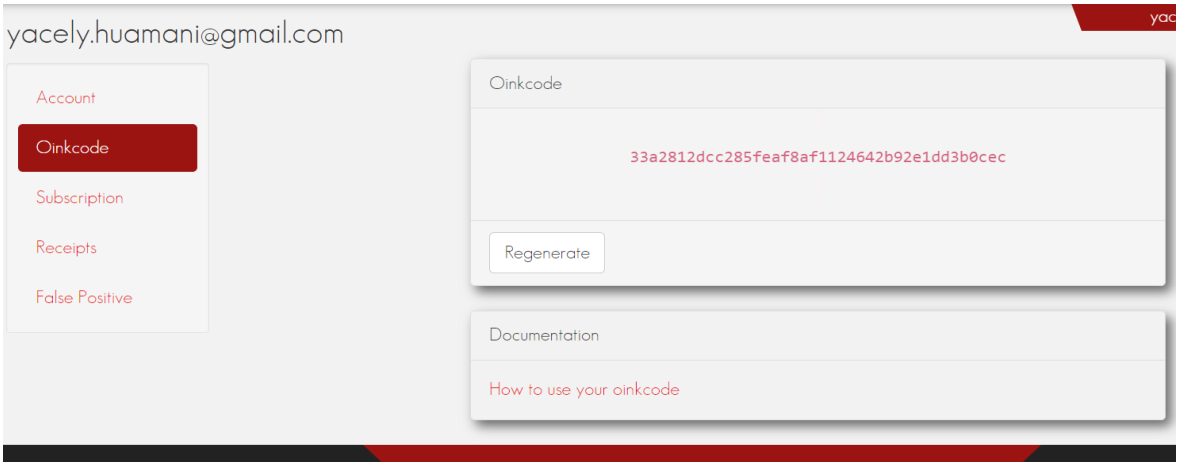


Figura 6.18. Código Oinkcode.

Reemplazamos el código oink, en el siguiente comando con su código personal.

```
wget https://www.snort.org/rules/snortrules-snapshot-29161.tar.gz?oinkcode=oinkcode -O ~/registered.tar.gz
```

Los conjuntos de reglas para los usuarios registrados incluyen una gran cantidad de útiles reglas de detección preconfiguradas. Si primero probó Snort con las reglas de la comunidad, puede habilitar reglas adicionales descomentando sus inclusiones hacia el final del archivo snort.conf . Una vez descargado, se extrae las reglas a su directorio de configuración.

```
sudo tar -xvf ~/registered.tar.gz -C /etc/snort
```

ANEXO C

28/8/2020 Consulta RUC: versión Imprimible

CONSULTA RUC: 20495157157 - MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C.

Número de RUC:	20495157157 - MULTINEGOCIOS & TRANSP. D.S HUAMANI S.A.C.		
Tipo Contribuyente:	SOCIEDAD ANONIMA CERRADA		
Nombre Comercial:	-		
Fecha de Inscripción:	25/01/2011	**Fecha Inicio de Actividades:**	21/02/2011
Estado del Contribuyente:	ACTIVO		
Condición del Contribuyente:	HABIDO		
Dirección del Domicilio Fiscal:	MZA. B LOTE. 11 BQ LOS OLIVOS (ESPALDAS DE CHECCOWASI) AYACUCHO - HUAMANGA - AYACUCHO		
Sistema de Emisión de Comprobante:	MANUAL	**Actividad de Comercio Exterior:**	SIN ACTIVIDAD
Sistema de Contabilidad:	COMPUTARIZADO		
Actividad(es) Económica(s):	Principal - 4923 - TRANSPORTE DE CARGA POR CARRETERA Secundaria 1 - 4620 - VENTA AL POR MAYOR DE MATERIAS PRIMAS AGROPECUARIAS Y ANIMALES VIVOS		
Comprobantes de Pago c/aut. de impresión (F. 806 u 816):	FACTURA BOLETA DE VENTA LIQUIDACION DE COMPRA GUIA DE REMISION - REMITENTE GUIA DE REMISION - TRANSPORTISTA		
Sistema de Emisión Electrónica:	FACTURA PORTAL DESDE 19/05/2018 BOLETA PORTAL DESDE 28/05/2018		
Afiliado al PLE desde:	-		
Padrones :	NINGUNO		

Imprimir

Figura 6.19. Ficha de la Empresa MULTINEGOCIOS & TRANS. D.S HUAMANI SAC.

ACTA DE SUSTENTACIÓN DE TESIS N° 020-2020-FIMGC

En la ciudad de Ayacucho, en cumplimiento a la **Resolución Decanal N° 145-2020-FIMGC-D,** siendo los diecisiete días del mes de setiembre del 2020, a horas 10.00 a.m.; se reunieron los jurados del acto de sustentación, en el Auditórium virtual google meet del Campus Universitario de la Universidad Nacional de San Cristóbal de Huamanga.

Siendo el Jurado de la sustentación de tesis compuesto por el Presidente el **Dr. Ing. Efraín Elías PORRAS FLORES**, Jurado el **Ing. José Antonio GUERRERO HINOSTROZA,** Jurado el **Mg. Ing. Hubner JANAMPA PATILLA,** Jurado el **Mg. Ing. Eloy VILA HUAMÁN** y Secretario del proceso **Ing. Christian LEZAMA CUELLAR**, con el objetivo de recepcionar la sustentación de la tesis denominada **"Snort Open Source como sistema de detección de intrusos para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020",** sustentado por la Bach. **Hayde Luisa HUAMANI SANTIAGO, bachiller en Ingeniería de Sistemas;** siendo el Asesor de la tesis el **Mg. Ing. Hubner JANAMPA PATILLA.**

El Jurado luego de haber recepcionado la sustentación de la tesis y realizado las preguntas, el sustentante al haber dado respuesta a las preguntas, y el Jurado haber deliberado; califica con la nota aprobatoria de **15 (Quince).**

En fe de lo cual, se firma la presente acta, por los miembros integrantes del proceso de sustentación.

Firmado digitalmente por Dr. Ing. Efrain Elias Porras Flores
Fecha: 2020.09.17 11:46:41 -05'00'

Dr. Ing. Efraín Elías PORRAS FLORES
Presidente

Ing. José Antonio GUERRERO HINOSTROZA
Jurado

Mg. Ing. Hubner JANAMPA PATILLA
Jurado - Asesor

I Mg. Ing. Eloy VILA HUAMÁN
Jurado

Ing. Christian LEZAMA CUELLAR
Secretario del Proceso

c.c.:
Bach. Hayde Luisa HUAMANI SANTIAGO
Jurados (4)
Archivo

UNSCH FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

CONSTANCIA DE ORIGINALIDAD DE TRABAJO DE INVESTIGACIÓN

El que suscribe; responsable verificador de originalidad de trabajos de tesis de pregrado en segunda instancia para las Escuelas Profesionales de la Facultad de Ingeniería de Minas, Geología y Civil; en cumplimiento a la Resolución de Consejo Universitario N° 039-2021-UNSCH-CU, Reglamento de Originalidad de Trabajos de Investigación de la UNSCH y Resolución Decanal N° 158-2021-FIMGC-UNSCH-D, deja constancia que:

- Apellidos y Nombres del Bach. : Huamaní Santiago Hayde Luisa
- Escuela Profesional : Ingeniería De Sistemas
- Título de la Tesis : Snort Open Source como sistema de detección de intrusos para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020.
- Evaluación de la originalidad : 24 % de similitud

Por tanto, según los artículos 12, 13 y 17 del Reglamento de Originalidad de Trabajos de Investigación, ***es procedente otorgar la constancia de originalidad*** para los fines que crea conveniente.

Ayacucho, 05 de junio del 2021

CCatamayo

Firmado digitalmente por Mg. Ing. Johnny Henrry Ccatamayo Barrios
Fecha: 2021.06.05 05:57:16 -05'00'

Mg. Ing. Ccatamayo Barrios Johnny Henrry
Verificador de originalidad de trabajos de tesis de pregrado de la FIMGC

Numero de constancia: 043-2021-FIMGC.

FACULTAD DE INGENIERÍA
DE MINAS, GEOLOGÍA Y CIVIL
Av. Independencia S/N
Ciudad Universitaria
Central Tel 066 312510
Anexo 151

“Snort Open Source como sistema de detección de intrusos para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020”

por Hayde Luisa Huamaní Santiago

Fecha de entrega: 04-jun-2021 08:59a.m. (UTC-0500)
Identificador de la entrega: 1600394909
Nombre del archivo: TESIS_HAYDE_LUISA_HUAMANI_SANTIAGO_EPIS.docx (6.48M)
Total de palabras: 20635
Total de caracteres: 111886

“Snort Open Source como sistema de detección de intrusos para la seguridad de la infraestructura de red en entornos libres aplicado a Pymes del Perú, 2020”

INFORME DE ORIGINALIDAD

24%	24%	2%	8%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	repositorio.unsch.edu.pe Fuente de Internet	3%
2	openaccess.uoc.edu Fuente de Internet	3%
3	dspace.espoch.edu.ec Fuente de Internet	1%
4	riunet.upv.es Fuente de Internet	1%
5	idoc.pub Fuente de Internet	1%
6	mendillo.info Fuente de Internet	1%
7	mtlsasturiasnoe.wordpress.com Fuente de Internet	1%
8	repositorio.itm.edu.co Fuente de Internet	1%

9	docplayer.es Fuente de Internet	1 %
10	tesis.ipn.mx Fuente de Internet	1 %
11	bascperu.org Fuente de Internet	1 %
12	core.ac.uk Fuente de Internet	1 %
13	madhu1108.files.wordpress.com Fuente de Internet	1 %
14	www.adminso.es Fuente de Internet	1 %
15	bibdigital.epn.edu.ec Fuente de Internet	<1 %
16	docshare.tips Fuente de Internet	<1 %
17	myslide.es Fuente de Internet	<1 %
18	tesis.ucsm.edu.pe Fuente de Internet	<1 %
19	mural.uv.es Fuente de Internet	<1 %
20	wiki.maite.sc Fuente de Internet	<1 %

21	Submitted to Universidad Pontificia Bolivariana Trabajo del estudiante	<1 %
22	cybertesis.uni.edu.pe Fuente de Internet	<1 %
23	gestion.pe Fuente de Internet	<1 %
24	andina.pe Fuente de Internet	<1 %
25	rodin.uca.es Fuente de Internet	<1 %
26	repositorio.unjbg.edu.pe Fuente de Internet	<1 %
27	Submitted to Universidad Nacional Abierta y a Distancia Trabajo del estudiante	<1 %
28	Submitted to Universidad Internacional de la Rioja Trabajo del estudiante	<1 %
29	es.scribd.com Fuente de Internet	<1 %
30	repositorio.espam.edu.ec Fuente de Internet	<1 %
31	repositorio.ucv.edu.pe Fuente de Internet	<1 %

32	repositorio.uss.edu.pe Fuente de Internet	<1 %
33	seguridadinformaticaufps.wikispaces.com Fuente de Internet	<1 %
34	www.buenastareas.com Fuente de Internet	<1 %
35	bibing.us.es Fuente de Internet	<1 %
36	es.slideshare.net Fuente de Internet	<1 %
37	www.slideshare.net Fuente de Internet	<1 %
38	Submitted to University of Hertfordshire Trabajo del estudiante	<1 %
39	stadium.unad.edu.co Fuente de Internet	<1 %
40	ciberseguridad.blog Fuente de Internet	<1 %
41	repositorio.urp.edu.pe Fuente de Internet	<1 %
42	lajc.epn.edu.ec Fuente de Internet	<1 %
43	documentop.com Fuente de Internet	<1 %

44	repositorio.unh.edu.pe Fuente de Internet	<1 %
45	silo.tips Fuente de Internet	<1 %
46	www.scribd.com Fuente de Internet	<1 %

Excluir citas	Activo	Excluir coincidencias	< 30 words
Excluir bibliografía	Activo		