UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

Maestría en
SEGURIDAD DE LA INFORMACIÓN
RPC-SO-28-NO.669-2021

Opción de Titulación:

Proyecto de titulación con componentes de investigación aplicada y/o de desarrollo

Tema:

INTEGRACIÓN DE SISTEMAS DE DETECCIÓN Y RESPUESTA ANTE INTRUSIONES EN LA "RED DE LA EMPRESA CONVERSA, CONVERTIDORA DE PAPEL S.A." MEDIANTE EL USO DE SOFTWARE LIBRE.

Autor:

CÉSAR GERMAN MARTINEZ MORA

Director:

JUAN CARLOS DOMÍNGUEZ AYALA

CUENCA – ECUADOR
2024

Autor:

César German Martinez Mora
Ingeniero de Sistemas.
Candidato a Magíster en Seguridad de la Información por la Universidad Politécnica Salesiana – Sede Cuenca.
cmartinezec@gmail.com

Dirigido por:

Juan Carlos Domínguez Ayala
Ingeniero en Sistemas.
Magister en Redes de Comunicaciones.
jdominguez@ups.edu.ec

Todos los derechos reservados.

Queda prohibida, salvo excepción prevista en la Ley, cualquier forma de reproducción, distribución, comunicación pública y transformación de esta obra para fines comerciales, sin contar con autorización de los titulares de propiedad intelectual. La infracción de los derechos mencionados puede ser constitutiva de delito contra la propiedad intelectual. Se permite la libre difusión de este texto con fines académicos investigativos por cualquier medio, con la debida notificación a los autores.

DERECHOS RESERVADOS
2024 © Universidad Politécnica Salesiana.
CUENCA – ECUADOR – SUDAMÉRICA
CÉSAR GERMAN MARTINEZ MORA
Integración de sistemas de detección y respuesta ante intrusiones en la “Red de la Empresa Conversa, Convertidora de Papel S.A.” mediante el uso de software libre.

DEDICATORIA

Dedico este trabajo a mi Esposa mis Hijas y mi Madre que han sido pilares fundamentales en el desarrollo de esta Maestría, brindándome siempre el apoyo y preocupación para finalizar esta Maestría.

AGRADECIMIENTO

Agradezco a toda mi Familia por el apoyo en el desarrollo de la esta Maestría, a los profesores que impartieron sus conocimientos en esta Maestría, a mi director de Tesis Juan Carlos Ayala por su ayuda y sus guías para finalizar este trabajo.

Tabla de Contenido

Resumen 7
Abstract 8
1. Introducción 9
2. Determinación del Problema 10
3. Marco teórico referencial 103
3.1 Objetivos 13
3.2 Cronograma de actividades ..... 1**¡Error! Marcador no definido.**
4. Materiales y metodología 33
5. Conclusiones 59
Referencias 60

Integración de sistemas de detección y respuesta ante intrusiones en la “red de la empresa CONVERSA, CONVERTIDORA DE PAPEL S.A.” mediante el uso de software libre.

Autor(es):

CESAR GERMAN MARTINEZ MORA

Resumen

El aumento de los ciberataques y las vulnerabilidades que presentan los sistemas han motivado el crecimiento del uso de herramientas de seguimiento y respuesta.

Ante posibles incidentes, esto permitirá tomar medidas antes de que un ataque se convierta en un problema y detenga las actividades normales de una organización.

Existen varios Software para este tipo, pero muchas empresas no cuentan con el presupuesto suficiente para contratar o implementar estas soluciones, es por esto que el uso del Software Libre puede ayudar a estas organizaciones a implementar soluciones a bajo costo y efectividad como herramienta de pago sin perder la efectividad que puedan tener los sistemas de pago.

Palabras clave:

IDS, Snort, Linux, Snort, Snortby

Abstract

The increase in cyberattacks and system vulnerabilities have motivated the growth in the use of monitoring and response tools.

In the event of possible incidents, this will allow action to be taken before an attack becomes a problem and stops the normal activities of an organization.

There are several Software for this type, but many companies do not have enough budget to hire or implement these solutions, which is why the use of Free Software can help these organizations implement solutions at low cost and effectiveness as a payment tool without lose the effectiveness that payment systems may have.

1. Introducción

Las redes de datos de las empresas se encuentran expuestas a muchos ataques informáticos por lo que es necesario implementar una seguridad para proteger los Activos de la empresa. [2]

Con el avance de la tecnología existen muchas formas de mejorar y/o prevenir estos ataques a través de un sistema de detección temprana que sea capaz de notificar al(los) administrador(es) de la red en el momento exacto que se inicie de tal forma que puedan tomar las medidas para que este ataque no llegue a ser perjudicial para la organización y sus Activos.[2]

Los sistemas de detección de intrusos son una evolución directa de los primeros sistemas de auditorías. Estos sistemas tienen como finalidad detectar en una red accesos no autorizados y/o ataques, a su vez emitir una alerta al(los) administrador(es) mediante un mail, mensaje de texto u otro método que esté disponible en la herramienta, además almacenará estos ataques en un log para un posterior análisis por parte de los encargados.[6] [11] [13]

Es importante tomar en cuenta que muchos de los ataques no solo provienen de forma externa (internet) sino también pueden generarse en la misma red, ya sea por un(os) equipo(s) infectado(s) con algún virus, troyano u otro similar, es por esto por lo que el control no debe ser solo por parte de un cortafuegos perimetral sino también de un antivirus en los equipos que permita realizar un bloqueo de estos ataques. [6] [11] [13]

Una ventaja de usar un prototipo basado en Software Libre permite a las organizaciones que no disponen de mucho presupuesto implementar una seguridad de calidad sin tener que invertir fuertes cantidades de dinero por el uso de software licenciado.[9] [11]

2. Determinación del Problema

Un problema para todas las empresas grandes o pequeñas es mantener la seguridad de su información lo más protegida y segura, es por esto que existe software en el mercado que permite mantener un monitoreo y notificaciones en tiempo real sobre actividades sospechosas, con el objetivo de prevenir un acceso no autorizado a su red. La mayoría de las empresas no cuentan con un presupuesto para realizar este tipo de inversiones ya que su costo puede ser elevado, es por esto que existen herramientas basadas en software libre que permiten implementar este tipo de soluciones con un costo bajo y con la misma efectividad del software de pago.

Este proyecto nace de la necesidad de contar con un prototipo de un sistema que disponga de una funcionalidad de detección de ataques y/o intrusiones hacia red de la empresa CONVERSA S.A. utilizando al 100% software libre.

Luego de realizar una prueba de preguntas al personal encargado de TI en la empresa CONVERSA S.A. se pudo determinar que no existe ninguna política y/o norma de seguridad ante ataques definida, es por esto la importancia de elaborar un prototipo inicial que permitirá ser implementado en un futuro para fortalecer y mantener un esquema de monitoreo constante para mantener a la organización segura, esto permitirá que no se detenga la organización en su giro de negocio diario.

Este prototipo inicial servirá como base para continuar con el desarrollo de una herramienta basada en software libre que permita

El poder tener un prototipo inicial propio para integrar un sistema de detección y respuesta ante intrusiones (IDS/IPS) para la empresa CONVERSA permitirá responder de manera efectiva a las amenazas y actividades maliciosas que puedan afectar la seguridad de la información y los sistemas con los que cuenta la organización. Estos sistemas desempeñan un papel crucial en la seguridad de la red y la infraestructura, ya que ayudan a identificar y mitigar posibles intrusiones y actividades sospechosas en tiempo real, algunos beneficios de este tipo de herramientas son:

La detección temprana de intrusiones: permiten monitorear constantemente el tráfico de red y los eventos del sistema en busca de patrones y comportamientos anómalos que puedan indicar de una posible intrusión o ataque. La detección temprana permite una respuesta rápida y efectiva antes de que los atacantes puedan causar daños significativos en las organizaciones.

Protección proactiva: Los sistemas IDS/IPS permiten identificar vulnerabilidades y posibles amenazas de seguridad en la infraestructura y los sistemas. Esto permite tomar medidas preventivas para mitigar los riesgos y cerrar las brechas de seguridad antes de que puedan ser explotadas por los atacantes.

Protección contra amenazas conocidas y desconocidas: Los IDS pueden identificar patrones de tráfico y comportamientos asociados con ataques conocidos, así como detectar actividades desconocidas que podrían indicar nuevas amenazas.

Respuesta automática: Los sistemas IPS pueden implementar respuestas automáticas en tiempo real cuando se detectan intrusiones o actividades maliciosas. Esto incluye bloquear direcciones IP, cerrar puertos, desactivar cuentas de usuario sospechosas, entre otras acciones inmediatas para detener o evitar rápidamente que este ataque pueda llegar a causar una para total en los sistemas por un corto o largo tiempo.

Análisis forense: Los datos registrados por el IDS en sus logs pueden utilizarse para realizar análisis forenses posteriores a un incidente, lo que facilita la investigación de intrusiones y correlaciones que puedan indicar una intrusión. Esto ayuda a tener una visión más completa de la situación y a tomar decisiones anticipadas para evitar futuros ataques.

Cumplimiento normativo: La implementación de sistemas IDS/IPS puede ayudar a cumplir con ciertos requisitos de seguridad establecidos por regulaciones y estándares de la industria para un mejoramiento continuo u obtención de certificaciones que ayuden al crecimiento de organización.

Protección de la reputación: Al implementar un IDS y prevenir intrusiones exitosas una empresa protege su reputación al evitar la divulgación no autorizada de datos sensibles y la interrupción de sus servicios.

Mejora de la capacidad de respuesta: Al integrar sistemas IDS/IPS con otros componentes del Sistema de Gestión de Seguridad de la Información (SGSI), se mejora la capacidad de la organización para responder a incidentes de seguridad de manera coordinada y efectiva.

Monitoreo continuo: El IDS funciona las 24 horas del día, los 7 días de la semana, proporcionando una vigilancia ininterrumpida y eliminando la necesidad de una supervisión por parte de una persona de forma continua.

Ahorro de costos: La detección temprana y la respuesta rápida a las intrusiones pueden ayudar a minimizar el tiempo de inactividad de los sistemas lo que a su vez puede reducir los costos asociados con la recuperación y/o mitigación de incidentes que ya fueron ejecutados.

La mejora continua es un aspecto fundamental en la implementación y operación de sistemas de detección de intrusiones (IDS). Un sistema IDS no es una solución estática, ya que las amenazas y los riesgos de seguridad de la información están en constante evolución. Para asegurar que el IDS siga siendo efectivo y se mantenga al día con los cambios en el entorno de seguridad, se deben aplicar prácticas de mejora continua.

Es necesario aclarar que la implementación de un sistema IDS no involucra solamente la elaboración de un prototipo sino también a las respuestas inmediatas que el personal encargado pueda dar para evitar que los ataques sean ejecutados y puedan causar fallas graves al desarrollo de las actividades de la organización.

Realizar un prototipo 100% basado en software libre con el objetivo de monitorear una red de equipos internos o externos para mantener los servicios y poder actuar antes de que una amenaza se materialice en la empresa CONVERSA S.A.

1. Objetivos

Objetivo general

Integrar un prototipo de un sistema de detección y respuesta ante intrusiones en la red de la empresa CONVERSA S.A., utilizando software libre, con el fin de fortalecer la seguridad y protección de la red contra posibles ataques y vulnerabilidades.

Para llegar al objetivo específico se cuenta con el todo el apoyo del personal y máximas autoridades de la organización.

Objetivos específicos

Definir la línea base de ciberseguridad de la infraestructura de la red de CONVERSA S.A. identificando sus características, componentes y posibles puntos de vulnerabilidad.

1. Definir los conceptos básicos de detección de Intrusos, analizar las técnicas y herramientas existentes de detección y respuesta ante intrusiones basados en software libre.
2. Diseñar e implementar un prototipo de un sistema de detección de intrusiones adaptado a la red corporativa específica en software libre.
3. Realizar pruebas de funcionamiento y rendimiento de los sistemas de detección y respuesta ante intrusiones, evaluando su efectividad en la detección y respuesta a posibles amenazas.
4. Elaborar un informe detallado que documente todo el proceso de integración de los sistemas de detección y respuesta ante intrusiones en la red de CONVERSA S.A, incluyendo las herramientas utilizadas, configuraciones realizadas y recomendaciones para su mantenimiento actualización e implementación futura. Preparar Planificación Diseño (metodología de diseño Top-Down) Implementación (prototipo VM) Operación Optimización
5. Conclusiones, recomendaciones y trabajado futuros.

2. Cronograma de actividades

Mírese pdf adjunto para más detalles

Materiales y metodología

La implementación de un sistema de detección de intrusiones (IDS) requiere una metodología estructurada para garantizar una configuración adecuada y una integración efectiva con el entorno de la organización.

A continuación, los pasos a seguir para cumplir con esta metodología:

- Identificar los activos críticos las vulnerabilidades y las amenazas potenciales que el IDS deberá enfrentar. Esta evaluación de riesgos ayudará a priorizar las áreas que requieren mayor atención y enfoque en la compañía

- Selección del IDS adecuado: Investigar y seleccionar un IDS que se ajuste a los requisitos y necesidades de la organización. Existen varios tipos de IDS de red y sistemas de detección y la elección de uno dependerá del alcance y los objetivos del proyecto.

- Diseño de la arquitectura del IDS: Definir la arquitectura del IDS, incluyendo la ubicación de los sensores de detección (software), la configuración de los dispositivos, los puntos de monitoreo y las políticas de detección y acción.

- Implementación y configuración del prototipo: Instalar y configurar el software IDS seleccionado en una máquina virtual según el diseño de la arquitectura, asegurarse de que los sistemas de alerta para una cobertura efectiva y que los umbrales de alerta estén configurados de una forma correcta.

- De ser posible el sistema de detección IDS puede integrarse con otros sistemas de seguridad de la organización, como sistemas de prevención de intrusiones (IPS), firewalls y sistemas de gestión de eventos e información de seguridad (SIEM).

- Pruebas y ajustes: Realizar pruebas exhaustivas del IDS para asegurarse que esté funcionando correctamente y detectando las amenazas de manera efectiva. Realizar ajustes y refinamientos según sea necesario para llegar al objetivo general.

- Capacitación del personal: Capacitar al personal de seguridad y TI sobre cómo utilizar y administrar el IDS de manera efectiva. Esto incluye la capacitación en la interpretación de alertas y la respuesta a incidentes en ejecución.

- Implementación de políticas y procedimientos: Establecer políticas y procedimientos para el uso y manejo del IDS, incluyendo la gestión de alertas, la respuesta a incidentes y la revisión periódica del sistema.

Es importante recalcar que la implementación de un IDS es parte de una estrategia más amplia de seguridad de la información y debe estar integrada con otros controles y medidas de seguridad para obtener una protección completa y efectiva.

CAPITULO 1

Definir los conceptos básicos de detección de Intrusos, analizar las técnicas y herramientas existentes de detección y respuesta ante intrusiones basados en software libre.

Al contar con un firewall no estamos exentos o protegidos por completo de recibir ataques que no puedan ser detectados por un Firewall, este provee una seguridad de frontera o perimetral, el IDS permite detectar ataques que muchas veces un Firewall no lo hace, por ejemplo:

- Ataques de vulnerabilidades en aplicaciones
- Ataques desde la misma red como spoofing
- Ataques por entulenamiento de tráfico.

Un sistema de detección de intrusiones es una parte fundamental de la seguridad de la información en una organización se utiliza para identificar actividades maliciosas o no autorizadas en los sistemas y redes. Su función es similar a la de un sistema de alarma es decir notifica cuando algo no deseado ocurre.

Un IDS puede ser utilizado para detectar muchas formas de comportamiento malicioso que pueden comprometer la seguridad de una red y de sistemas que se utilizan en las Organizaciones, incluyendo ataques de red contra servicios vulnerables, explotación de datos, propagación de malware, etc.

Capacidades de un IDS

Los sistemas de detección de intrusos permiten monitorear y enviar alertas tempranas para evitar ataques a una red.

Los ids presentan muchas bondades y ventajas que permiten a los administradores tomar precauciones antes que los ataques lleguen a causar problemas en los equipos de la red.

Beneficios de un IDS

Usar un Sistema de Detección de Intrusos (IDS) tiene muchos beneficios para para la seguridad de una red y los equipos que están funcionando dentro de ella podemos mencionar algunos como:

Permite detectar amenazas en tiempo real que en un tráfico de red y alertar de los eventos del sistema que sean considerados como sospechosos y/o maliciosas con el objetivo de responder de forma inmediata a las amenazas en tiempo real.

Al generar registros (logs) puede identificar intrusiones o intentos de intrusiones esto permite a los administradores de seguridad a tomar medidas correctivas adecuadas para que no causen daños o que se detengan los servicios hacia los usuarios.

Los tiempos de respuesta son mínimos ya que permite alertar de forma inmediata sobre actividades sospechosas lo que minimiza el impacto y el alcance de un ataque que pueda ser fatal para una organización.

Herramientas

Todo IDS debe estar configurado para recibir una tolerancia a fallos o capacidad de respuesta para situaciones inesperadas y/o cambios que pueden producir en el entorno donde está funcionando, estos cambios pueden ser graduales o bruscos, para esto el IDS debe ser capaz de responder de una forma adecuada. Por ejemplo, un reinicio inesperado o forzado de varias máquinas o un intento de engaño hacia el IDS; este último punto es especialmente crítico ya que, si un atacante consigue modificar el comportamiento del IDS y él no se da cuenta de ello, la intrusión nunca será notificada, con estos dos posibles problemas puede implicar la falsa sensación de seguridad que produce un IDS que no genera ninguna alarma es un grave inconveniente para lograr sistemas seguros.

A continuación, un cuadro comparativo de herramientas IDS analizadas para este prototipo

SNORT

Características	Ventajas	Desventajas
Es uno de los más populares detectores de intrusos y se puede instalar en Equipos con Sistemas operativos Linux o Windows.	Herramienta de código abierto y bajo consumo de recursos Hardware y rendimiento de red	Menor capacidad de procesamiento en comparación con otros sistemas
Utiliza reglas para analizar el tráfico en busca de patrones y firmas conocidas de ataque	Permite una personalización y configuración flexible según las necesidades de una red	Cantidad de falsos positivos en comparación con otros sistemas
Los usuarios pueden crear 'firmas' basadas en las características de los nuevos ataques de red y enviarlas a la lista de correo de firmas de Snort, para que así todos los usuarios de Snort se puedan beneficiar.	Las reglas se pueden aplicar a los encabezados de la capa de red y transporte (IP, TCP, UDP, ICMP) o incluso a los encabezados de la capa de aplicación (FTP, HTTP, etc.) y permite integrar con una interfaz Web administrativa.	Para una adecuada detección depende de una correcta configuración y actualizaciones constantes
		Versiones de sistemas operativos limitados para recolectar análisis desde otros sensores.
Permite el almacenamiento de las bitácoras en bases de datos más populares como MySQL Mariadb	Amplia comunidad de usuarios y desarrolladores que aportan conocimientos, reglas y mejoras al producto	No ofrece características de inspección profunda de paquetes (DPI)
Desarrollado por Cisco Systems bajo licencia GPLv2 y comercial	Permite la integración con la mayoría de IPS, Firewall y otros sistemas de seguridad.	Puede generar muchos falsos positivos que si no son analizados de forma correcta el atacante puede esconderse detrás.
Permite usar múltiples sensores con reglas para el monitoreo muy encillas y efectivas de implementar.	Ofrece una serie de reglas y filtros ya predefinidos que se pueden ajustar durante su instalación y configuración	
Puede funcionar como un sniffer es decir permite ver por la consola en tiempo real lo que sucede en la red.	Provee de cientos de filtros o reglas para backdoor, ddos, finger, ftp, ataques web, CGI, escaneos Nmap entro otros.	Incapacidad de detener o bloquear ataques

SURICATA

Características	Ventajas	Desventajas
Fue desarrollado como un reemplazo moderno par Snort	Esta diseñado para ser altamente eficiente y escalable	Requiere un mayor consumo de hardware puede ser un problema para empresas con recursos limitados.
Es de código abierto y propiedad de una comunidad OISF sin fines de lucro, su código fuente esta licenciada bajo GPLv2	Permite manejar grandes volúmenes de trafico de red sin afectar significativamente el rendimiento del sistema	La curva de aprendizaje puede ser más compleja para usuarios menos experimentados que administren este sistema
Es Multi-threading que permite ejecutar varios procesos y subprocesos de forma simultanea	Utiliza reglas y firmas para detectar amenazas, incluyendo intrusiones, malware, ataque de denegación de servicio, escaneo de puertos y otros	La comunidad de usuarios y la disponibilidad de reglas son menores en comparación con otras herramientas disponibles
Muestra estadísticas de rendimiento con un módulo que permite llevar un conteo de variada información y presentarlos como estadística al administrador en un formato log.	Realiza una inspección profunda de paquetes de red lo que permite analizar el contenido y las características de los paquetes incluyendo protocolos de aplicación específicos como http, Smtp, Ftp, dns, entro otros.	No posee una interfaz administrativa
		Bajo nivel de detección para nuevos tipos de ataques.
		Dificultad de configuración ya que requiere ser detallada y precisa para funcionar
Escalable, esto permite que el Hardware pueda alcanzar hasta 10 gigabit de velocidad de tráfico real, sin sacrificar la cobertura del conjunto de reglas creadas.	Permite el intercambio de información con otras aplicaciones en tiempo real y soporta protocolo ipv6	Tiene una comunidad activa y una documentación amplia, pero no cuenta con soporte oficial directo, lo que puede ser un problema para usuarios que necesitan ayuda rápida y efectiva.
Identificación del archivo, sumas de verificación MD5 y archivos de extracción para validación.	Compatible con reglas de snort	No se integra directamente con otros sistemas de seguridad, puede requerir configuraciones adicionales para implementar una solución de seguridad completa.

AIDE (Advanced Intrusion Detection Environment)

Características	Ventajas	Desventajas
Está basado en host que se centra en la monitorización de la integridad de los archivos y directorios de un sistema	Realiza un seguimiento de cambios en los archivos del sistema incluyendo adiciones, modificaciones y eliminaciones	Enfoque limitado a la detección de cambios en archivos y directorios, no cubre otros tipos de ataques
Compara las características y propiedades de los archivos con una base de datos de referencia para detectar cambios no autorizados	Calcula y almacena resúmenes criptográficos (hashes) de los archivos monitoreados durante la inicialización, luego compara los resúmenes almacenados en tiempo real para verificar la integridad de los archivos	Puede generar un gran volumen de alertas falsas, lo que puede ser desalentador para los administradores de seguridad.
Se basa en cambios de archivos y configuraciones lo que permite ser usado como un complemento a otros sistemas de detección de intrusiones y medidas de seguridad para una protección más completa.	Envía alertas y registros a los sistemas de gestión de eventos e información de seguridad (SIEM), lo que permite una mejor visibilidad y correlación de eventos de seguridad en toda la infraestructura.	No puede detectar amenazas recientemente descubiertas (0-day) ya que utiliza firmas de intrusión previamente conocidas.
		Requiere un mantenimiento regular para asegurarse de que las firmas de intrusión estén actualizadas y funcionen correctamente.
Puede ayudar a identificar problemas o errores de seguridad en la red o en los dispositivos.	Ofrece una gran flexibilidad en la configuración de las políticas de monitoreo	No proporciona detección en tiempo real de actividades maliciosas o ataques en curso
Soporta algoritmos de validación Md5, sha1, rmd160 entre otros	Es capaz de generar informes detallados sobre los cambios detectados en los sistemas de archivos.	Requiere configuración y mantenimiento adecuado de la base de datos de referencia.

Es importante contar una distribución que sea compatible, estable para la implementación del prototipo por lo que se analiza algunas distribuciones basadas en Kernel de Linux para realizar la implementación de este prototipo.

Distribución	Snort 2.9.20	Suricata	Aide	Apache	Mariabd 10.3	Ruby 2.9	Librerías adicionales	Snortby
Ubuntu 20.04	Si	Si	Si	Si	Si	No	No	Si
Ubuntu 22.04	Si	Si	Si	Si	Si	No	No	No
Debian 11	Si	Si	Si	Si	Si	No	No	Si
Debian 12	Si	Si	Si	Si	Si	No	No	No
Centos 7	Si	Si	Si	Si	Si	Si	Si	Si
Rocky/Alma 8	No	Si	Si	Si	Si	No	No	No

Una vez realizado el análisis se determina que la mejor distribución para este prototipo es Centos7, adicional al sistema operativo se utilizará otras herramientas de software libre entre las cuales podemos detallar las siguientes

- Máquina virtual basada en KVM
- Base de datos Mariadb 10.3
- Servidor web Apache
- Lenguaje de programación Ruby, Rvm, Passenger
- Herramientas Snort, Barnyard2, Snorby

Centos 7 es un sistema operativo de código abierto, basado en la distribución Red Hat Enterprise Linux

Snort es un software ampliamente utilizado por administradores de redes de todos los tamaños y se considera una parte sólida de la infraestructura de seguridad de la red, se puede utilizar en una variedad de escenarios para proteger las redes de amenazas cibernéticas.

Mariadb es un sistema de base de datos que proviene de MySQL, pero con licencia GPL, se trata de una “sustituto” de MySQL que corre bajo una licencia GPL.

Apache http Server es un software de servidor web gratuito y de código abierto para plataformas Unix.

Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos

Rvm es un manejador de versiones para Ruby

Passenger es un servidor de aplicaciones web que se integra con Apache y/o Nginx

Barnyard2 es una herramienta diseñada para procesar y analizar los registros de tráfico de red generados por el sistema de detección de intrusiones (IDS) de Snort.

Snorby es una interfaz que se suele instalar en Snort esta interfaz es de acceso web muy, fácil de gestionar para los administradores.

Config Security and Firewall (CSF) es un firewall avanzado y gratuito para la mayoría de las distribuciones de Linux

CAPITULO 2

Diseñar e implementar un prototipo de un sistema de detección de intrusiones adaptado a la red corporativa específica en software libre.

Para la implementación del prototipo se realizó un análisis de la red para proponer la mejor ubicación del prototipo, se llegó a la conclusión que la mejor ubicación es después del Firewall, esto permitirá que el tráfico que no sea detectado por el Firewall sea considerado como malicioso, en un alto porcentaje de los casos la posibilidad de falsas alarmas o falsos positivos será menor y tendremos alarmas más reales.

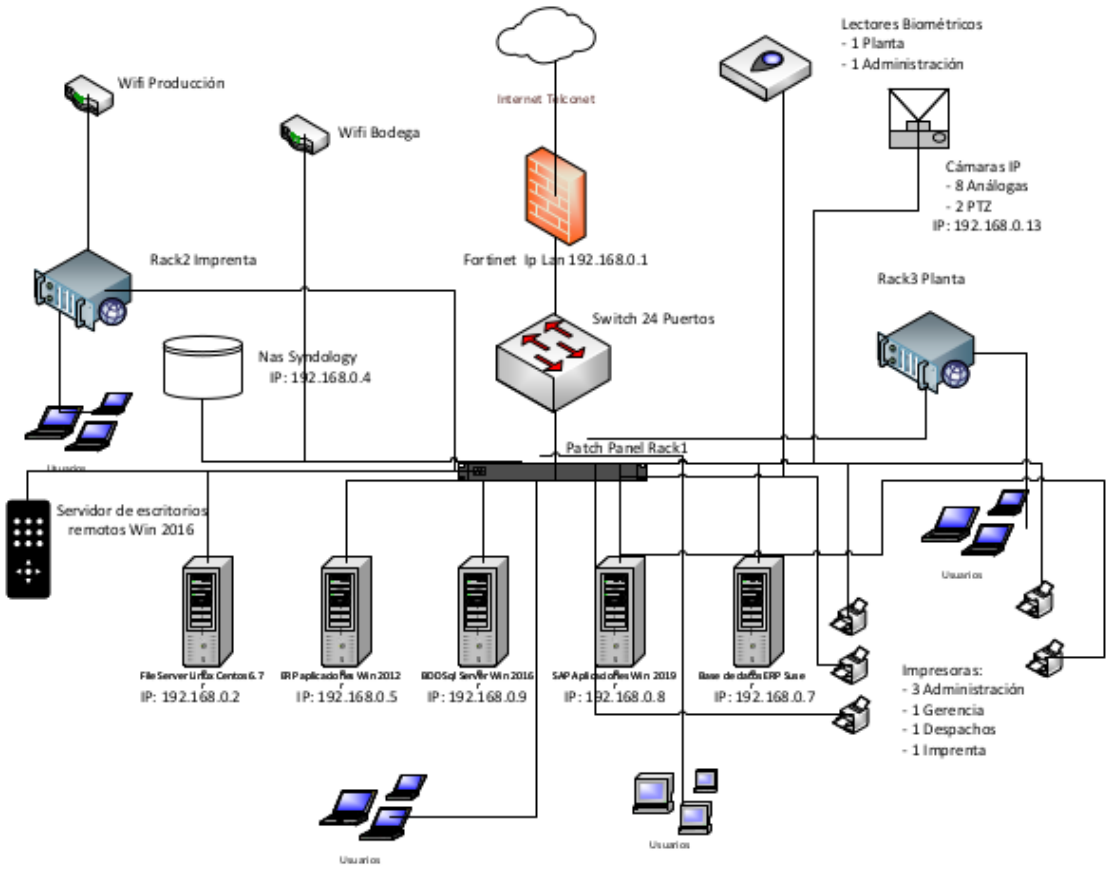


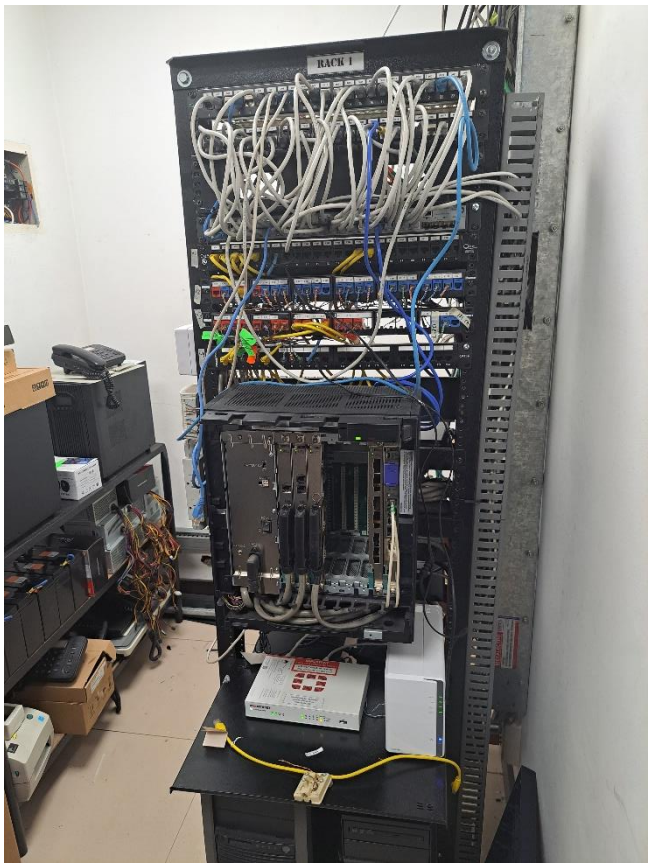
Imagen Esquema de red Conversa S.A.

Detalle de Infraestructura

- El servicio de internet provee la empresa Telconet la velocidad contrata es 15 MB plan corporativo simétrico 1/1 con un backup de 5 MB por medio de radio enlace con una compartición de 2/1.
- La seguridad interna y externa es mediante un firewall Fortinet quien se encarga de los servicios:
    - Dhcp para equipos
    - Dns local
    - Filtrado de contenido para navegación de páginas
    - Vpn para usuarios remotos y conexiones entre una sucursal.

Después del switch el servicio de internet llega a una switch marca Hp modelo V14910-48G, donde se conectan todos los equipos de la red.

- El cableado de la red funciona a 1000 Gbps
- Existen 3 racks adicionales en otros edificios desde donde se provee el servicio, estos están conectados directamente al switch principal en el rack principal.
- Se cuenta con 7 servidores con sistemas operativos Linux y Windows que brindan distintos servicios a los usuarios, así como un sistema de backups para almacenar el respaldo de la información generada por los usuarios y sistemas internos.
- Como equipos periféricos cuentan con 2 equipos biométricos, 10 cámaras de video, 2 equipos Wifi y 6 impresoras distribuidas en las distintas zonas administrativas y de producción.
- Todos los viernes el proveedor de internet envía reportes del tráfico y más detalles generados por el firewall al correo de la persona encargada del área de IT



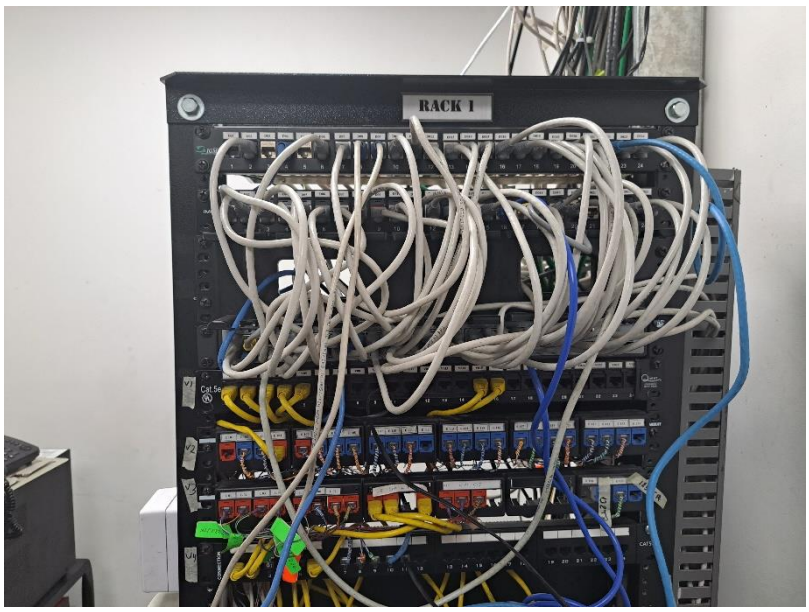


Imagen Rack principal

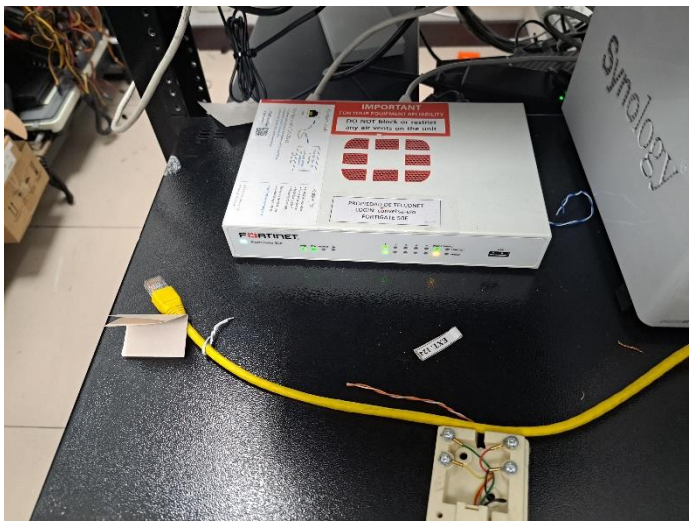


Imagen Firewall Fortinet

Imagen Conexión Fibra Internet

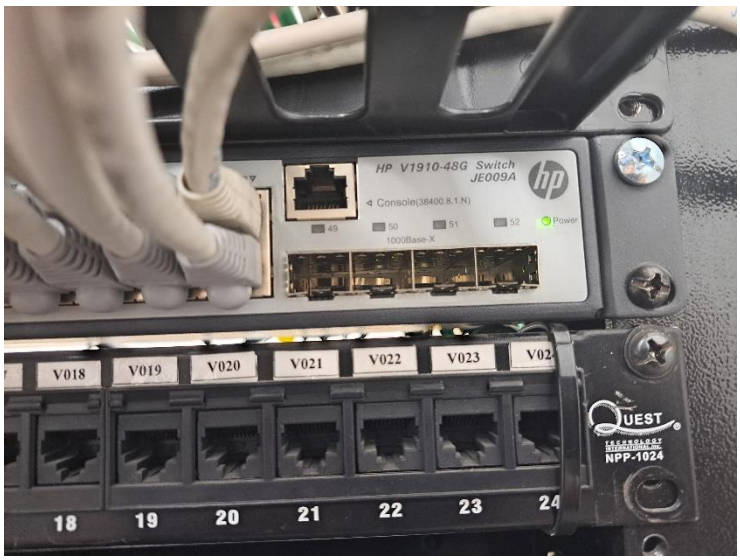


Imagen Switch principal

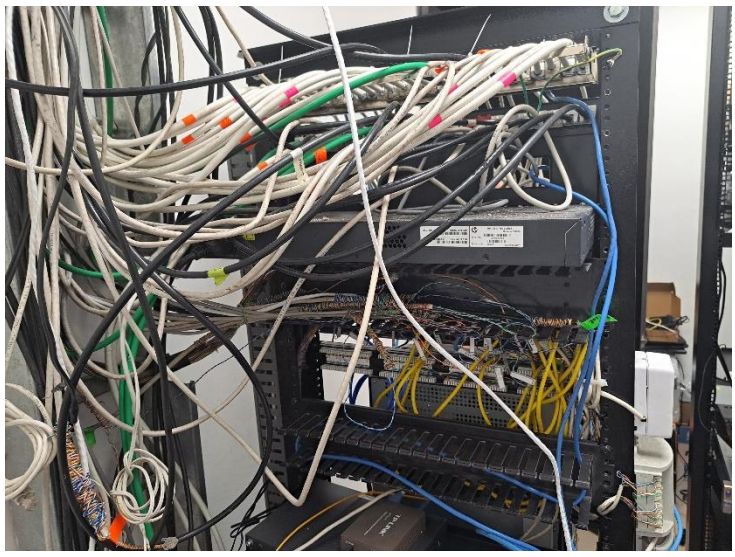


Imagen Cableado de Red

Imagen Servidores

Imagen Ups

Imagen Aire acondicionado

Detalle de la implementación del prototipo

Al mantener un firewall Fortigate 50E el cuál es 100% administrado por la empresa Telconet y todas las reglas de filtrado y más servicios como el acceso desde el exterior para clientes remotos es a través de una VPN y no se tiene ningún servicio publicado al internet se presenta la solución ideal para la implementación del prototipo.

Configurando el prototipo IDS después del Firewall es lo más conveniente ya que analiza el tráfico que el Firewall deja pasar a la Red, esto permite evitar falsas alarmas o falsos positivos.

Al realizar este prototipo se debe garantizar el funcionamiento en conjunto otros elementos de la red como switches, routers y firewalls para que tenga un eficiente trabajo y exactitud en las detecciones que realice.

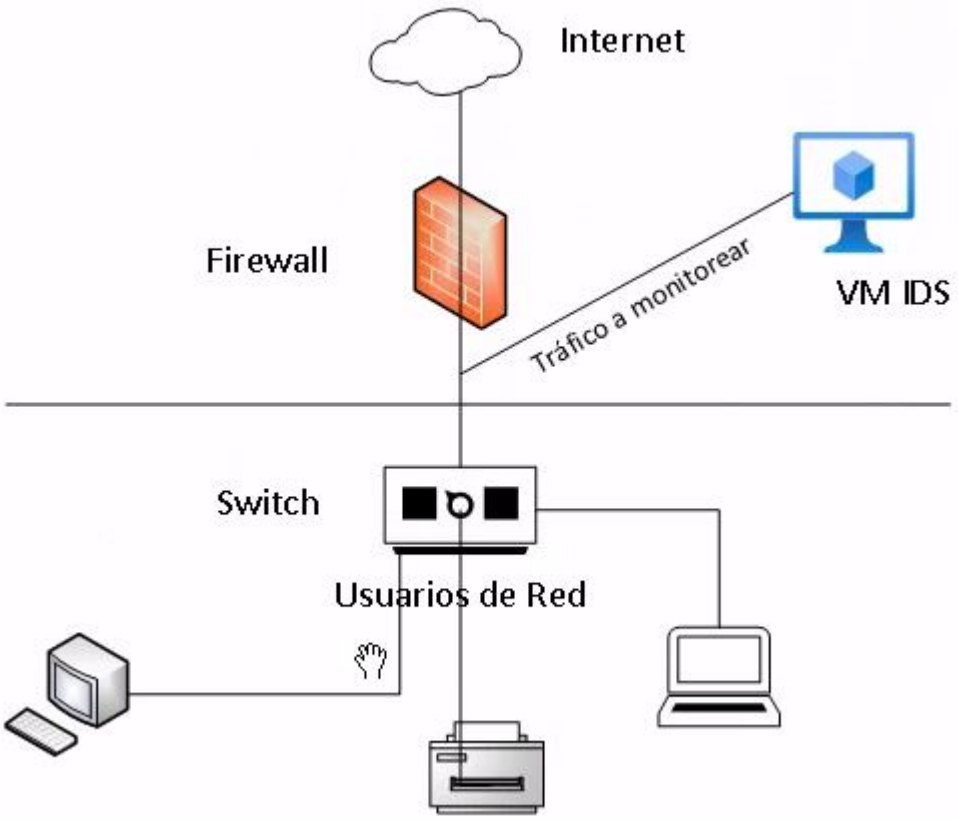


Imagen: Propuesta prototipo IDS

Al realizar este prototipo se persigue tener una alternativa con un costo bajo en comparación con herramientas de pago, sin dejar el soporte por el lado del fabricante y/o otras opciones que mantienen las versiones de pago perpetuo o licenciamiento mensual.

CAPITULO 3

Realizar pruebas de funcionamiento y rendimiento de los sistemas de detección y respuesta ante intrusiones, evaluando su efectividad en la detección y respuesta a posibles amenazas.

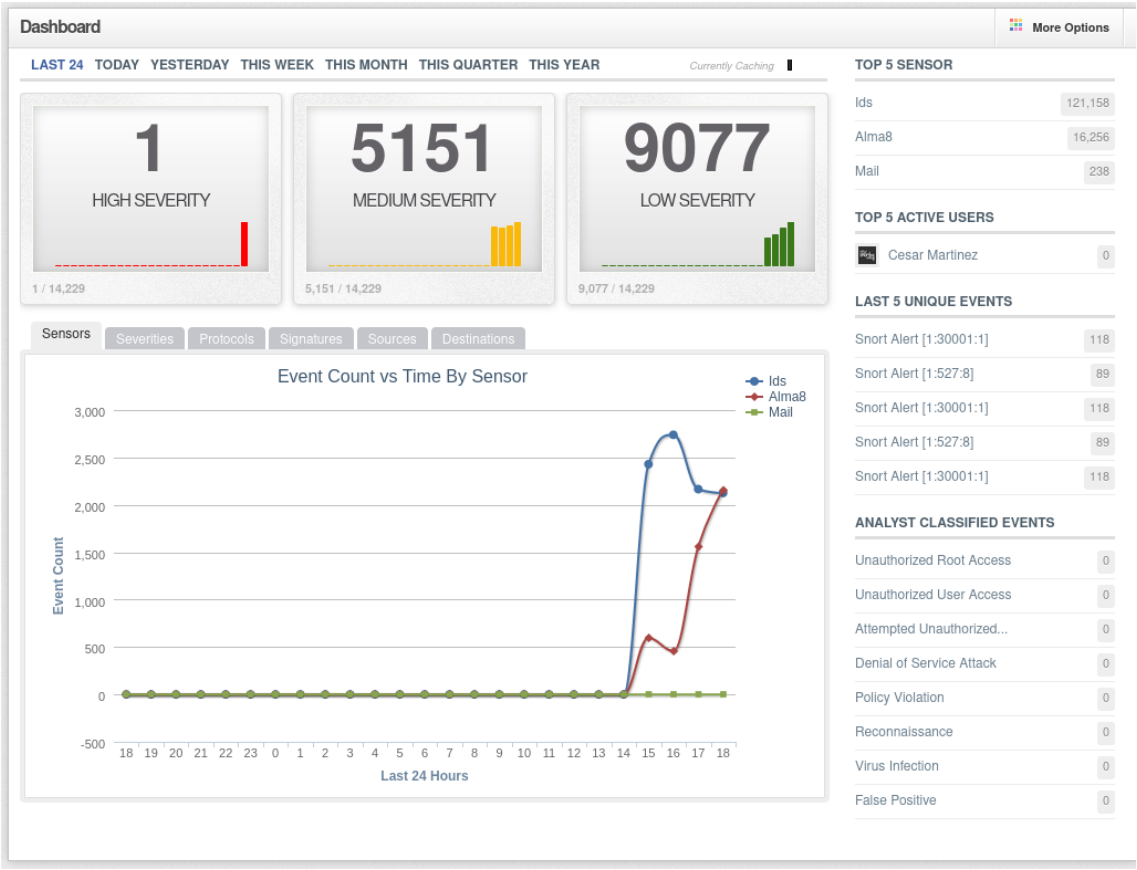
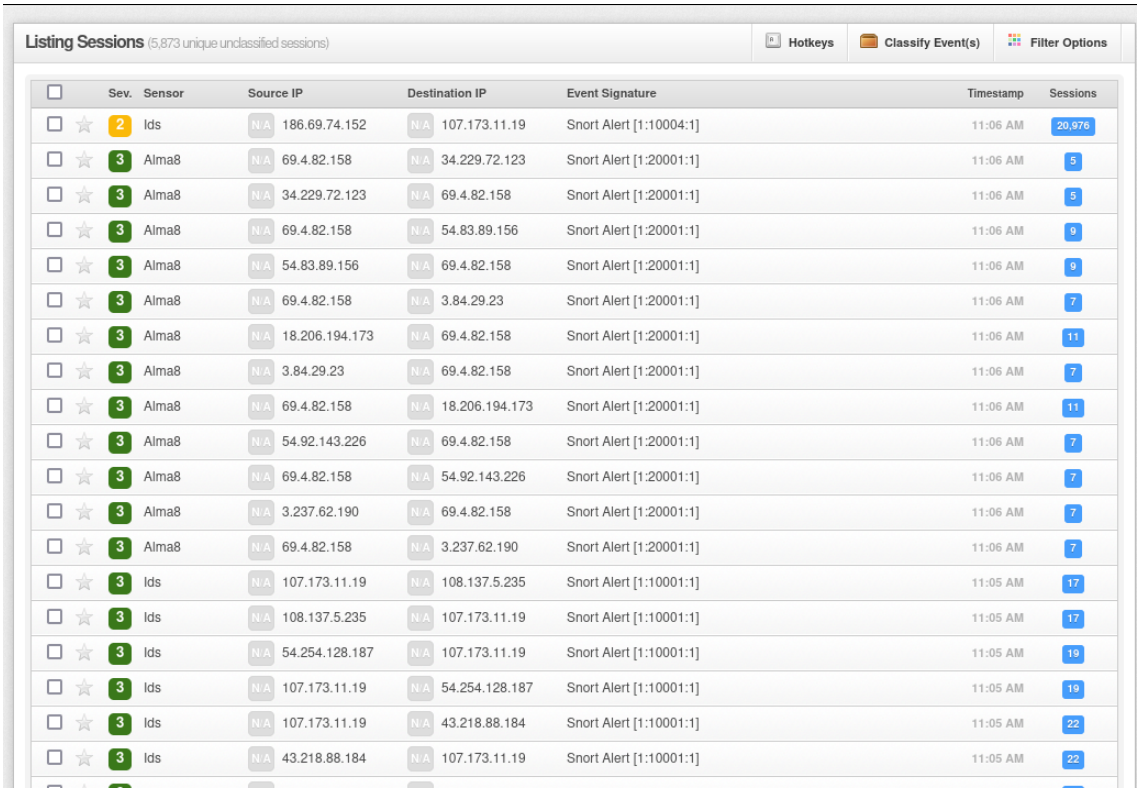


Figura Dashboard Snorby

Una vez realizada la implementación del prototipo se procede a realizar pruebas para verificar el correcto funcionamiento de la herramienta implementada.



Sev.	Sensor	Source IP	Destination IP	Event Signature	Timestamp	Sessions
2	lds	186.69.74.152	107.173.11.19	Snort Alert [1:10004:1]	11:06 AM	20,976
3	Alma8	69.4.82.158	34.229.72.123	Snort Alert [1:20001:1]	11:06 AM	5
3	Alma8	34.229.72.123	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	5
3	Alma8	69.4.82.158	54.83.89.156	Snort Alert [1:20001:1]	11:06 AM	9
3	Alma8	54.83.89.156	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	9
3	Alma8	69.4.82.158	3.84.29.23	Snort Alert [1:20001:1]	11:06 AM	7
3	Alma8	18.206.194.173	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	11
3	Alma8	3.84.29.23	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	7
3	Alma8	69.4.82.158	18.206.194.173	Snort Alert [1:20001:1]	11:06 AM	11
3	Alma8	54.92.143.226	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	7
3	Alma8	69.4.82.158	54.92.143.226	Snort Alert [1:20001:1]	11:06 AM	7
3	Alma8	3.237.62.190	69.4.82.158	Snort Alert [1:20001:1]	11:06 AM	7
3	Alma8	69.4.82.158	3.237.62.190	Snort Alert [1:20001:1]	11:06 AM	7
3	lds	107.173.11.19	108.137.5.235	Snort Alert [1:10001:1]	11:05 AM	17
3	lds	108.137.5.235	107.173.11.19	Snort Alert [1:10001:1]	11:05 AM	17
3	lds	54.254.128.187	107.173.11.19	Snort Alert [1:10001:1]	11:05 AM	19
3	lds	107.173.11.19	54.254.128.187	Snort Alert [1:10001:1]	11:05 AM	19
3	lds	107.173.11.19	43.218.88.184	Snort Alert [1:10001:1]	11:05 AM	22
3	lds	43.218.88.184	107.173.11.19	Snort Alert [1:10001:1]	11:05 AM	22

Figura Registro de eventos

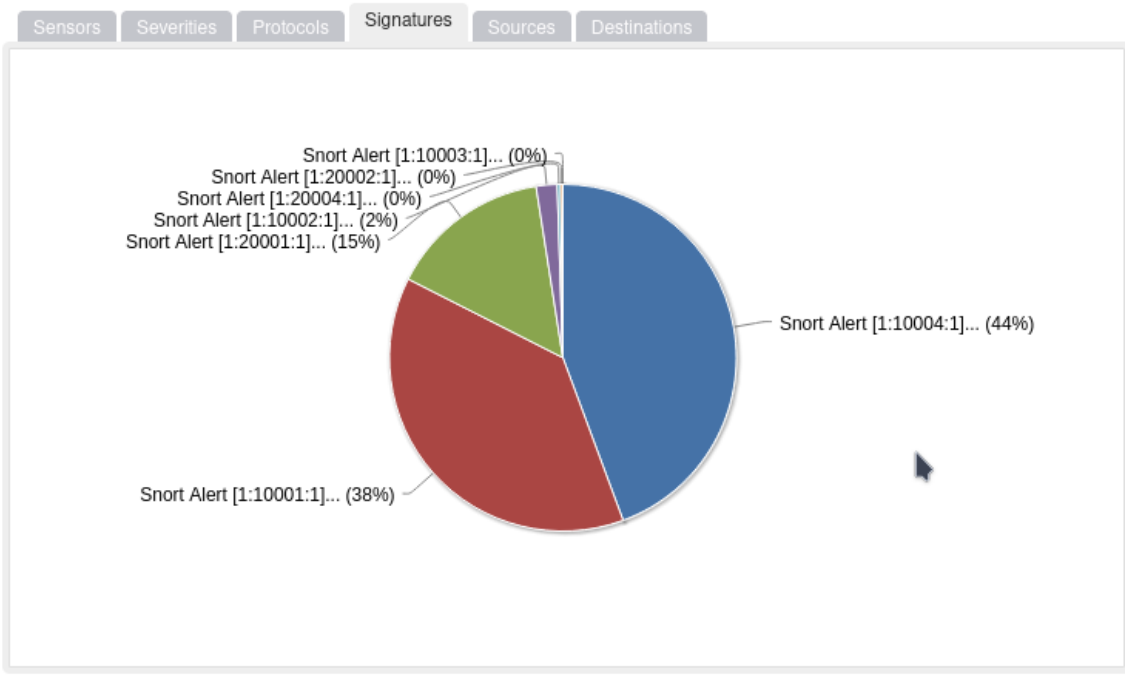
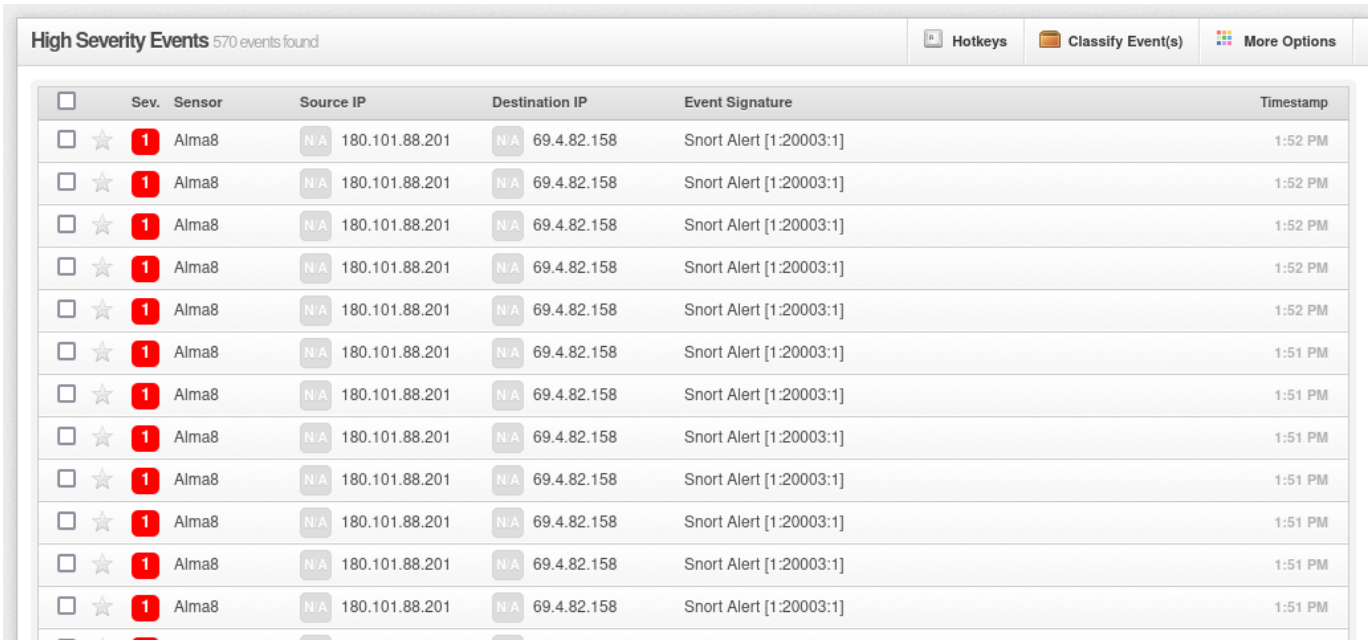


Figura Tipos de alertas



High Severity Events 570 events found

Hotkeys | Classify Event(s) | More Options

Sev.	Sensor	Source IP	Destination IP	Event Signature	Timestamp
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:52 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:52 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:52 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:52 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:52 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM
1	Alma8	180.101.88.201	69.4.82.158	Snort Alert [1:20003:1]	1:51 PM

Figura: Alerta por Snorby ssh

De root@websrv.servicomecuador.com

A cesar@total-linux.com

Asunto **lfd on websrv.servicomecuador.com: blocked 180.101.88.197 (CN/China/-)**

Responder | Reenviar | Archivar | No de

```
Time:     Fri Apr  5 13:12:42 2024 -0500
IP:       180.101.88.197 (CN/China/-)
Failures: 5 (sshd)
Interval: 3600 seconds
Blocked:  Permanent Block [LF_SSHD]

Log entries:

Apr  5 13:12:29 websrv sshd[3672]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=180.101.88.197  user=root
Apr  5 13:12:31 websrv sshd[3672]: Failed password for root from 180.101.88.197 port 29406 ssh2
Apr  5 13:12:34 websrv sshd[3672]: Failed password for root from 180.101.88.197 port 29406 ssh2
Apr  5 13:12:37 websrv sshd[3672]: Failed password for root from 180.101.88.197 port 29406 ssh2
Apr  5 13:12:41 websrv sshd[3675]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=180.101.88.197  user=root
```

Figura: Notificación bloqueo de la Ip en Firewall por fallo ssh

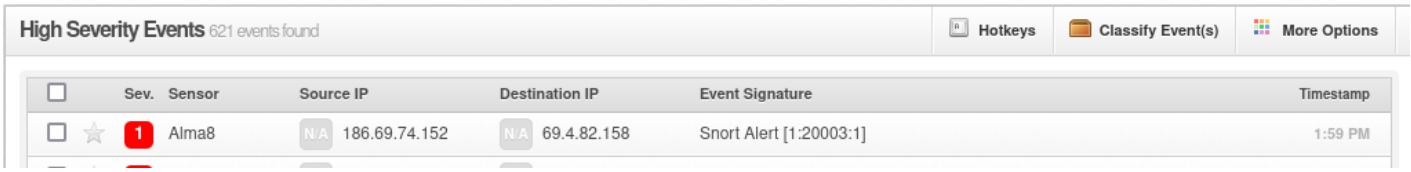


Figura: Notificación Snort regla SSH

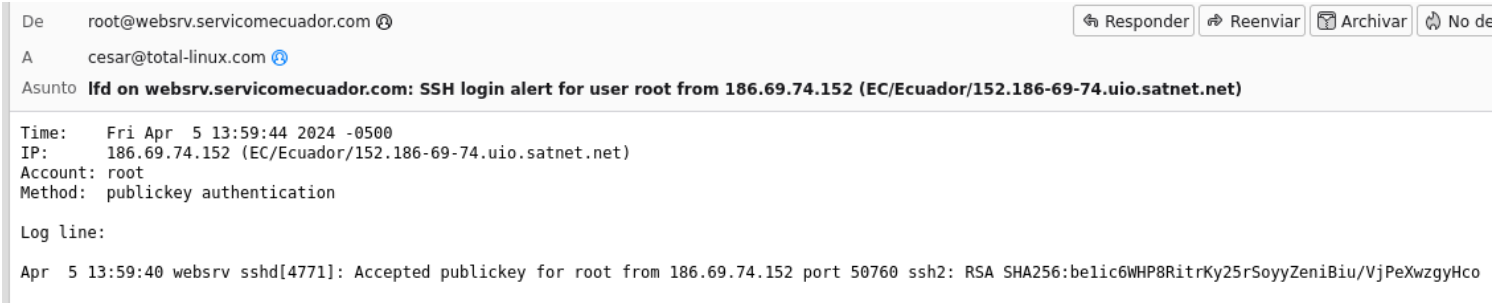


Figura: Notificación acceso ssh válido

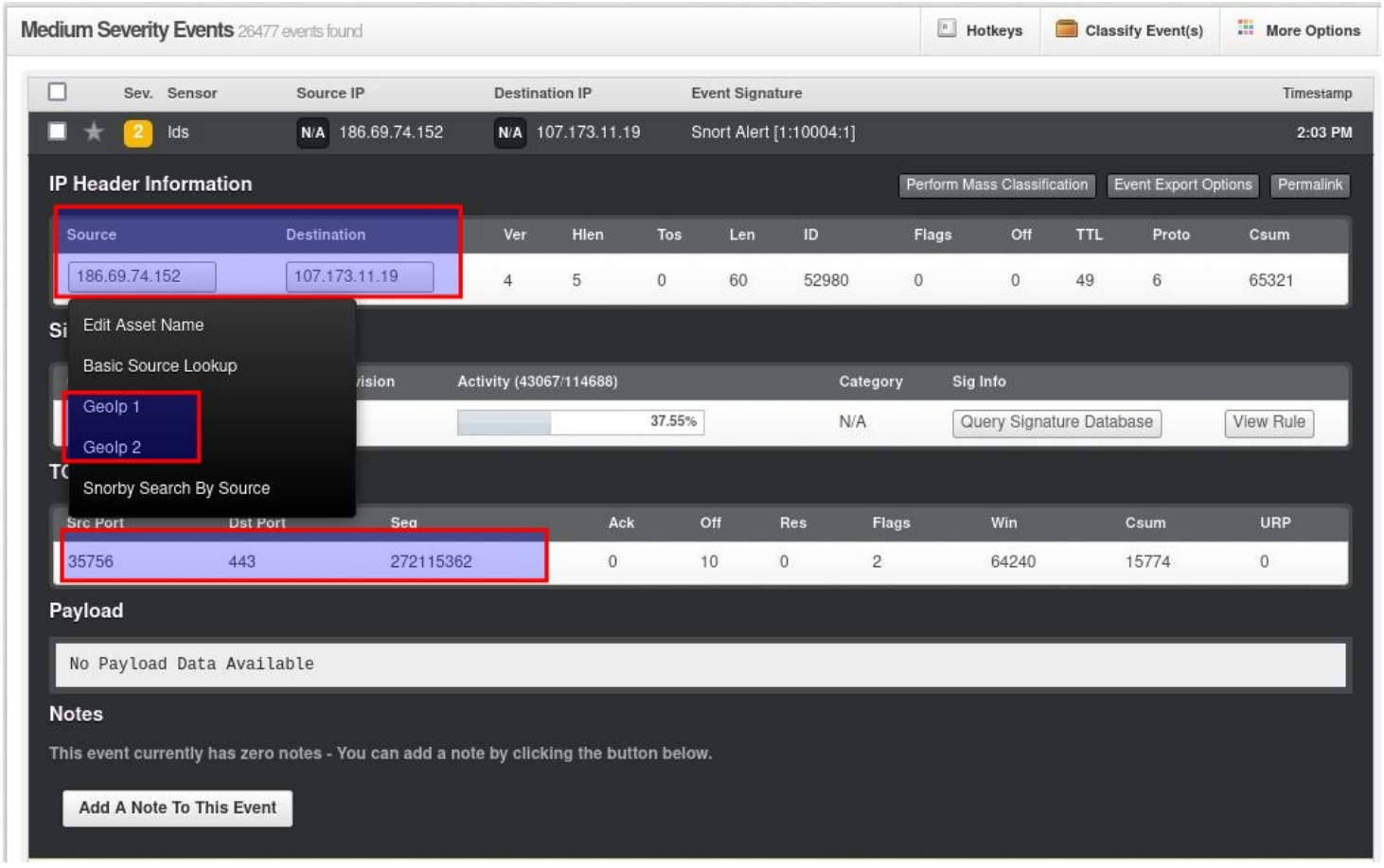


Figura: Detalle de alerta notificada

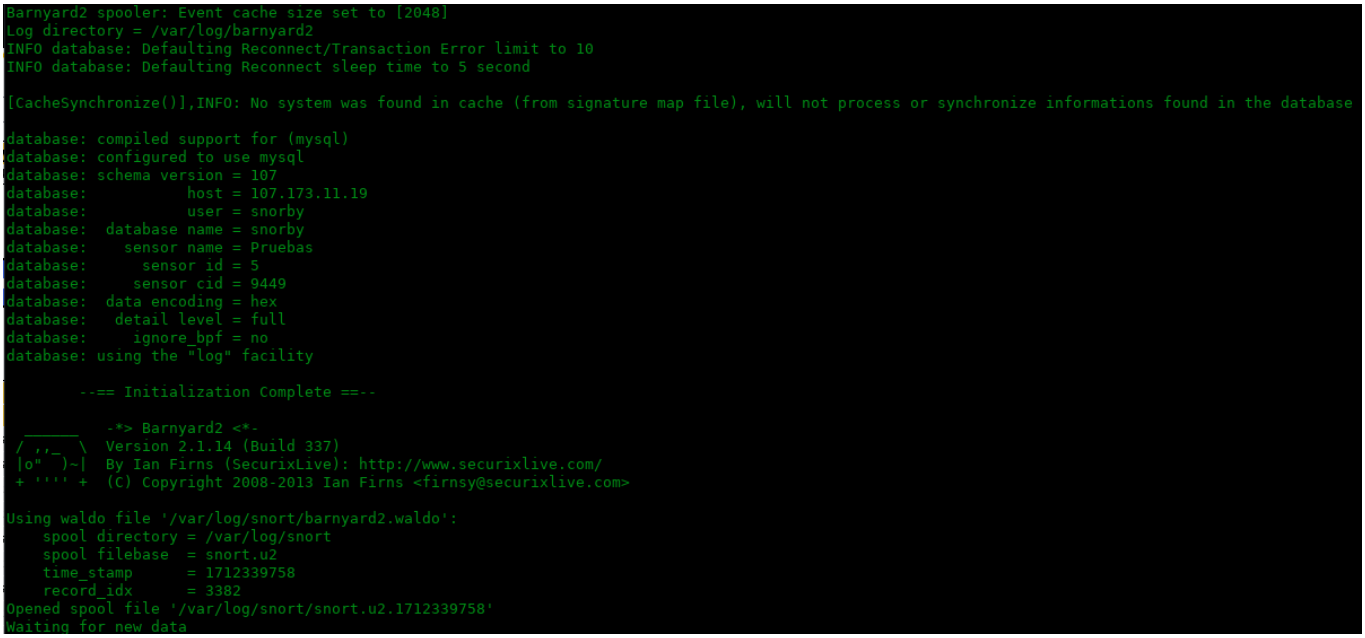


Figura Validación de funcionamiento snort cliente

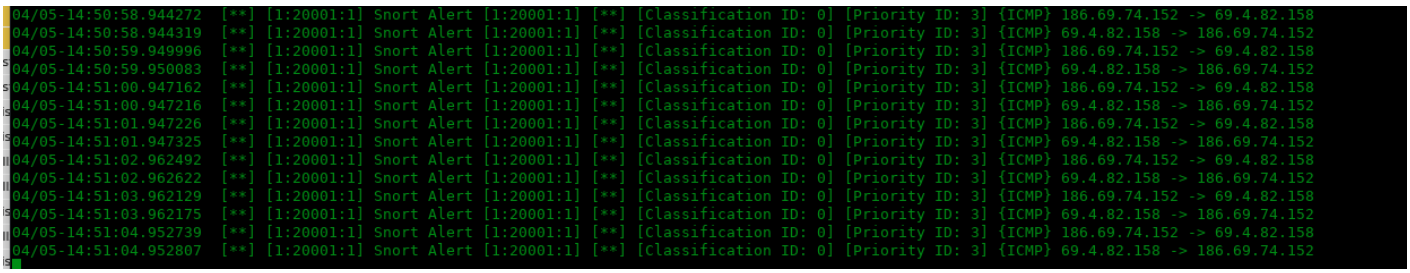


Figura: Recepción alerta de Icmp cliente

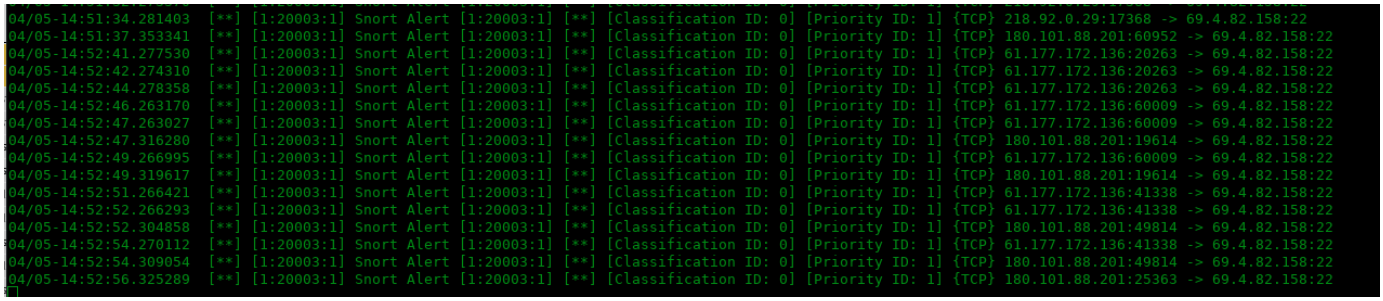


Figura: Recepción alerta de conexión ssh cliente

Cuando Snort muestra la alerta del ataque efectuado por ejemplo por ssh actúa el firewall CSF quién realiza el bloqueo de la ip atacante y notifica al administrador de este evento, de esta forma se cierra el círculo de la notificación y la acción a tomar contra el ataque.

CAPITULO 4

Elaborar un informe detallado que documente todo el proceso de integración de los sistemas de detección y respuesta ante intrusiones en la red de CONVERSA S.A, incluyendo las herramientas utilizadas, configuraciones realizadas y recomendaciones para su mantenimiento actualización e implementación futura. Preparar Planificación Diseño (metodología de diseño Top-Down) Implementación (prototipo VM) Operación Optimización

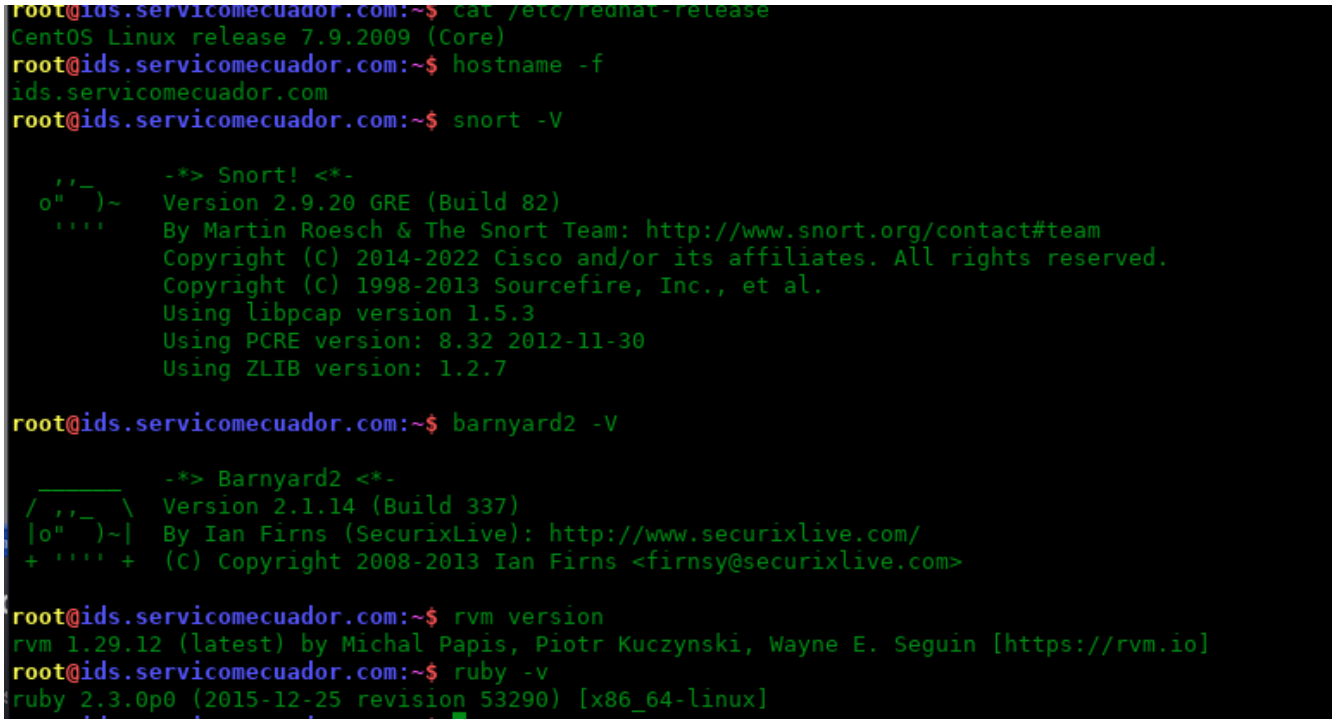


Figura: Versiones de software usado

Requerimientos de Hardware

Procesador 4 cores mínimo
Memoria Ram 4 GB mínimo
Disco duro 100GB en estado sólido mínimo
1 tarjeta de red

Proceso instalación Servidor

Una vez instalado y actualizado Centos 7.9, procedemos a deshabilitar el firewall y selinux para que no causen problemas de bloqueo a los servicios.

El orden de instalación del software para el ids es:

- Instalar todas las librerías y dependencias necesarias
- Instalar daq la versión 2.0.7
- Instalar snort la versión 2.9.20
- Instalar baryard2

- Instalar mariadb 10.3 y apache
- Instalar ruby, rails, gemas para ruby
- Instalar Snorby

Proceso:

1.- Dependencias y paquetes necesarios

```
yum install openssl-devel readline-devel libxml2-devel libxslt-devel mariadb mariadb-devel mariadb-server urw-fonts libX11-devel libXext-devel git fontconfig-devel libXrender-devel unzip wget xorg-x11-server-Xvfb libyaml libyaml-devel gdbm-devel db4-devel libffi-devel ethtool httpd  httpd-devel ImageMagick ImageMagick-devel curl libcurl libcurl-devel libmnl-devel gcc zlib-devel jansson-devel libnet-devel libnetfilter_queue-devel xorg-x11-fonts-Type1 xorg-x11-fonts-75dpi postgresql-devel java-1.8.0-openjdk
```

2.- Instalar daq la versión 2.0.7

```
- Descargarse el código fuente wget https://www.snort.org/downloads/snort/daq-2.0.7.tar.gz
- Descomprimir tar xvzf daq-2.0.7.tar.gz
- Acceder a los archivos fuentes cd daq-2.0.7
- Una vez dentro debemos preparar y compilar los archivos fuentes para la instalación
- ./configure && make && make install
```

Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

3.- Instalar snort la versión 2.9.20

```
Descargarse el código fuente wget https://www.snort.org/downloads/snort/snort-2.9.20.tar.gz
- tar xvzf snort-2.9.20.tar.gz
- cd snort-2.9.20
- Una vez dentro debemos preparar y compilar los archivos fuentes para la instalación
- ./configure -enable-sourcefire && make && make install
. Ejecutamos ldconfig este comando comprueba los nombres de cabecera y de archivo de las bibliotecas que encuentra al determinar qué versiones deben tener sus enlaces actualizados ignora los enlaces simbólicos al buscar bibliotecas.
- Crear un enalce simbólico hacia snort
ln -s /usr/local/bin/snort /usr/sbin/snort
```

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

- Creamos el usuario y el grupo corresponde para trabajar snort:

```
groupadd snort
useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
```

- Creamos las carpetas donde se guardarán las reglas:

```
mkdir -p /etc/snort/rules
mkdir /var/log/snort
mkdir /etc/snort/preproc_rules
mkdir /etc/snort/so_rules
mkdir /var/log/snort
mkdir /var/log/snort/archived_logs
mkdir /usr/local/lib/snort_dynamicrules
```

- Asignamos los permisos de ejecución a todos los directorios y archivos de trabajo:

```
chmod -R 5775 /etc/snort
chmod -R 5775 /var/log/snort
chmod -R 5775 /usr/local/lib/snort_dynamicrules
chmod -R 5775 /var/log/snort/archived_logs
chmod -R 5775 /etc/snort/so_rules
chown -R snort:snort /etc/snort
chown -R snort:snort /var/log/snort
chown -R snort:snort /usr/local/lib/snort_dynamicrules
```

-Creamos los archivos donde se agregarán las reglas

```
touch /etc/snort/rules/white_list.rules
touch /etc/snort/rules/black_list.rules
touch /etc/snort/rules/local.rules
touch /etc/snort/sid-msg.map
```

-Copiamos los archivos de configuración para Snort

```
cp etc/*.conf* /etc/snort
cp etc/*.map /etc/snort
cp etc/*.dtd /etc/snort
```

-Si todo esta correcto podemos verificar la versión de Snort instalada con el comando

```
snort --version
```

```
-Descargamos las reglas de la comunidad
wget https://www.snort.org/rules/community -O community.tar.gz
Descomprimimos
tar -xvf community.tar.gz
Copiamos hacia el directorio donde se almacenarán las reglas
cp community-rules/* /etc/snort/rules

-Podemos descargar reglas de snort que son solo para usuarios registrados, previo a esto
se debe realizar un pequeño registro en el link https://www.snort.org/users/sign_in
Una vez registrado se obtendrá un código el mismo que se usará para la descarga de las
reglas
wget https://www.snort.org/rules/snortrules-snapshot-29190.tar.gz?oinkcode=[CODIGODE
SUSCRIPCIÓN] -O snortrules-snapshot-2990.tar.gz

-Descomprimimos y movemos a la carpeta de rules las reglas descargadas
tar -xvzf snortrules-snapshot-2990.tar.gz -C /etc/snort/rules
```

-Configuramos los parámetros para que Snort pueda empezar a realizar el monitoreo
Usamos un editor en este caso usamos vim, pero se puede usar nano u otro similar.
vim /etc/snort/snort.conf

Dentro de este archivo debemos modificar un par de líneas
linea 45
ipvar HOME_NET <rango de ip's>/24 Aquí colocaremos el rango y/o ip(s)s que se desee que formen parte del monitoreo de snort usando el siguiente formato 192.168.10.0/24, 192.168.10.5

linea 104
Agregar los parámetros correspondientes para que snort pueda leer las distintas rutas donde se encontraran las reglas

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/
var BLACK_LIST_PATH /etc/snort/rules/

linea 521
Agregar esta línea para que snort pueda guardar en sus logs
output unified2: filename snort.log, limit 128

linea 547
Esta línea se encuentra comentada hay que descomentar para que pueda encontrar la ruta y aplicar las distintas rules
include $RULE_PATH/local.rules
agregar debajo para que aplique las reglas de la comunidad previamente descargadas
include $RULE_PATH/community.rules

- Validar si la configuración está bien con el siguiente comando
snort -T -c /etc/snort/snort.conf

Si no hay errores debemos ver un mensaje como este

```
Total snort Fixed Memory Cost - MaxRss:54864
Snort successfully validated the configuration!
Snort exiting
```

Figura: Verificación configuración Snort

Si existe algún error mostrará el detalle del error y deberá ser corregido hasta que se muestre como la figura anterior

- Podemos probar si Snort se encuentra trabajando, realizando el monitoreo de la siguiente forma

Editamos el archivo /etc/snort/rules/local.rules dentro agregamos una regla sencilla que permitirá ver el flujo de peticiones por ICMP
alert icmp any any -> $HOME_NET any (msg:"ICMP Conexion"; sid:10001; rev:001; priority:3;)

Grabamos y salimos del archivo
En la consola ejecutamos la siguiente instrucción
snort -A console -i [id de tarjeta de red] -u snort -g snort -c /etc/snort/snort.conf
reemplazamos [id de tarjeta de red] por el id de la tarjeta de red que vamos a usar, para ello escribimos el comando ifconfig y vemos el id de la tarjeta de red

Esto lanzará una instrucción e indicará a snort que empiece a escuchar peticiones de icmp

```
root@ids.servicomecuador.com:~$ ifconfig
ens18: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 107.173.11.19  netmask 255.255.255.240  broadcast 107.173.11.31
        ether 6a:f1:b5:64:9f:84  txqueuelen 1000  (Ethernet)
        RX packets 1378692  bytes 206498511 (196.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 596690  bytes 216024422 (206.0 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 338  bytes 248459 (242.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 338  bytes 248459 (242.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figura: Verificación id tarjeta de red

```
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.5.3
           Using PCRE version: 8.32 2012-11-30
           Using ZLIB version: 1.2.7

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.2  <Build 1>
           Preprocessor Object: appid  Version 1.1  <Build 5>
           Preprocessor Object: SF_S7COMMPLUS  Version 1.0  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
Commencing packet processing (pid=32585)
```

Figura: Verificación ejecución Snort

Iniciamos un ping desde otra terminal y podremos validar que Snort empieza a receptar la alerta en base a la regla creada que fue ICMP

```
04/06-11:56:09.728687  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 52.81.36.109 -> 107.173.11.19
04/06-11:56:09.728722  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 52.81.36.109
04/06-11:56:09.958013  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 71.136.73.75 -> 107.173.11.19
04/06-11:56:09.958071  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 71.136.73.75
04/06-11:56:55.771899  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 68.183.21.19 -> 107.173.11.19
04/06-11:57:06.051898  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 13.231.111.91 -> 107.173.11.19
04/06-11:57:06.051987  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 13.231.111.91
04/06-11:57:06.055760  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 54.199.249.189 -> 107.173.11.19
04/06-11:57:06.055780  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 54.199.249.189
04/06-11:57:06.067826  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 13.230.40.231 -> 107.173.11.19
04/06-11:57:06.067841  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 13.230.40.231
04/06-11:57:07.756030  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 140.179.97.200 -> 107.173.11.19
04/06-11:57:07.756078  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 140.179.97.200
04/06-11:57:07.801526  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 52.81.64.253 -> 107.173.11.19
04/06-11:57:07.801586  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 52.81.64.253
04/06-11:57:07.922903  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 52.80.184.225 -> 107.173.11.19
04/06-11:57:07.922948  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 107.173.11.19 -> 52.80.184.225
04/06-11:57:09.167998  [**] [1:10001:1] ICMP Conexion [**] [Priority: 3] {ICMP} 71.136.86.32 -> 107.173.11.19
```

Figura: Captura de paquetes ICMP capturados por Snort

Para detener este monitoreo presionamos control + c

Una vez configurado y validado que todo esté funcionando crearemos un servicio que permita iniciar de forma automática si el equipo se apaga o reinicia, o realizamos un cambio en la configuración de Snort

- Crear un demonio o un servicio para Snort
vim /lib/systemd/system/snort.service

Dentro colocamos el siguiente contenido

[Unit]
Description=Snort NIDS Daemon
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i [id tarjeta de red]

[Install]
WantedBy=multi-user.target

Grabamos y salimos

-Realizamos una recarga de los demonios del sistema con el comando
systemctl daemon-reload

- Luego podemos iniciar el servicio de snort con el comando
systemctl start snort

- Para activar el inicio automático de snort cuando el equipo se apague o reinicie con el comando
systemctl enable snort

- Verificamos que el servicio este activo y ejecutándose con el comando
systemctl status snort

```
root@ids.servicomecuador.com:~$ systemctl status snort
● snort.service - Snort NIDS Daemon
   Loaded: loaded (/usr/lib/systemd/system/snort.service; enabled; vendor preset: disabled)
   Active: active (running) since sáb 2024-04-06 09:50:32 -05; 2h 13min ago
 Main PID: 19220 (snort)
   CGroup: /system.slice/snort.service
           └─19220 /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i ens18

abr 06 09:50:32 ids.servicomecuador.com systemd[1]: Stopped Snort NIDS Daemon.
abr 06 09:50:32 ids.servicomecuador.com systemd[1]: Started Snort NIDS Daemon.
```

Figura: Verificación estado del servicio Snort

Reglas

Podemos definir reglas propias o las que existan sin límite, las reglas dependerán mucho del tipo de servicio que se desee alertar por ejemplo la regla de icmp que hemos usado
alert icmp any any -> $HOME_NET any (msg:"ICMP Conexion"; sid:10001; rev:001; priority:3;)

Esta regla define una conexión hacia la IP que está configurada en la tarjeta de red ens18, con una prioridad 2 que es alta para ser mostrada dentro de snortby como crítica.

Instalar barnyard2

Esta herramienta nos permitirá interpretar los ficheros en formato unified2 y almacenarlos en una base de datos en MariaDB o Mysql.

Previo a la instalación deberemos tener instalado y configurado mariadb para lo cual podemos seguir este tutorial para instalar y configurar
https://www.digitalocean.com/community/tutorials/how-to-install-mariadb-on-centos-8

-Descargamos el código fuente Barynard2
wget https://github.com/firnsy/barnyard2/archive/master.tar.gz -O barnyard2-Master.tar.gz

Descomprimir el archivo descargado
tar zxvf barnyard2-Master.tar.gz

Acceder al directorio donde se realizó la descompresión
cd barnyard2-master

Ejecutamos el comando autoreconf para actualizar los ficheros de configuración del código fuente
autoreconf -fvi -I ./m4

Creamos los enlaces simbólicos necesarios para acceder a la librería dnet.h, que hace falta para compilar barnyard2

ln -s /usr/include/dumbnet.h /usr/include/dnet.h

Ejecutar
ldconfig

Iniciar la compilación del código fuente indicando que vamos a utilizar como motor de la base de datos para almacenamiento mariadb/mysql
./configure --with-mysql --with-mysql-libraries=/usr/lib64/mysql

Ejecutar
make && make install

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

Una vez finalizado podemos comprobar la versión instalada con el comando
barnyard2 -V

Finalizada la instalación vamos a configurar los archivos necesarios para que se pueda integrar con snort

Copiamos el archivo de configuración de barnyard2 dentro de snort

cp etc/barnyard2.conf /etc/snort/

Creamos el archivo de logs para barnyard2
mkdir /var/log/barnyard2

Asignamos al archivo el grupo snort para que puedan interactuar
chown snort.snort /var/log/barnyard2

Creamos y asignamos el grupo de losg para barnyard2
touch /var/log/snort/barnyard2.waldo
chown snort.snort /var/log/snort/barnyard2.waldo

Crearemos la base de datos snort
Acceder como root al motor de la base de datos
mysql -u root -p
Al ejecutar este comando nos solicitará la clave ingresada cuando instalamos y configuramos el motor en este caso Mariadb

Una vez dentro
Creamos la base de datos snort y accedemos a ella
create database snort;
use snort;

Ejecutamos el script para la creación de las tablas para el almacenamiento
source /usr/src/snort_src/barnyard2-master/schemas/create_mysql

Creamos el usuario con su respectiva clave
CREATE USER 'snort'@'localhost'IDENTIFIED BY '**********';

Asignamos los permisos necesarios para que este usuario pueda realizar sus tareas dentro de la base de datos snort
grant create, insert, select, delete, update on snort.* to 'snort'@'localhost';

Aplicamos privilegios y salimos del motor
FLUSH privileges;
exit

Configurar para que las alertas puedan ser almacenadas dentro de la base de datos snort creada anteriormente para ello editar el archivo
vim /etc/snort/barnyard2.conf

Agregamos al final la línea con los parámetros de conexión alertas y le asignamos un nombre a este sensor en este caso sensor01
output database: log, mysql, user=snort password='********' dbname=snort host=localhost
sensor name=sensor01

Modificamos la configuración en snort para decirle que ahora deberá leer los logs que fueron interpretados por barnyard2

Editamos el archivo vim /etc/snort/snort.conf
Y en la línea 521 agregar
output unified2: filename snort.u2, limit 128

reiniciamos el servicio de snort
systemctl restart snort

Como la contraseña está dentro del archivo en formato visible tenemos que prevenir que otros usuarios puedan leerla, para esto aplicamos permisos al archivo para que solo pueda ser leído

chmod o-r /etc/snort/barnyard2.conf

Una vez que Snort está ejecutándose en modo NIDS, vamos a poner a funcionar a Barnyard2

Para esto vamos a la consola y ejecutamos
barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort

Explicamos las opciones:

-c : El path del archivo de configuración
-d: Donde tiene que leer las salidas de Snort
-f: El nombre del archivo que tiene que buscar
-w: La ubicación del archivo waldo (bookmark file)
-u: Usuario con el que se ejecuta
-g: Grupo con el que se ejecuta

Para validar que barnyard2 está interpretando los archivos de alerta enviados por snort debemos acceder a la base de datos y verificar en la tabla de eventos si existen registros para ello ejecutaremos mysql -u root -p -D snort -e "select count(*) from event" nuevamente solicitará la clave de root del motor y podremos ver el total de eventos almacenados

```
root@ids.servicomecuador.com:~$ mysql -u root -p -D snort -e "select count(*) from event"
Enter password:
+----------+
| count(*) |
+----------+
|     4482 |
+----------+
```

Figura: Verificación eventos en motor de base de datos

Una vez que hemos comprobado que está funcionando debemos crear un demonio a este servicio de tal forma que se pueda encender reiniciar o recargar para ello

Crear el archivo
vim /etc/systemd/system/barnyard2.service

Dentro colocar

```
[Unit]
Description=Barnyard2 service
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -
w /var/log/snort/barnyard2.waldo -g snort -u snort
ExecReload=/bin/kill -HUP $MAINPID
PIDFile=/var/run/barnyard2/barnyard2.pid
Restart=on-failure
RestartSec=30s

[Install]
WantedBy=multi-user.target
```

Grabamos y salimos

-Realizamos una recarga de los demonios del sistema con el comando
systemctl daemon-reload

- Luego podemos iniciar el servicio de barnyard2 con el comando
systemctl start barnyard2

- Para activar el inicio automático de barnyard2 cuando el equipo se apague o reinicie con el comando
systemctl enable barnyard2

- Verificamos que el servicio este activo y ejecutándose con el comando
systemctl status barnyard2

Instalar Snorby

Para poder instalar Snorby que es la plataforma de administración donde podemos monitorear todas las alertas que emite Snort y las interpreta almacena barnyard2 debemos instalar Ruby, apache y otras herramientas.

Instalar apache
yum install apache

- Instalar un manejador de paquetes para la instalación de ruby
curl -L get.rvm.io | bash -s stable

Al ejecutar este comando se generará un código que debemos copiar y pegar para configurar el repositorio al que vamos a conectarnos para instalar Ruby y sus herramientas adicionales que se requiere.

Ejemplo

```
gpg2 --keyserver hkp://keys.gnupg.net --recv-keys
409B6B1796C275462A1703113804BB82D39DC0E3

- Una vez ejecutado vamos a actualizar el repositorio con el código que se generó al
ejecutar el comando anterior
curl -L get.rvm.io | bash -s stable

- Indicamos el source u origen del repositorio en el equipo local
source /etc/profile.d/rvm.sh

- Recargamos y validamos los requerimientos
rvm reload
rvm requirements run

- Instalamos ruby mediante rvm la versión compatible y que funciona es la 2.3.0
rvm install 2.3.0

- Instalamos el compilador para ruby y sus gemas
gem install bundler

- Instalamos rails
gem install rails
```

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

- Una vez instaladas las librerías y complementos necesarios de Ruby vamos a configurar el ambiente de Snorby en apache

```
cd /var/www/html

- Descargamos y clonamos la fuente del proyecto de git
git clone http://github.com/Snorby/snorby.git

- Creamos el archivo de configuración con los parámetros de configuración para snortby
vim /var/www/html/snorby/config/snorby_config.yml

Dentro agregamos los siguientes parámetros
production:
  baseuri: ''
  domain: 'ids.servicomecuador.com'
  wkhtmltopdf: /usr/local/bin/wkhtmltopdf
  ssl: false
  mailer_sender: 'ids@servicomecuador.com'
  geoip_uri: "http://geolite.maxmind.com/download/geoip/database/GeoLiteCountry/
GeoIP.dat.gz"
  rules:""
  authentication_mode: database
  timezone_search: true
  time_zone: 'America/Guayaquil'
```

Accedemos para crear el vhost de Snorby y sus configuraciones

Grabamos y salimos

- Creamos la base de datos para snorby
mysql -u root -p
Al ejecutar este comando solicitará clave de root que se configuró al momento de instalar y configurar el motor de la base de datos

```
CREATE DATABASE snorby;
GRANT ALL PRIVILEGES ON snorby.* TO snorby@'%' IDENTIFIED BY '********';
GRANT ALL PRIVILEGES ON snorby.* TO snorby@'localhost' IDENTIFIED BY '*****';
FLUSH PRIVILEGES;
Exit
```

- Configuramos la conexion a snorby con Mariadb
vim /var/www/html/snorby/config/database.yml

```
snorby: &snorby
  adapter: mysql
  username: snorby
  password: "*********"
  host: localhost

production:
  database: snorby
  <<: *snorby
```

Accedemos al directorio donde descargamos el código fuente del proyecto
cd /var/www/html/snorby/

Instalamos y compilamos el código fuente y las herramientas que se requiere para que snorby pueda trabajar
bundle install
bundle exec rake snorby:setup RAILS_ENV=production

-Instalamos Passenger que será en webservice para la conexión entre apache y ruby
gem install passenger

- Instalamos el módulo para que funcione con apache
passenger-install-apache2-module -a

- Una vez que finaliza este proceso mostrará una información que se deberá agregar a un par de archivos dentro de apache para que pueda funcionar snorby

- Agregar el módulo passenger dentro de Apache
vim /etc/httpd/conf.modules.d/passenger.conf

```
LoadModule passenger_module
/usr/local/rvm/gems/ruby-2.3.0/gems/passenger-6.0.20/buildout/apache2/
mod_passenger.so
```

Agregamos la configuración del módulo de passenger

```
vim /etc/httpd/conf.d/passenger.conf
<IfModule mod_passenger.c>
    PassengerRoot /usr/local/rvm/gems/ruby-2.3.0/gems/passenger-6.0.20
    PassengerDefaultRuby /usr/local/rvm/gems/ruby-2.3.0/wrappers/ruby
</IfModule>

- Creamos el vhost para que pueda funcionar Snorby
vim /etc/httpd/conf.d/snorby.conf
<VirtualHost *:80>
    ServerName ids.servicomecuador.com
    DocumentRoot /var/www/html/snorby/public
    RailsEnv production
    <Directory /var/www/html/snorby/public>
     AllowOverride all
     Options -MultiViews
    </Directory>
     ServerSignature Off
     LogLevel info
     CustomLog /var/log/httpd/ids.servicomecuador.com-access.log combined
     ErrorLog /var/log/httpd/ids.servicomecuador.com-error.log
</VirtualHost>

-  Hay que el propietario de los archivos que contiene la carpeta snortby para ello ejecutar
chown apache:apache /var/www/html/snorby/ -R

- Reiniciar apache
Systemctl restart httpd
```

- Una vez que se realice este proceso abrir el navegador y escribir el nombre del vhost creado para validar que esté funcionando, si existen errores o no muestra la pantalla de login se puede validar los mimos dentro del directorio /var/log/httpd/error.log

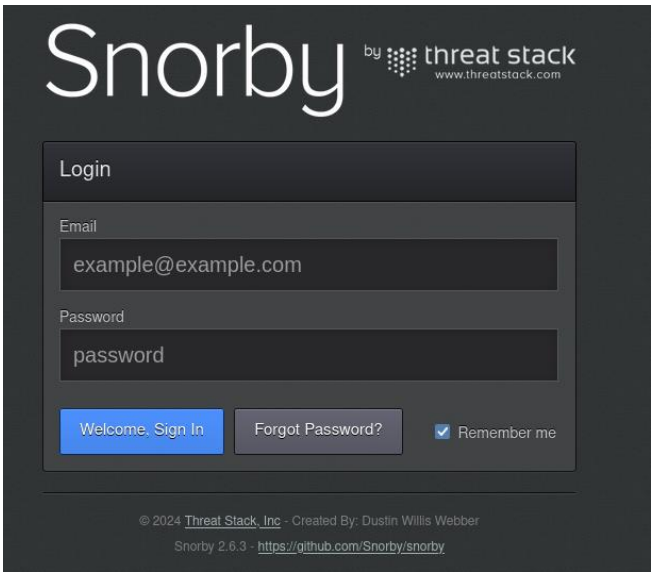


Figura: Ventana de acceso Snorby

- Los datos por defecto para el acceso por primera vez son
Email: snorby@example.com
Password: snorby

Una vez dentro se deberá crear un usuario y desactivar el que está por defecto
Por default el acceso será por un protocolo no seguro a través del puerto 80 para ello se debe instalar un certificado ssl válido usaremos un certificado emitido por Letscript para ellos seguir el siguiente procedimiento

- Instalar paquetes certbot para el certificado ssl siguiendo esta guía https://www.digitalocean.com/community/tutorials/how-to-secure-apache-with-let-s-encrypt-on-centos-7

- Cambiar la configuración del archivo /var/www/html/snorby/config/snorby_config.yml dentro de su configuración el parámetro ssl está en false, cambiarlo por true.

- Cambiar el parámetro del puerto 80 al 443 dentro del vhost en el archivo /etc/httpd/conf.d/snorby.conf

Una vez realizado estos procedimientos reiniciar apache y probar que el acceso debería ser solo por https

Dentro de la configuración de snortby hay varios parámetros que se debe configurar como iniciar la base de datos Geoip y el proceso de recolección de datos.

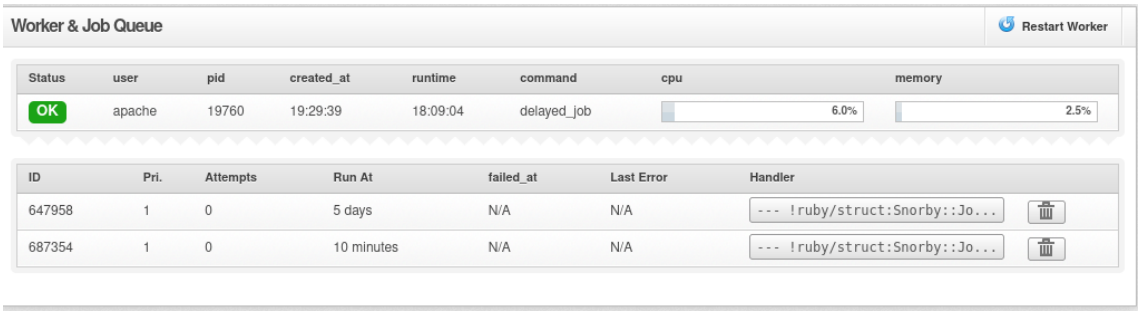


Figura: Verificación de servicios en Snorby

Figura: Verificación Whois para verificar ips de origen

Con este procedimiento hemos instalado y configurado para que snort emita alertas y monitorear servicios en el mismo servidor donde está instalado, pero una de las bondades de esta herramientas es que snort pueda recolectar y/o monitorear alertas de equipos externos o que estén dentro de la misma lan, cabe recalcar que snort es compatible con equipos Windows y Linux pero para el monitoreo externo, solo soporta equipos con sistemas operativos Linux ya que se necesita en los equipos remotos del software barnyard2 para que interprete los logs generados por snort y para sistemas con sistemas operativos Windows no existe este paquete.

Configuración equipos remotos

Para este fin se requiere que los equipos remotos Linux tengan instalado
- Snort
- Barnyard2

Los clientes Linux pueden ser cualquier distribución y se puede instalar Snort y Daq sin necesidad de compilar el código fuente (si existe el código en los repositorios oficiales de Snort), pero barnyard2 si ya que no existe un paquete listo para su instalación desde un repositorio con yum o apt para las distribuciones más populares que se usa.

Instalación de software para clientes basados en distribuciones derivadas en Rhel

1.- Instalar snort y daq desde su repositorio oficial

```
yum install https://www.snort.org/downloads/snort/snort-2.9.20-1.centos.x86_64.rpm
```

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

- Creamos el usuario y el grupo corresponde para trabajar Snort:

```
groupadd snort
useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
```

- Creamos las carpetas donde se guardarán las reglas:

```
mkdir -p /etc/snort/rules
mkdir /var/log/snort
mkdir /etc/snort/preproc_rules
mkdir /etc/snort/so_rules
mkdir /var/log/snort
mkdir /var/log/snort/archived_logs
mkdir /usr/local/lib/snort_dynamicrules
```

- Asignamos los permisos de ejecución a todos los directorios y archivos de trabajo:

```
chmod -R 5775 /etc/snort
chmod -R 5775 /var/log/snort
chmod -R 5775 /usr/local/lib/snort_dynamicrules
chmod -R 5775 /var/log/snort/archived_logs
chmod -R 5775 /etc/snort/so_rules
chown -R snort:snort /etc/snort
chown -R snort:snort /var/log/snort
chown -R snort:snort /usr/local/lib/snort_dynamicrules
```

-Creamos los archivos donde se agregarán las reglas

```
touch /etc/snort/rules/white_list.rules
touch /etc/snort/rules/black_list.rules
touch /etc/snort/rules/local.rules
touch /etc/snort/sid-msg.map
```

Copiamos los archivos de configuración para snort

cp etc/*.conf* /etc/snort
cp etc/*.map /etc/snort
cp etc/*.dtd /etc/snort

-Si todo esta correcto podemos verificar la versión de snort instalada con el comando snort --version

-Descargamos las reglas de la comunidad
wget https://www.snort.org/rules/community -O community.tar.gz
Descomprimimos
tar -xvf community.tar.gz
Copiamos hacia el directorio donde se almacenarán las reglas
cp community-rules/* /etc/snort/rules

-Podemos descargar reglas de snort que son solo para usuarios registrados, previo a esto se debe realizar un pequeño registro en el link https://www.snort.org/users/sign_in
Una vez registrado se obtendrá un código el mismo que se usará para la descarga de las reglas
wget https://www.snort.org/rules/snortrules-snapshot-29190.tar.gz?oinkcode=[CODIGODE SUSCRIPCIÓN] -O snortrules-snapshot-2990.tar.gz

-Descomprimimos y movemos a la carpeta de rules las reglas descargadas
tar -xvzf snortrules-snapshot-2990.tar.gz -C /etc/snort/rules

-Configuramos los parámetros para que snort pueda empezar a realizar el monitoreo
Usamos un editor en este caso usamos vim, pero se puede usar nano u otro similar.
vim /etc/snort/snort.conf

Dentro de este archivo debemos modificar un par de líneas

linea 104
Agregar los parámetros correspondientes para que snort pueda leer las distintas rutas donde se encontraran las reglas

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/
var BLACK_LIST_PATH /etc/snort/rules/

linea 521
Agregar esta línea para que snort pueda guardar en sus logs
output unified2: filename snort.log, limit 128

linea 547
Esta línea se encuentra comentada hay que descomentar para que pueda encontrar la ruta y aplicar las distintas rules
include $RULE_PATH/local.rules
agregar debajo para que aplique las reglas de la comunidad previamente descargadas
include $RULE_PATH/community.rules

- Validar si la configuración está bien con el siguiente comando
snort -T -c /etc/snort/snort.conf

```
Total snort Fixed Memory Cost - MaxRss:54864
Snort successfully validated the configuration!
Snort exiting
```

Figura: Verificación configuración Snort

Si no hay errores debemos ver un mensaje como este

Si existe algún error mostrará el detalle del error y deberá ser corregido hasta que se muestre como la figura anterior

- Podemos probar si Snort se encuentra trabajando, realizando el monitoreo de la siguiente forma

Editamos el archivo /etc/snort/rules/local.rules dentro agregamos una regla sencilla que permitirá ver el flujo de peticiones por ICMP
alert icmp any any -> $HOME_NET any (msg:"ICMP Conexion"; sid:10001; rev:001; priority:3;)

Grabamos y salimos del archivo
En la consola ejecutamos la siguiente instrucción
snort -A console -i [id de tarjeta de red] -u snort -g snort -c /etc/snort/snort.conf
reemplazamos [id de tarjeta de red] por el id de la tarjeta de red que vamos a usar, para ello escribimos el comando ifconfig y vemos el id de la tarjeta de red

2.- Instalar barnyard2

Esta herramienta nos permitirá interpretar los ficheros en formato unified2 y almacenarlos en una base de datos en MariaDB o Mysql.

-Descargamos el código fuente Barynard2
wget https://github.com/firnsy/barnyard2/archive/master.tar.gz -O barnyard2-Master.tar.gz

Descomprimir el archivo descargado
tar zxvf barnyard2-Master.tar.gz

Acceder al directorio donde se realizó la descompresión
cd barnyard2-master

Ejecutamos el comando autoreconf para actualizar los ficheros de configuración del código fuente
autoreconf -fvi -I ./m4

Creamos los enlaces simbólicos necesarios para acceder a la librería dnet.h, que hace falta para compilar barnyard2

ln -s /usr/include/dumbnet.h /usr/include/dnet.h

Ejecutar

ldconfig

Iniciar la compilación del código fuente indicando que vamos a utilizar como motor de la base de datos para almacenamiento mariadb/mysql
./configure --with-mysql --with-mysql-libraries=/usr/lib64/mysql

Ejecutar
make && make install

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

Una vez finalizado podemos comprobar la versión instalada con el comando
barnyard2 -V

Finalizada la instalación vamos a configurar los archivos necesarios para que se pueda integrar con snort

Copiamos el archivo de configuración de barnyard2 dentro de snort
cp etc/barnyard2.conf /etc/snort/

Creamos el archivo de losg para barnyard2
mkdir /var/log/barnyard2

Asignamos al archivo el grupo snort para que puedan interactuar
chown snort.snort /var/log/barnyard2

Creamos y asignamos el grupo de losg para barnyard2
touch /var/log/snort/barnyard2.waldo
chown snort.snort /var/log/snort/barnyard2.waldo

Configurar para que las alertas puedan ser almacenadas dentro de la base de datos snort en el servidor remoto creada anteriormente para ello editar el archivo
vim /etc/snort/barnyard2.conf

Agregamos al final la línea con los parámetros de conexión alertas y le asignamos un nombre a este sensor en este caso sensor01
output database: log, mysql, user=snorby password="CLAVE" dbname=snorby host=IP_SERVERSNORT_PRINCIPAL sensor_name=Alma8

Modificamos la configuración en snort para decirle que ahora deberá leer los logs que fueron interpretados por barnyard2

Editamos el archivo vim /etc/snort/snort.conf
Y en la línea 521 agregar
output unified2: filename snort.u2, limit 128

reiniciamos el servicio de snort
systemctl restart snort

Como la contraseña está dentro del archivo en formato visible tenemos que prevenir que otros usuarios puedan leerla, para esto aplicamos permisos al archivo para que solo pueda ser leído

```
chmod o-r /etc/snort/barnyard2.conf
```

Una vez que Snort está ejecutándose en modo NIDS, vamos a poner a funcionar a Barnyard2

Para esto vamos a la consola y ejecutamos
barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort

Explicamos las opciones:
-c : El path del archivo de configuración
-d: Donde tiene que leer las salidas de Snort
-f: El nombre del archivo que tiene que buscar
-w: La ubicación del archivo waldo (bookmark file)
-u: Usuario con el que se ejecuta
-g: Grupo con el que se ejecuta

Para validar que barnyard2 está interpretando los archivos de alerta enviados por Snort debemos acceder a la base de datos del servidor remoto y verificar en la tabla de eventos si existen registros para ello ejecutaremos mysql -u root -p -D snort -e "select count(*) from event" nuevamente solicitará la clave de root del motor y podremos ver el total de eventos almacenados

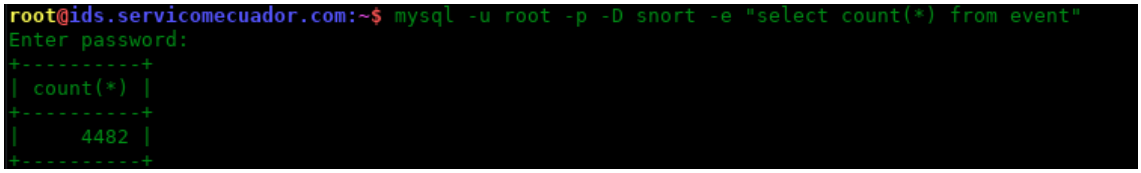


Figura: Verificación de almacenamiento de eventos en Mariadb

Una vez que hemos comprobado que está funcionando debemos crear un demonio a este servicio de tal forma que se pueda encender reiniciar o recargar para ello

Crear el archivo
vim /etc/systemd/system/barnyard2.service

Dentro colocar

```
[Unit]
Description=Barnyard2 service
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort
ExecReload=/bin/kill -HUP $MAINPID
PIDFile=/var/run/barnyard2/barnyard2.pid
Restart=on-failure
RestartSec=30s

[Install]
WantedBy=multi-user.target
```

Grabamos y salimos

-Realizamos una recarga de los demonios del sistema con el comando
systemctl daemon-reload

- Luego podemos iniciar el servicio de barnyard2 con el comando
systemctl start barnyard2

- Para activar el inicio automático de barnyard2 cuando el equipo se apague o reinicie con el comando
systemctl enable barnyard2

- Verificamos que el servicio este activo y ejecutándose con el comando
systemctl status barnyard2

Verificamos que el servicio este activo y ejecutándose con el comando
systemctl status barnyard2

Una vez agregado el cliente dentro de la interfaz web de Snorby deberá aparecer el equipo que acabamos de configurar como cliente

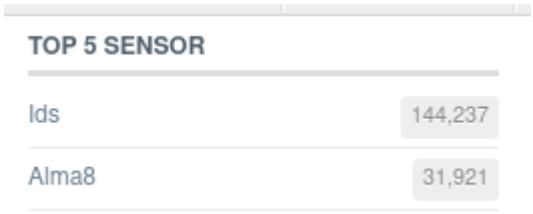


Figura: Verificación de Sensores que están siendo monitoreados

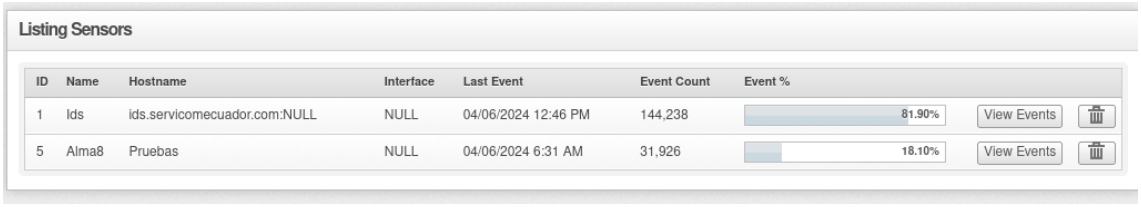


Figura: Verificación de Sensores que están siendo monitoreados

Instalación de software para clientes basados en distribuciones Debian Ubuntu

1.- Instalar Snort y Daq desde su repositorio oficial
apt install snort

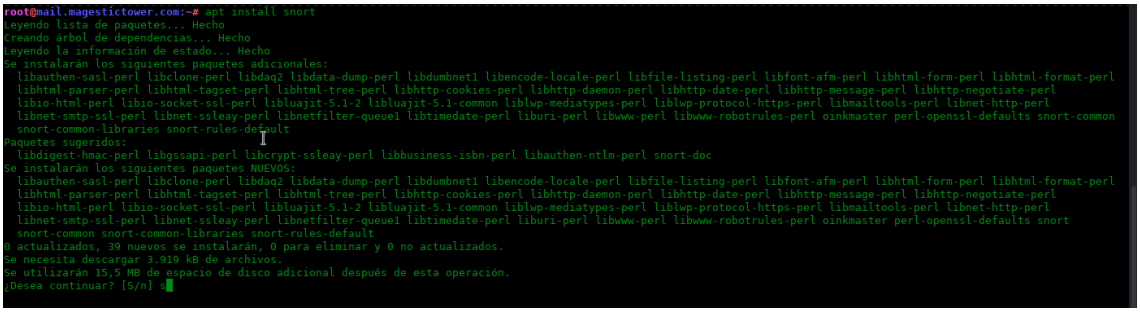


Figura: Instalación Snort Ubuntu

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

- Creamos las carpetas donde se guardarán las reglas:

```
mkdir -p /etc/snort/rules
mkdir /etc/snort/preproc_rules
mkdir /etc/snort/so_rules
mkdir /var/log/snort
mkdir /var/log/snort/archived_logs
mkdir /usr/local/lib/snort_dynamicrules
```

- Asignamos los permisos de ejecución a todos los directorios y archivos de trabajo:

```
chmod -R 5775 /etc/snort
chmod -R 5775 /var/log/snort
chmod -R 5775 /usr/local/lib/snort_dynamicrules
chmod -R 5775 /var/log/snort/archived_logs
chmod -R 5775 /etc/snort/so_rules
chown -R snort:snort /etc/snort
chown -R snort:snort /var/log/snort
chown -R snort:snort /usr/local/lib/snort_dynamicrules
```

-Creamos los archivos donde se agregarán las reglas

```
touch /etc/snort/rules/white_list.rules
touch /etc/snort/rules/black_list.rules
touch /etc/snort/rules/local.rules
touch /etc/snort/sid-msg.map
```

-Si todo esta correcto podemos verificar la versión de snort instalada con el comando snort --version

-Descargamos las reglas de la comunidad

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

Descomprimimos

```
tar -xvf community.tar.gz
```

Copiamos hacia el directorio donde se almacenarán las reglas

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

Podemos descargar reglas de Snort que son solo para usuarios registrados, previo a esto se debe realizar un pequeño registro en el link https://www.snort.org/users/sign_in
Una vez registrado se obtendrá un código el mismo que se usará para la descarga de las reglas
wget https://www.snort.org/rules/snortrules-snapshot-29190.tar.gz?oinkcode=[CODIGODE SUSCRIPCIÓN] -O snortrules-snapshot-2990.tar.gz

-Descomprimimos y movemos a la carpeta de rules las reglas descargadas

```
tar -xvzf snortrules-snapshot-2990.tar.gz -C /etc/snort/rules
```

-Configuramos los parámetros para que snort pueda empezar a realizar el monitoreo
Usamos un editor en este caso usamos vim, pero se puede usar nano u otro similar.

```
vim /etc/snort/snort.conf
```

Dentro de este archivo debemos modificar un par de líneas

linea 521
Agregar esta línea para que snort pueda guardar en sus logs
output unified2: filename snort.log, limit 128

linea 591
Agregar debajo para que aplique las reglas de la comunidad previamente descargadas
include $RULE_PATH/community.rules

- Validar si la configuración está bien con el siguiente comando
snort -T -c /etc/snort/snort.conf

Si no hay errores debemos ver un mensaje como este

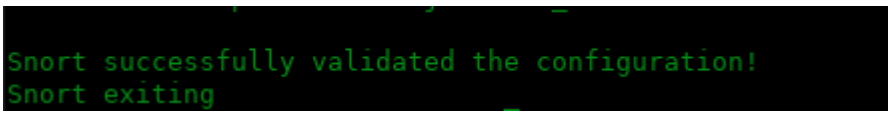


Figura: Verificación configuración Snort

Si existe algún error mostrará el detalle del error y deberá ser corregido hasta que se muestre como la figura anterior

- Podemos probar si snort se encuentra trabajando, realizando el monitoreo de la siguiente forma

Editamos el archivo /etc/snort/rules/local.rules dentro agregamos una regla sencilla que permitirá ver el flujo de peticiones por ICMP
alert icmp any any -> $HOME_NET any (msg:"ICMP Conexion"; sid:10001; rev:001; priority:3;)

Grabamos y salimos del archivo
En la consola ejecutamos la siguiente instrucción
snort -A console -i [id de tarjeta de red] -u snort -g snort -c /etc/snort/snort.conf
reemplazamos [id de tarjeta de red] por el id de la tarjeta de red que vamos a usar, para ello escribimos el comando ifconfig y vemos el id de la tarjeta de red

Instalar barnyard2

Esta herramienta nos permitirá interpretar los ficheros en formato unified2 y almacenarlos en una base de datos en MariaDB o Mysql.

Instalar paquetes adicionales que requiere para compilar barnyard2

apt-get install build-essential libtool autoconf git autoconf -y
apt-get install libpcap-dev libmysqld-dev -y
apt-get install -y autoconf libtool autogen libtool shtool libmysql++-dev libmysqlclient-dev

-Descargamos el código fuente Barynard2
wget https://github.com/firnsy/barnyard2/archive/master.tar.gz -O barnyard2-Master.tar.gz

Descomprimir el archivo descargado
tar zxvf barnyard2-Master.tar.gz

Acceder al directorio donde se realizó la descompresión

```
cd barnyard2-master
```

Ejecutamos el comando autoreconf para actualizar los ficheros de configuración del código fuente

```
autoreconf -fvi -I ./m4
```

Creamos los enlaces simbólicos necesarios para acceder a la librería dnet.h, que hace falta para compilar barnyard2

```
ln -s /usr/include/dumbnet.h /usr/include/dnet.h
```

Ejecutar

```
ldconfig
```

Iniciar la compilación del código fuente indicando que vamos a utilizar como motor de la base de datos para almacenamiento mariadb/mysql

```
./configure --with-mysql --with-mysql-libraries=/usr/lib/x86_64-linux-gnu
```

Ejecutar

```
make && make install
```

Nota: Durante este proceso puede haber errores por falta de librerías y/o paquetes los cuales deberán ser descargados y/o compilados

Una vez finalizado podemos comprobar la versión instalada con el comando

```
barnyard2 -V
```

Finalizada la instalación vamos a configurar los archivos necesarios para que se pueda integrar con Snort

Copiamos el archivo de configuración de barnyard2 dentro de Snort

```
cp etc/barnyard2.conf /etc/snort/
```

Creamos el archivo de logs para barnyard2

```
mkdir /var/log/barnyard2
```

Asignamos al archivo el grupo Snort para que puedan interactuar

```
chown snort.snort /var/log/barnyard2
```

Creamos y asignamos el grupo de logs para barnyard2

```
touch /var/log/snort/barnyard2.waldo
chown snort.snort /var/log/snort/barnyard2.waldo
```

Configurar para que las alertas puedan ser almacenadas dentro de la base de datos Snort en el servidor remoto creada anteriormente para ello editar el archivo

```
vim /etc/snort/barnyard2.conf
```

Agregamos al final la línea con los parámetros de conexión alertas y le asignamos un nombre a este sensor en este caso sensor01

output database: log, mysql, user=snorby password="CLAVE" dbname=snorby host=IP_SERVERSNORT_PRINCIPAL sensor_name=Alma8
Modificamos la configuración en Snort para decirle que ahora deberá leer los logs que fueron interpretados por barnyard2

Editamos el archivo vim /etc/snort/snort.conf
Y en la línea 551 agregar
output unified2: filename snort.u2, limit 128

reiniciamos el servicio de snort
systemctl restart snort

Como la contraseña está dentro del archivo en formato visible tenemos que prevenir que otros usuarios puedan leerla, para esto aplicamos permisos al archivo para que solo pueda ser leído

chmod o-r /etc/snort/barnyard2.conf

Una vez que Snort está ejecutándose en modo NIDS, vamos a poner a funcionar a Barnyard2

Para esto vamos a la consola y ejecutamos
barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort

Explicamos las opciones:

-c : El path del archivo de configuración
-d: Donde tiene que leer las salidas de Snort
-f: El nombre del archivo que tiene que buscar
-w: La ubicación del archivo waldo (bookmark file)
-u: Usuario con el que se ejecuta
-g: Grupo con el que se ejecuta

Para validar que barnyard2 está interpretando los archivos de alerta enviados por snort debemos acceder a la base de datos del servidor remoto y verificar en la tabla de eventos si existen registros para ello ejecutaremos mysql -u root -p -D snort -e "select count(*) from event" nuevamente solicitará la clave de root del motor y podremos ver el total de eventos almacenados

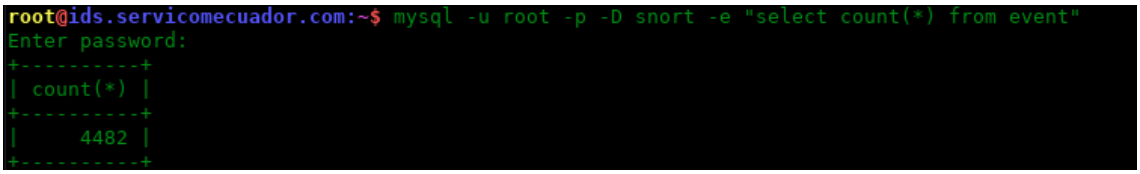


Figura: Verificación de eventos almacenados en la Base de datos

Una vez que hemos comprobado que está funcionando debemos crear un demonio a este servicio de tal forma que se pueda encender reiniciar o recargar para ello

Crear el archivo

vim /etc/systemd/system/barnyard2.service

Dentro colocar
[Unit]
Description=Barnyard2 service
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort
ExecReload=/bin/kill -HUP $MAINPID
PIDFile=/var/run/barnyard2/barnyard2.pid
Restart=on-failure
RestartSec=30s

[Install]
WantedBy=multi-user.target

Grabamos y salimos

-Realizamos una recarga de los demonios del sistema con el comando
systemctl daemon-reload

- Luego podemos iniciar el servicio de barnyard2 con el comando
systemctl start barnyard2

- Para activar el inicio automático de barnyard2 cuando el equipo se apague o reinicie con el comando
systemctl enable barnyard2

- Verificamos que el servicio este activo y ejecutándose con el comando
systemctl status barnyard2

Una vez agregado el cliente dentro de la interfaz web de Snorby deberá aparecer el equipo que acabamos de configurar como cliente

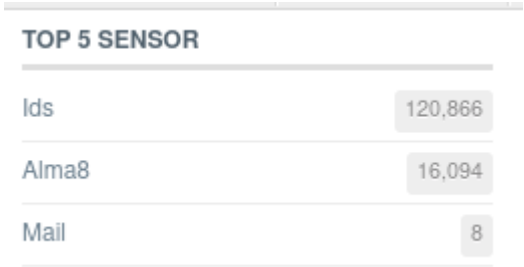


Figura: Verificación de Sensores que están siendo monitoreados

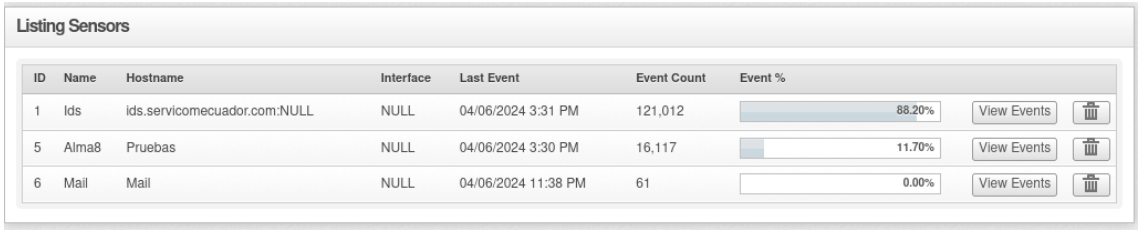


Figura: Verificación de Sensores que están siendo monitoreados

Recomendaciones para su mantenimiento actualización e implementación futura.

- Mantener actualizaciones periódicas de las reglas para mantener actualizadas las alertas que Snort controla para los servicios
- Mantener actualizados los sistemas operativos que están siendo monitoreados para agregar una capa adicional de seguridad contra ataques
- Integrar esta solución con otros sistemas como Firewalls, Ips y otros para tener una solución completa y efectiva ante cualesquier tipo de amenaza y ataque a la organización.
- Apoyo a proyectos de este tipo para que la comunidad y las empresas se sientan motivadas a incursionar en herramientas de código abierto.

CAPITULO 5

Conclusiones, recomendaciones y trabajos futuros.

Se concluye que este prototipo puede ser instalado y configurado en todo tipo de empresas, al utilizar herramientas libres pude reducir considerablemente el valor que se paga por licencias en equipos que realizan las mismas tareas de monitoreo.

Se recomienda apoyar el uso de software libre en empresas para otras implementaciones como servicio de correo, firewalls, de tal forma que se pueda mantener los sistemas con un bajo presupuesto.

En el futuro se podría realizar una sola integración de estas herramientas para que se convierta en una distribución que pueda ser usado por muchas empresas a nivel nacional e internacional.

Con el inicio de este prototipo se empezará a integrar nuevas herramientas para obtener un sistema más completo para prevenir ataques a las organizaciones y que esto perjudique al desempeño de las actividades empresariales.

Que este proyecto no se quede solo en un prototipo, sino que se materialice de forma definitiva incorporando una solución proactiva para la organización

En la empresa no solo depender de un equipo Firewall que pueda brindar todos los servicios ya que puede ocasionar que haya falsos positivos que puedan convertirse en riesgos reales para la organización.

Se concluye que el uso de herramientas de software libre ayuda y colabora en el desarrollo de las actividades de las empresas de cualquier estilo brindando seguridad y estabilidad en los servicios que se implementen bajo estas plataformas.

1. Referencias bibliográficas

[1] https://www.cisco.com/c/en/us/support/docs/availability/high-availability/15112-HAS-baseline.html

[2] https://programoweb.com/antecedentes-de-los-sistemas-de-deteccion-de-intrusos/

[3] https://www.cisco.com/c/en/us/support/docs/availability/high-availability/15112-HAS-baseline.html

[4] http://repository.unipiloto.edu.co/bitstream/handle/20.500.12277/10876/Proyecto%20IDS%20Grodco.pdf?sequence=1

[5] https://www.welivesecurity.com/la-es/2014/01/13/primeros-pasos-implementacion-ids-snort/

[6] https://repositorio.uniandes.edu.co/bitstream/handle/1992/15857/u239811.pdf?sequence=1&isAllowed=y

[7] https://dspace.uniandes.edu.ec/bitstream/123456789/1528/1/TUISIS020-2015.pdf

[8] https://abcxperts.com/que-es-un-sistema-de-deteccion-de-intrusiones-ids/#:~:text=Un%20Sistema%20de%20Detecci%C3%B3n%20de%20Intrusiones%2C%20IDS%20(Intrusion%20Detection%20System,de%20computadoras%20o%20un%20sistema.

[9] https://protecciondatos-lopd.com/empresas/sistema-deteccion-intrusiones-ids/

[10] https://es.wikipedia.org/wiki/Sistema_de_detecci%C3%B3n_de_intrusos#Referencias

[11] https://rua.ua.es/dspace/bitstream/10045/15687/1/JDARE-08-H.pdf

[12] https://geekflare.com/es/best-ids-and-ips-tools/

[13] https://ciberseguridad.com/servicios/sistema-deteccion-intrusos-ids/

[14] https://allcloud.io/blog/how-to-install-snorby/
[15] https://www.techanarchy.net/home-ids-with-snort-and-snorby/
[16] https://docs.snort.org/rules/options/general/rem
[17] https://hackertarget.com/snort-tutorial-practical-examples/
[18] https://linuxhint.com/configure-snort-ids-create-rules/

[19] https://manuelfrancoblog.wordpress.com/2017/10/23/barnyard2-unificar-alertas-de-snort-en-mysql/
[20] https://es.linux-console.net/?p=14489
[21] https://ciberseguridad.blog/reglas-snort-deteccion-de-intrusos-y-uso-no-autorizado/
[22] https://github.com/beave/sagan-rules/blob/master/openvpn.rules
[23] https://www.techanarchy.net/home-ids-with-snort-and-snorby/