ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

PROPUESTA DE UNA SOLUCIÓN DE SEGURIDAD QUE PROVEA UNA RESPUESTA ACTIVA FRENTE A ATAQUES DE DENEGACIÓN DE SERVICIO BASADA EN LA INTEGRACIÓN DE HERRAMIENTAS OPEN SOURCE SOBRE UN PROTOTIPO DE RED JERÁRQUICA

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ROBERTO FABIÁN CÁRDENAS VILLARREAL
roberttosp@hotmail.com

DIRECTOR: ING. DANNY SANTIAGO GUAMÁN LOACHAMÍN, MSC
danny.guaman@epn.edu.ec

Quito, mayo 2016

DECLARACIÓN

Yo Roberto Fabián Cárdenas Villarreal, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____

Roberto Cárdenas Villarreal

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Roberto Fabián Cárdenas Villarreal, bajo mi supervisión.

Ing. Danny Guamán, M.Sc.
DIRECTOR DEL PROYECTO

AGRADECIMIENTOS

Agradezco a Dios por darme la fuerza y tranquilidad
que me permitió poder llevar a cabo este proyecto.

A mi padre por el interés y preocupación
por cada paso que doy en mi vida.

A mi madre por tenerme la taza de
té caliente todas las mañanas.

DEDICATORIA

A mis abuelitos, Paquito y Teresita.

A la memoria de mi abuelita Chanita.

A mi hermano y
a la mujer que me muestra el camino
correcto día a día, Vane.

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El proyecto se encuentra constituido por 5 capítulos, los cuales se describen a continuación:

El Capítulo 1 presenta la fundamentación teórica de los temas implementados en el proyecto. Se inicia con una revisión general de las tecnologías de virtualización de redes y máquinas virtuales, VNX y KVM respectivamente, herramientas que son requeridas para desplegar un escenario de pruebas, contemplado por una red con topología jerárquica. De la misma manera se presenta la fundamentación teórica de los IRS, y finalmente una revisión de la naturaleza de los ataques de denegación de servicio.

El Capítulo 2 contempla una revisión del estado de la tecnología de los sistemas de respuesta activa, mediante una presentación de las soluciones tanto open source como comerciales que existen en la actualidad, y que características presentan para mitigar un ataque y capacidad de interoperabilidad con otras herramientas.

En el Capítulo 3 se diseña la solución de seguridad. Para llevar a cabo esta actividad se presentará la arquitectura a utilizar y las herramientas a configurar, con la finalidad de cumplir con los objetivos del proyecto.

En el capítulo 4 se llevan a cabo las pruebas, que sirven para validar la solución presentada y desplegada sobre la plataforma virtualizada.

El capítulo 5 se encentra constituido por las conclusiones y recomendaciones obtenidas tras desplegar los escenarios de red y validar la solución de seguridad.

PRESENTACIÓN

La Tecnología se encuentra presente en todo ámbito en el mundo actual. Con la aparición del internet, dispositivos inteligentes, cloud computing y demás tecnologías modernas, han creado una dependencia de cualquier actividad que se realice con la tecnología. Es preciso preguntarse qué tan íntegras, accesibles y confiables son las plataformas o infraestructuras que albergan la información, y cuan valiosa es la misma para una organización.

A la vez que la era tecnológica avanza de manera muy rápida, las soluciones de seguridad y protección son tareas esenciales y primordiales que no han perdido el ritmo de actualización, pero a pesar de ello, cada vez la tecnología y metodología de generación de ataques se desarrollan de mejor manera siendo más eficaces a la hora de actuar.

Toda organización que tenga información y servicios informáticos a su haber, se preocupa de brindar protección a las mismas, unas con mayor énfasis que otras. En muchos de los casos el nivel de interés que se muestra por la protección de servicios informáticos es alto pero el despliegue e implementación viene de la mano con los costos monetarios que estos representan, es decir que soluciones que brinden protección contra ataques más peligrosos como DDoS o ataques de día cero tienen costos muy elevados.

Otro de los factores que inciden en el manejo o compra de una solución de seguridad comercial es que, por más herramientas y funcionalidades que presente la solución, no cubrirá con los requerimientos específicos que el cliente demanda, de tal manera que deberá integrarse con otras herramientas para buscar una solución más completa.

Para solventar todos estos inconvenientes, se propone implementar, una solución de seguridad basada en herramientas open source

CAPÍTULO I

FUNDAMENTOS TEÓRICOS

En el presente capítulo se presenta una descripción de la base conceptual de las tecnologías empleadas en el proyecto, como lo son: topología de red jerárquica, virtualización de redes sobre Linux (VNX), Sistemas de Respuesta Activa frente a Intrusiones, como se encuetra constitiudo y cuales son sus predecesoresy los ataques de Denegación de Servicio (DoS)

1.1 TOPOLOGÍA DE RED JERÁRQUICA [5], [6]

El modelo de red jerárquico propuesto por Cisco, basa su diseño en 3 capas, y es aceptado e implementado por un gran número de organizaciones, independientemente de su enfoque de negocio. Las capas de una topología de red jerárquica son: acceso, distribución y núcleo. Como se puede observar en la Figura 1.1, dentro de la topología se debe definir sitios para: Granja de servidores, DMZ [1] (Bloque de servicios externos), firewall y router de frontera (Borde de internet), y hosts de la organización (campus de la organización). Es importante mencionar que esta topología es utilizada para redes LAN [2] y WAN [3]. La conectividad entre capas y zonas dentro de la topología [15] se consigue a través de switches y routers, seleccionados y localizados de acuerdo a los requerimientos de la red.

1.1.1 Capa de acceso [33], [34]

Esta capa brinda acceso a la red desde cualquier dispositivo de usuario final, estos pueden ser: laptops, PCs de escritorio, teléfonos, etc. Consta de dispositivos de conexión de capa dos (switches o conmutadores), haciendo referencia al modelo OSI[4].

1DMZ (Zona desmilitarizada). - Terminología que se le otorga a aquella parte de la red que se ubica entre la red interna y red externa de una organización. La DMZ permite conexiones desde la red interna y externa; no obstante, las conexiones desde la DMZ hacia la red interna no son permitidas.
2 LAN (Local Area Network). - Redes de área Local
3 WAN (Wide Area Netowrk). -Redes de Área extendida
4 OSI (Open System Interconnection).- Modelo de Referencia de Interconexión de Sistemas Abiertos que define las siguientes capas: física, enlace de datos, red, transporte, sesión, presentación y aplicación.

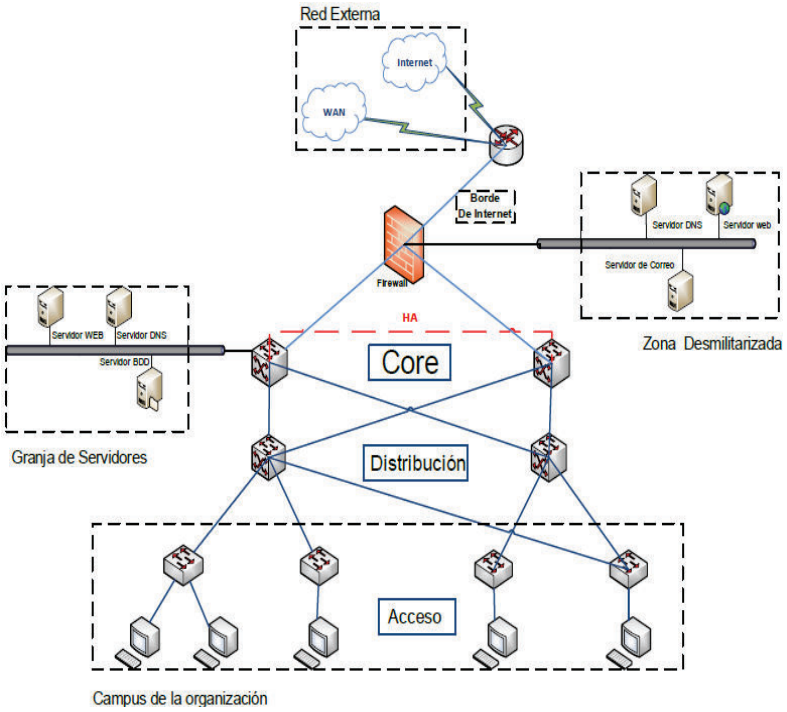


Figura 1.1 Capas de Red Jerárquica

Las funciones principales de la capa de acceso son:

- Conmutación de paquetes (Switching)
- Alta disponibilidad (HA)
- Calidad de servicio (QoS[5])
- Inspección de ARP[6]
- Controles de acceso virtual.
- Implementación de Spanning tree[7]
- Implementación de VLAN[8]

1.1.1 CAPA DE DISTRIBUCIÓN

Provee los enlaces necesarios para transportar la información que ingresa a la capa de acceso y que se dirige hacia la capa Core, basado en políticas de la organización. Además se encarga de enrutar el tráfico basado en tecnologías de

5 QoS (Quality of Service). - Protocolo que brinda calidad de servicio basado en etiquetas de prioridad de tráfico.
6 ARP (Address Resolution Protocol). - Protocolo utilizado para traducir una dirección física en una dirección lógica.
7 Spanning Tree. - Protocolo que se utiliza para evitar bucles en enlaces redundantes
8 VLAN (Virtual Local Area Network). - LAN Virtual que permite crear dominios de broadcast basados en etiquetas colocadas en tramas de capa 2.

subnetting mediante el manejo de VLANs. Los dispositivos que se encuentran en esta capa son switches de capa 3 del modelo OSI y son de mayor capacidad que los dispositivos de la capa de acceso. Las funciones principales de la capa de distribución son:

- Agregación de VLAN
- Seguridad basada en políticas ACL[9]
- Servicios de enrutamiento VLAN
- Redundancia y HA
- Balanceo de carga
- Control de dominios de broadcast

1.1.2 CAPA NÚCLEO O CORE

También se la conoce como la red backbone. Al ser la capa principal y central constará de dispositivos de capa 3 de alta velocidad, los cuales deben presentar la capacidad de conmutar paquetes a una tasa de velocidad alta. También brindará el acceso a la granja de servidores. Es la capa que proveerá de interconectividad entre los dispositivos de la capa de distribución. Las funciones principales de la capa de core son:

- Proveer alta velocidad de conmutación y enrutamiento
- Confiabilidad
- Tolerancia a fallas
- Escalabilidad
- Flexibilidad

1.1.3 BENEFICIOS DE RED JERÁRQUICA

Los beneficios que se tiene al implementar esta topología son los siguientes:

- Administración y mantenimiento: La subdivisión en capas y aislamiento de áreas propuesta por Cisco, permitirá encontrar de manera pronta cualquier incidente que suceda.

---

9 ACL(Access Control List). – Son listas de control de acceso que ofrecen un control basado en privilegios asignados.

- Seguridad: Cada dispositivo de conectividad permite crear reglas para controlar el tráfico hacia determinadas rutas o segmentos de la red.
- Redundancia: Cada dispositivo de conectividad tendrá un backup, el cual a su vez será el principal de un segundo enlace, los mismo permitirán crear enlaces redundantes.
- Escalabilidad: Permite agregar nuevos host de manera fácil y sin afectar en la operatividad de la red.
- Disponibilidad: La red podrá proveer de un alto nivel de disponibilidad en el servicio aprovechando la redundancia de los enlaces.

1.2 KVM

1.2.1 VIRTUALIZACIÓN [3], [26]

Permite a una máquina simple ser particionada o dividida en múltiples máquinas virtuales, basándose en la abstracción del hardware y presentándose para el consumo de recursos como software. Desde la perspectiva de cada sistema operativo, estas corren sobre su propio hardware, esto quiere decir que cada máquina virtual contará con sus propios recursos de procesamiento, memoria, almacenamiento, red, y otros parámetros.

La interfaz que brinda acceso a los recursos de hardware se denomina hipervisor. El hipervisor también proporciona la funcionalidad necesaria para administrar los sistemas operativos virtualizados. El host que alberga al hipervisor se conoce como Host Anfitrión. Las técnicas de virtualización han ido evolucionando y adaptándose a las necesidades que han surgido con el tiempo. La Tabla 1.1 presenta los tipos de virtualización que existen en la actualidad [3], [27].

Para el desarrollo del proyecto se utiliza KVM, que es un hipervisor Open Source con un alto rendimiento para el despliegue de máquinas virtuales. Además no requiere costos de inversión para su implementación y operación. Las características y funcionalidad del mismo se presentan a continuación.

Nombre	Funcionamiento	Ejemplo
Virtualización Completa [3], [64]	Es un tipo de virtualización que permite mantener el sistema operativo sin modificaciones. Se reservan y asignan recursos físicos a la máquina virtual a través de la creación de zonas, las cuales se aislan unas de otras. No se comparten recursos entre máquinas virtuales.	Oracle VM para arquitecturas x86 y Sparc.
Virt-in-virt	También denomina nested virt, esta tecnología permite que el Hipervisor refleje las características de CPU a un conjunto de máquinas virtuales, lo cual permitirá tener una virtualización sobre virtualización.	Utilizada para probar hipervisores dentro de una máquina virtual.
Virtualización Nativa [28], [29]	También conocida como virtualización completa asistida por hardware. Permite el acceso parcial al hardware real y parcialmente se simula el hardware con el fin de permitir que se pueda cargar un sistema operativo completo. El hipervisor solo intervendrá cuando se emita una instrucción sensible que pueda afectar al ambiente de virtualización o a los recursos físicos. El desarrollo de Intel® Virtualization Technology (VT-x) [10] y AMD Virtualization (AMD-V) [11] en procesadores modernos de 64-bits (x86_64) hacen que este tipo de virtualización sea posible.	ESXi[12] RHEV[13] KVM

Tabla 1.1 Tipos de Virtualización (1/2)

[10] **Virtualization technology.-** Característica integrada en el chip de Intel que permite crear ambientes de virtualización.
[11] **AMD Virtualization.-** Es un conjunto de funciones con chips que permiten y mejoran el rendimiento en ambientes de virtualización.
[12] **ESXi Hipervisor de VMware.-** Software que permite abstraer la infraestructúra física y presentarlas para su cosnsumo como software.
[13] **RHEV.- Red Hat Enterprise Virtualization.** Tecnología que permite crear virtualización en sistemas operativos Red Hat

Nombre	Funcionamiento	Ejemplo
Paravirtualización	Es una técnica donde, el Hipervisor provee a un sistema operativo huésped de una interfaz especial para que pueda comunicarse de manera más eficiente con él, es decir que se ejecutan versiones modificadas de los sistemas operativos. Este tipo de virtualización no requiere de extensiones o compatibilidades de virtualización con el host físico, por lo tanto, permite la virtualización en arquitecturas de hardware que no admiten virtualization nativa. Para que esta tecnología no presente inconvenientes se debe realizar un trabajo en conjunto con el kernel[14].	XEN[15] Oracle VM [16] para virtualizar sistemas operativos Windows.

Tabla 1.1 Tipos de Virtualización (2/2)

1.2.2 CARACTERÍSTICAS DE KVM [3], [25]

Es el acrónimo de Kernel-based Virtual Machine (Máquina virtual basada en kernel). Es una tecnología moderna de virtualización, integrada directamente en el kernel de Linux, esto permitirá ejecutar al kernel de Linux en bare-metal[17] y actuar directamente como su propio hipervisor.

Funciona de manera eficiente, debido a que muchas de las características que requiere un hipervisor ya están implementadas por el kernel de Linux; algunas de ellas son: administración de memoria, manejo de drivers de dispositivos físicos y gestión de energía.

KVM al ser un tipo de virtualización nativa, requiere de herramientas adicionales que permitan emular el conjunto de recursos complementarios a una máquina virtual como red, driver de sonido, etc. Dichas herramientas se describen en las siguientes secciones.

[14] **Kernel.-** Interpreta las tareas enviadas por el software para que puedan ser implementadas en el hardware.
[15] **XEN.-** Hipervisor de código abierto desarrollado por la universidad de Cambridge
[16] **Oracle VM.-** Plataforma de virtualización sobre arquitectura SPARC .
[17] **Bare-metal.-** Permite ejecutar un sistema operativo sin modificación sobre hardware estándar.

1.2.2.1 Qemu [3], [31]

Permite emular hardware a través de dispositivos modelos. Es importante mencionar que solo el hardware periférico será emulado, es decir dispositivos como: interfaces de red, tarjetas de video, controladores de disco y el BIOS. La CPU no será emulada. Una versión modificada del emulador QEMU es qemu-kvm, utilizado por KVM como dispositivo modelo.

Cada máquina virtual ejecuta su propio dispositivo modelo qemu-kvm, el cual consiste en presentar hardware propio e independiente a cada máquina. El hardware que se presenta a cada máquina es relativamente básico pero funcional, esto permite optimizar el rendimiento de cada máquina virtual sin recurrir en un excesivo uso de recursos físicos.

1.2.2.2 VitIo [32]

El funcionamiento del hardware virtual emulado no es tan rápido como el hardware real. Esto se debe a que el software que se genera de la emulación hace que se ocupe una mayor cantidad de procesamiento lo cual repercute en su rendimiento. Una manera de solventar este inconveniente es que el hipervisor provea drivers de E/S[18] para que puedan comunicarse con los dispositivos modelos. El problema ahora se encuentra en que cada sistema operativo necesita drivers independientes. La tecnología que permite solventar estos inconvenientes es VirtIO, el cual es un proyecto que permite estandarizar drivers de E/S para que de manera fácil se adapten a los diseños de diferentes hipervisores.

1.3 VIRTUAL NETWORK OVER LINUX - VNX

Se crea a partir de la necesidad de virtualizar ambientes de red, en los cuales se puedan realizar pruebas de protocolos, funcionamiento de dispositivos de red y aplicaciones. VNX permite construir redes de pruebas (testbed), en las cuales se pueda diseñar una topología de red y desplegar máquinas virtuales sobre ella, todo sobre un mismo host físico [8].

---

[18] **(E/S).-** Acrónimo utilizado para representar la Entrada y Salida desde un dispositivo electrónico o hardware de computación

1.3.1 VNUML (VIRTUAL NETWORKS USER MODE LINUX) PREDECESOR DE VNX [9]

Es una herramienta Open Source, cuyo propósito es construir escenarios de red controlados. El funcionamiento se basa en el flujo que se puede apreciar en la Figura 1.2. VNUML consta de dos componentes, el primero de ellos es XML, el cual es el lenguaje de especificación del escenario. El segundo lo conformará el intérprete del lenguaje.

Las 3 operaciones básicas que se maneja en este proyecto son:

- Construcción del escenario: Es un proceso basado en la especificación de un escenario virtual, en el cual se crean máquinas virtuales y se las interconecta dependiendo de la topología del escenario.
- Ejecución de Comandos: El usuario podrá ejecutar directamente comandos sobre las máquinas virtuales o automatizar la ejecución mediante comandos propios de VNUML.
- Liberación del escenario: Proceso para desplegar máquinas y redes virtuales.

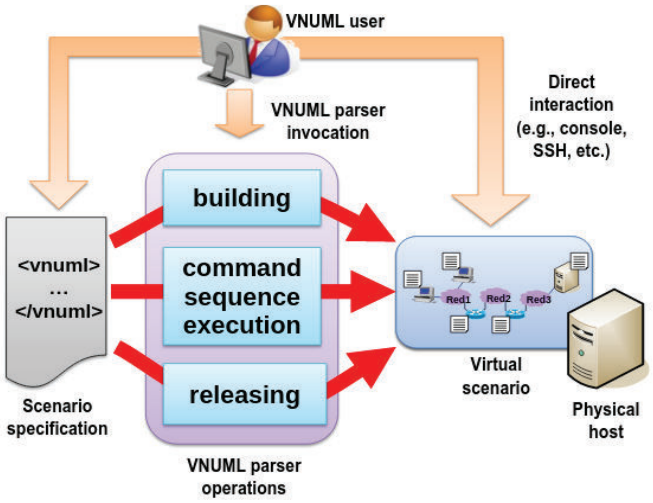


Figura 1.2 Flujo de Trabajo de VNUML [8]

VNUML fue creado en 2003 y es utilizado en algunos proyectos y laboratorios de universidades. Posteriormente, para exteriorizar los escenarios de red y que estos

puedan ser utilizados en host distintos se creó EDIV, el cual permite crear escenarios virtuales distribuidos alojados en hosts que conforman un clúster[19].

1.3.2 ARQUITECTURA VNX [2], [8], [9]

Es una sobre escritura de VNUML que presenta, entre otras, las siguientes mejoras: compatibilidad con sistemas operativos Windows, administración individual de cada máquina virtual y mejor gestión sobre un ambiente virtualizado. Basa su funcionamiento en una arquitectura modular que mediante APIs permite dar a cabida a nuevas plataformas de virtualización, como se puede ver en la Figura 1.3. Los plugins que se integran en este proyecto son:

- Dynamips: Es un software que soporta emulación de dispositivos CISCO.
- UML: Es un lenguaje mejorado de VNUML.
- Libvirt: Es un API estándar para virtualización, que provee una interfaz con la cual se puede tener acceso a varias plataformas de virtualización como XEN, KVM, VMware, etc.
- APIs Internos: Son versiones personalizadas del API libvirt, en el cual se han añadido comandos para interactuar con las máquinas virtuales directamente. A continuación se detallan los comandos añadidos.

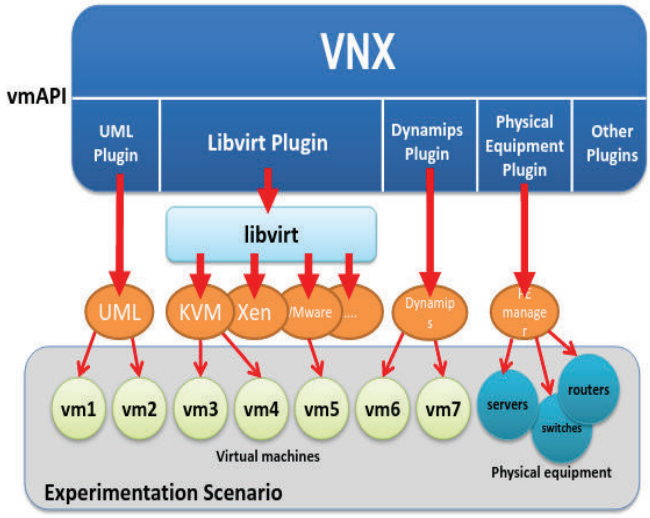


Figura 1.3 APIs de integración de VNX con varias tecnologías [8]

19 Clúster. - Unión de dos o más computadores a través de una red de alta velocidad, con el objetivo de brindar alta disponibilidad a un servicio.

Comando	Descripción
defineVM	Define una nueva máquina virtual
undefineVM	Retira la definición a una máquina virtual existente
startVM	Inicia una máquina virtual
shutdownVM	Apaga una máquina virtual
destroyVM	Elimina una máquina virtual
saveVM	Coloca a la máquina virtual en estado de hibernación
restoreVM	Restaura a una máquina virtual del estado de hibernación
suspendVM	Suspende una máquina virtual, guardando el estado de memoria
resumeVM	Restaura a una máquina virtual del estado suspendido
resetVM	Resetea una máquina virtual (destroy+define +start)
rebootVM	Reinicia una máquina virtual (shutdown+define+start)

Tabla 1.2 Comandos añadidos al API de libvirt [8]

1.3.3 LENGUAJE DE REFERENCIA [2]

Uno de los plugins desarrollados por VNX es la adopción e interpretación de lenguajes XML, el mismo que es una mejora de VNUML, el cual será el responsable de escribir la topología de red que se utilizará en los escenarios de prueba. El formato que deberá seguir el archivo UML consta de 5 secciones:

- Introducción
- Definiciones globales
- Redes Virtuales
- Máquinas Virtuales - Configuraciones del Host

1.3.3.1 Introducción

El lenguaje de referencia de VNX se encuentra basado en XML[20] y definido por el uso del esquema XSD[21] el mismo que describe la semántica de las etiquetas a utilizar. Se deberá especificar la dirección en donde se encuentre el archivo xsd, el cual también puede ser reemplazado por un URI.

---

20 XML (eXtensible Markup Language). La expresión se forma a partir del acrónimo de la expresión inglesa eXtensible Markup Language. Se trata también de un lenguaje estándar que posee una Recomendación del World Wide Web .

21 XSD (XML Schema Definition).- Describir la estructura y las restricciones de los contenidos de documentos XML

Los archivos VNX deberán iniciar con estas líneas:

- <? Xml version = "1.0" encoding = "UTF-8"?>
- <xmlns VNX: xsi = " http://www.w3.org/2001/XMLSchema- instance " xsi: noNamespaceSchemaLocation = "/ usr / share / xml / VNX / VNX-2.00.xsd">

Los comentarios deberán cumplir el siguiente formato:

- <!-- Formato de comentario, de esta manera se podrá comentar dentro del archivo de configuración sin que esta línea afecte en su funcionalidad -->

Las variables con las cuales inicia cada sección son:

- <global>, para descripción de los parámetros globales del escenario virtual.
- <net>, para descripción de las redes virtuales.
- <vm> , para descripción de las características de las máquinas virtuales.
- <host>, para descripción del host administrador del escenario virtual.

A continuación se presentan las etiquetas más utilizadas en los escenarios que se implementan en el capítulo 4.

1.3.3.2 Definiciones Globales

En esta sección serán definidas y configuradas las etiquetas que brindarán funciones a todo el escenario, estas etiquetas no representan funciones específicas, las etiquetas más utilizadas en esta sección son:

- <version>. - Especifica la versión de VNX a utilizar.
- <scenario_name>. - Especifica el nombre del escenario, este deberá ser único.
- <ssh_version>. - Establece la versión de ssh[22] a utilizar para acceder remotamente a una máquina virtual
- <ssh_key>. - Para especificar el path[23] absoluto donde se encuentra la clave pública de ssh.

---

22 SSH (Secure SHell).- Servicio utilizado por detfecto en el puerto22 que es utilizado para crear una conexión segura hacia una interfaz remota.

23 Path .- Terminología que hace referencia a la localización en el árbol de directorio que manejan los sistemas Linux.

- <automac>. - Especifica una dirección MAC, en caso de utilizar la etiqueta vacía se genera una dirección MAC automáticamente.
- <vm_mgmt>. - Sirve para configurar algunos aspectos relacionados con la interfaz de la máquina virtual. El objetivo de la interfaz es proveer un mecanismo mediante el cual se pueda interactuar con la máquina virtual. Esta etiqueta maneja los siguientes atributos:
    - privado: se crea una conexión punto a punto con una máquina virtual, utilizando una interfaz de administración con el siguiente formato: “name-e0”. donde “name” representa al nombre de la máquina virtual. La conexión es creada desde el host hacia la máquina virtual utilizando una máscara 30. Es importante destacar que la interfaz de administración deberá estar en modo bridge, por lo tanto, para utilizar este argumento se requiere contar con permisos de administrador.
    - net: la comunicación con este argumento se realizará a través de un conmutador virtual denominado uml_switched, el cual creará una interfaz en cada máquina virtual denominada eth0. El direccionamiento ip se encuentra a cargo de vnumlparser.pl (archivo que envía comandos a las máquinas virtuales ejecutando secuencias de comandos remotos).
    - ninguno: No se configuran interfaces de administración en ninguna máquina virtual. Para la comunicación con las máquinas virtuales se necesitará otro mecanismo, por ejemplo, puede ser mediante una entrada directa.
- <vm_defaults>. - Esta etiqueta asigna funciones que se ejecutarán en todas las máquinas virtuales. Siempre va acompañada de otras etiquetas con funciones más específicas para cada máquina virtual, estas pueden ser: <filesystem>, <mem>,<kernel>, <shell>, <basedir>, <mng_if>, <console>, <xterm>, <route>,<forwarding>, <user> y <filetree>. La función de esta etiqueta será anulada si en la sección de máquinas virtuales se vuelve a invocar a cualquiera de estas etiquetas.
- <dynamips_ext>. - Permite invocar a un archivo en el cual se encuentre la configuración de los dispositivos Cisco a utilizar.

- <vnx_cfg>. - Invocará a un archivo el cual contenga la configuración de la posición y tamaños de las consolas dentro de la pantalla física. Dichas consolas representan a cada una de las máquinas virtuales con las que se trabajará en los escenarios.

1.3.3.3 Redes Virtuales

La etiqueta <net> configura las redes virtuales presentes en la simulación. Cada de red virtual creada tiene la capacidad de ser un punto de interconexión entre las máquinas virtuales. Esta etiqueta no configura direcciones ip, máscaras o gateway; pero, si determina los siguientes aspectos que son fundamentales para la implementación de redes virtuales dentro del escenario.

- name: atributo para identificar a la red.
- mode: cuenta con dos valores permitidos:
    - Redes basada en virtual bridged[24]. - En este valor es admitido cualquier direccionamiento que se quiera otorgar a la red. Esta red será creada en el equipo anfitrión mediante la asignación de una tarjeta de red virtual con la capacidad de generar paquetes acordes a la red asignada, de tal manera que parecerá que cada máquina virtual está conectada mediante un cable a la tarjeta virtual.
    - Redes basadas en uml_switch .- La implementación de red virtual se realiza a través del uso de uml_switch. Un switch virtual al cual se direccionará todo el tráfico de red de las máquinas virtuales de una determinada red.
- Type: cuenta con dos valores disponibles.
    - Redes basadas en LAN. - El tipo de atributo que aceptará es LAN o default. Es la red ethernet convencional, la cual permite que todas las máquinas virtuales que se encuentren bajo el mismo dominio de broadcast tengan conexión entre sí.
    - Redes basadas en PPP. -Este valor tiene como objetivo el de establecer una comunicación ppp (point to point) entre dos

24 Bridged: Termino asignado a redes virtuales tipo puente, utilizadas para conservar su dirección IP, a pesar de que la tarjeta de red física tenga otro direccionamiento.

máquinas virtuales. El atributo bw indicará el ancho de banda que se establece para la conexión.

1.3.3.4 Máquinas Virtuales

- <vm>. - Describe una máquina virtual UML, basándose en atributos, los cuales brindarán de funcionalidades particulares a cada máquina virtual y su funcionamiento en el escenario. Los atributos más utilizados son:
    - order: solo acepta valores enteros positivos, este número establece el orden con el cual una máquina será procesada. En caso de no contar con este atributo el orden de procesamiento dependerá de la posición en la que la máquina aparece en el archivo xml.
    - type: Indica el API con el cual la máquina virtual será creada. Este valor siempre depende de la tecnología de virtualización que se utilice.
    - subtype: Tecnología de virtualización requerida para la creación de máquinas virtuales.
    - os: Sistema operativo que tendrá la máquina virtual. El valor trabaja en conjunto con type y subtype para generar los drivers necesarios que permitan un óptimo rendimiento de la máquina virtual.
    - arch: Arquitectura de la imagen del sistema operativo. Es obligatorio este valor en caso de que se trate de una arquitectura x86_64.

Dentro de esta sección existen otros parámetros configurables que describen la funcionalidad de cada máquina virtual, como las siguientes:

- <filesystem>. - Es la ruta absoluta donde se encuentra el root filesystem[25] de cada máquina virtual. Usa los siguientes argumentos:
    - type=direct, indica el archivo que se utilizará para arrancar la máquina virtual.
    - type=cow, indica el archivo que se utilizará para arrancar la máquina virtual.

25 Root Filesystem.- partición de disco físico que contiene información de arranque, configuración del sistema operativo, configuración de herramientas propias de una máquina virtual. Este archivo será invocado al iniciar una máquina virtual a través de un Hipervisor

- <mem>. - Especifica la cantidad de memoria RAM que se asignará a la máquina virtual. Los sufijos que puede tomar son k o K de Kilobytes y m o M de Megabytes.
- <if>. - Permite gestionar las interfaces de red presentes en cada máquina virtual. Presenta los siguientes atributos que le permitirán configurar y especificar la funcionalidad de cada interfaz.
    - id: identificador de número de tarjeta virtual dentro de una máquina virtual.
    - net: invocación de una red virtual a la cual pertenece la tarjeta de red previamente configurada.
- <ipv4>. - Dirección ipv4 asignada a la interfaz de red virtual.
- <ipv4 mask>. - Variación de la etiqueta ipv4 con la integración de la máscara que puede ser asignada a la interfaz de red.
- <ipv6>. - Especifica la dirección ipv6 asignada a la interfaz de red virtual
- <route>. - Especifica una ruta estática que puede ser configurada en la tabla de enrutamiento de una máquina virtual durante la construcción de la topología. Tiene dos argumentos:
    - type: indica la versión de protocolo ip a utilizar, “ipv4” o “ipv6”.
    - gw: especifica la dirección de la puerta de enlace.
- <forwarding>. - Activa el reenvío de paquetes IP para la máquina virtual. Paquetes que lleguen a una interfaz de la máquina virtual pueden ser reenviados a otra interfaz de acuerdo a la información de la tabla de enrutamiento.
- <console>. - Permite acceder a una máquina virtual por medio de una interfaz gráfica, un tty[26] o a través de un emulador de terminal denominado xterm. Los argumentos permitirán el uso de cualquiera de estas opciones.
    - id: identificador de un interfaz gráfico en cuyo caso será “0”, y “1” para identificar al modo consola.
    - display: si se requiere que se visualice o no el identificador anterior.
- <exec>. - Especifica un comando a ser ejecutado por la máquina virtual durante el uso de comandos en modo de secuencia.

[26] **TTY (Emulador de Terminal).-** Es una consola que permite el acceso al sistema operativo Linux fuera del entorno gráfico

1.3.3.5 Configuración de Host

- <hostif>. - Especifica la red virtual a la cual pertenece la tarjeta de red creada para el host.

1.4 SISTEMAS DE RESPUESTA ACTIVA

Existen herramientas de seguridad perimetral tradicionales que habitualmente son desplegadas en una organización, el firewall es un ejemplo de ellas. Dicho dispositivo se encarga de filtrar el tráfico entre redes, permitiendo o denegando el acceso en base a políticas de la organización. Sin embargo, para garantizar la seguridad integral dentro de una organización, se requieren herramientas adicionales que permitan detectar intrusiones y llevar a cabo respuestas en el menor tiempo posible [16]. A continuación se presentan algunas herramientas que pueden integrarse a soluciones tradicionales, buscando una protección integral, dinámica y eficiente.

1.4.1 IDS [17] [18]

Es un dispositivo físico o virtual que monitorea la actividad de una red y genera eventos de seguridad en caso de encontrar comportamientos anómalos. Un comportamiento anómalo será un intento de intrusión, el cual tendrá como objetivo comprometer la seguridad de la red o sistema.

A partir del primer modelo de IDS desarrollado por Dorothy Denning, han sido propuestos algunos modelos, tanto para el ámbito investigativo como comercial. Cada uno de ellos emplea distintas técnicas para recopilar y analizar los datos, a pesar de ello, la mayoría basa su diseño en la arquitectura presentada en la Figura 1.4.

- Dispositivos de recopilación de datos (sensores): Componente encargado de recopilar datos de la red o sistema al cual se está monitoreando.
- Detector (Motor de análisis detector de intrusiones): Procesa los datos recopilados por los sensores para identificar comportamientos anómalos.
- Base de conocimiento: Almacena en formato legible los datos recopilados y procesados. Esta información es utilizada por otras herramientas.

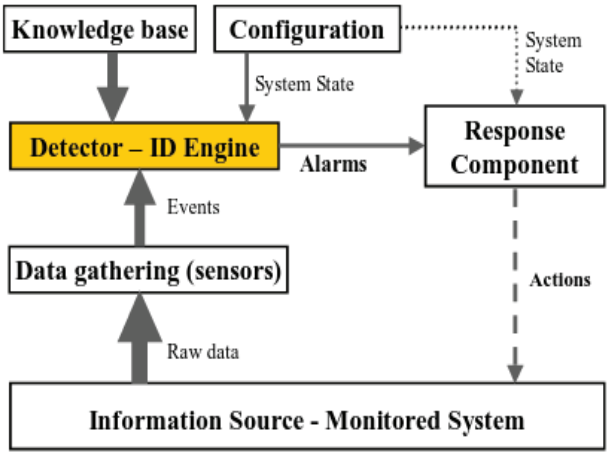


Figura 1.4 Arquitectura básica de un IDS [18]

- Dispositivo de configuración: Configura la funcionalidad del IDS. Presenta su estado actual y corrige errores en caso de poseerlos.
- Componente de respuesta: Ejecuta una acción cuando una intrusión es detectada. Las respuestas pueden ser automatizadas (activas) o requieren de la intervención del administrador (inactivas).

1.4.1.1 Clasificación de IDS

A pesar de que la mayoría de IDS cumplen con la arquitectura presentada anteriormente, cada IDS utiliza diferentes técnicas en cada uno de los componentes. En función de la técnica empleada se ha realizado una clasificación que se presenta en la Figura 1.5.

1.4.1.1.1 Basada en Sensores o fuente de información monitoreada

Esta clasificación depende de la técnica que se utiliza para recopilar datos, siendo de esta manera el prmiero de ellos un IDS basado en red (NIDS), el cual detecta ataques capturando paquetes de un segmento de red y analizándolos en busca de un comportamiento anómalo.

Esta técnica generalmente se encuentra conformada por un conjunto de sensores, los cuales deberán estar localizados estratégicamente, de tal manera que puedan brindar seguridad a toda la red.

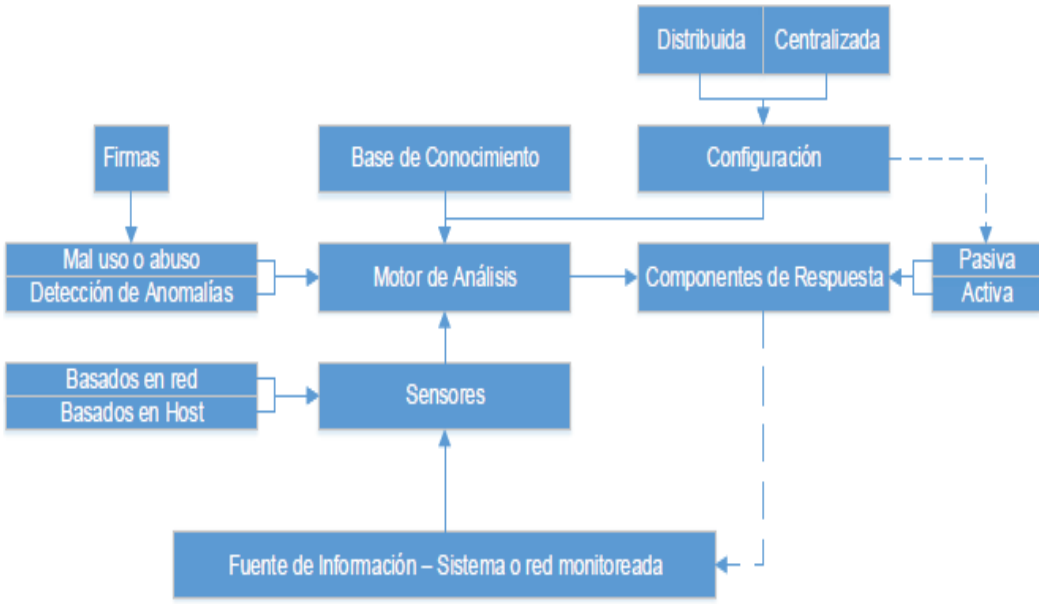


Figura 1.5 Clasificación IDS en función de componentes [18]

El segundo es un IDS basado en host (HIDS), el cual cuenta con la capacidad de monitorizar y analizar los procesos, archivos de configuración, comportamiento y servicios con los que cuenta y ofrece un determinado host o computadora. Son un complemento de los NIDS ya que podrán detectar ataques que suceden dentro de un sistema operativo y que los NIDs no hayan podido detectar, como por ejemplo, un ataque donde se manejen datos cifrados.

El despliegue de una solución basada en host es mucho más dificultosa que al hacerlo con un NIDs, ya que implica configuración en cada máquina donde se requiera la solución.

1.4.1.1.2 Basada en motor de análisis o principios de detección

Existen dos técnicas para el análisis de eventos que detectan ataques. El primero de ellos denominado *detección de mal uso o abuso*, el mismo que cuenta con una base de conocimiento que está conformado por ataques conocidos y vulnerabilidades de los sistemas. Esta información proviene de estudios realizados por personal experto en seguridad. Aplica una técnica que funciona en base a firmas. Una firma es la identificación de un evento de intrusión dentro de

una base de datos de conocimiento de ataques. El funcionamiento se basa en el monitoreo de eventos de una red, los cuales son emparejados con la base de datos de firmas de ataques para identificar una intrusión.

Las firmas pueden ser actualizadas manualmente, cuando un nuevo ataque haya sido descubierto y estudiado, pero esto vuelve vulnerable al sistema o red, dejándolo desprotegido hasta que la firma sea agregada a la base de datos. Mientras la firma sea más precisa para detectar un determinado ataque, menor será la probabilidad de generar falsos negativos.

El segundo es la detección de anomalías, creando perfiles de comportamiento de host, redes o usuarios, basando su funcionamiento en la construcción de perfiles, los cuales a su vez se crean recolectando información, datos históricos de la operación de los usuarios de un sistema operativo en tiempos determinados. Esta técnica genera una alerta al encontrar desviaciones entre un comportamiento y una medida creada por la herramienta de monitoreo.
Las medidas en la mayoría de los casos vienen dadas por umbrales compuestos por ciertos atributos de comportamiento. Los atributos pueden ser por ejemplo, la cantidad de procesamiento que utiliza una aplicación, intentos fallidos para acceder a un sistema, modificación de un archivo de configuración crítico, etc.

Las medidas también pueden ser estadísticas, donde los atributos se forman a partir de valores históricos tomados en un tiempo determinado, esto formará patrones de comportamiento, los cuales serán comparados con determinadas conductas y en caso de no coincidir se genera una alerta, permitiendo detectar ataques no conocidos.

1.4.1.1.3 Basada en la estrategia de control

Es una clasificación de acuerdo a la configuración y localización de un IDS, de esta manera se tiene un despliegue Centralizado, cuando se refiere a la localización del IDS qu se encuentra en una posición específica dentro de la topología de red de una organización. Es una técnica muy utilizada cuando se trabaja en una red pequeña.

En la actualidad, existen firewalls de nueva generación que implementan módulos de IDS en su funcionamiento, lo cual permite a estos dispositivos aumentar el nivel en el control y análisis de tráfico de la red. Este es un ejemplo de una arquitectura centralizada, ya que centra su funcionamiento en un solo dispositivo.

La otra manera de localizar un IDS es a través de un despliegue distribuido, el cual provee de una arquitectura que permitirá al IDS distribuir su funcionalidad en distintos sitios de una red. Esta técnica es utilizada en organizaciones con topología de red grande. La ventaja principal al utilizar esta técnica es la cobertura de una solución de seguridad desplegada a través de toda la red, ubicando sensores de monitoreo en lugares estratégicos y centrando la gestión de los IDS en un solo lugar.

1.4.1.1.4 Basada en el componente de respuesta

Una vez que se ha detectado una intrusión mediante cualquier análisis descrito anteriormente, el componente de respuesta del IDS permitirá, de ser el caso, efectuar alguna reacción de respuesta, la cual se clasifica de la siguiente manera:

La primera clasificación contempla una Respuesta Pasiva, de tal manera que dependerá únicamente del administrador de la red, encargado de revisar las alertas generadas por los IDS. Cada alerta tiene información detallada de lo que la originó, de esta manera el administrador se encontrará en la capacidad de efectuar una acción correctiva y reactiva ante un determinado ataque.

La segunda consiste en una Respuesta Activa, mediante la Implementación de una acción automática cuando se detecte un determinado tipo de intrusión. Las respuestas activas pueden ejecutarse de diferente manera, una de ellas consiste en aumentar el nivel de sensibilidad de los sensores de tal manera que permitan obtener una mayor cantidad de pistas sobre un posible ataque.

Otra respuesta activa, puede ejecutarse cerrando una conexión establecida con el atacante una vez que se detectó una anomalía o coincidió con una firma, a su vez, este podría ser bloqueado evitando un ataque posterior.

1.4.1.2 Localización de IDS [6] [13] [19]

La localización y configuración de cada sensor dependerá principalmente de las áreas o segmentos de red más críticos a proteger. Se busca una protección total de la red, por lo tanto, la localización de los sensores deberá cumplir con ciertos criterios, de tal manera que: trabajen de manera eficiente, el costo operativo y de inversión sean manejables, no se produzca alguna intromisión en el tráfico de red normal y no se convierta en la puerta de entrada a posibles ataques. Cada localización presenta ventajas y desventajas, las cuales serán asumidas para la configuración de cada sensor. El objetivo es que las ventajas puedan optimizar el funcionamiento de la solución y las desventajas sean manejables y no presente algún tipo de obstrucción en el funcionamiento de la red.

En los escenarios de localización que se presenta en las siguientes secciones se utiliza el término sensibilidad. La sensibilidad es el nivel de precisión con el cual se escribe una firma. Asimismo la sensibilidad está determinada por el número de firmas que se han configurado en cada sensor.

1.4.1.2.1 Localización en zona roja

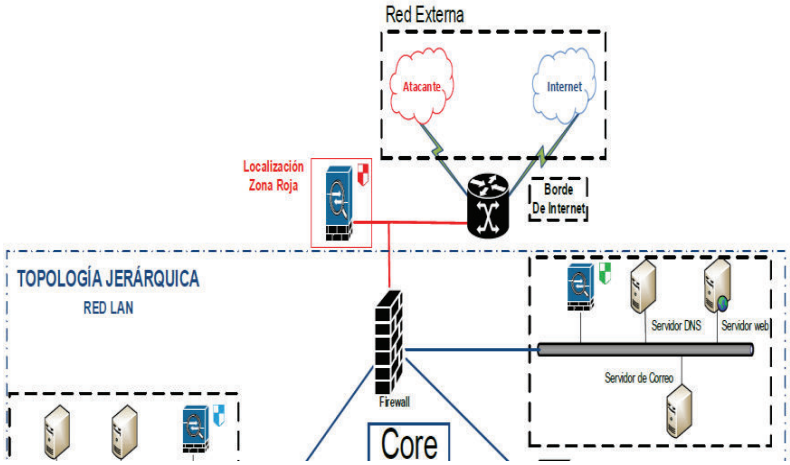


Figura 1.6 Localización zona roja

La localización de este IDS es delante del firewall, como se puede ver en la Figura 1.6, de tal manera que pueda detectar los ataques que provienen desde fuera de la red. Una de las ventajas principales es que permite detectar o reconocer señales para iniciar un ataque, tal es el caso del escaneo de puertos. Al ser una

zona de alto riesgo, el IDS detecta un gran número de alertas y pueden generar falsos negativos. En este caso la configuración del IDS debe ser poco sensible.

1.4.1.2.2 Localización en zona verde

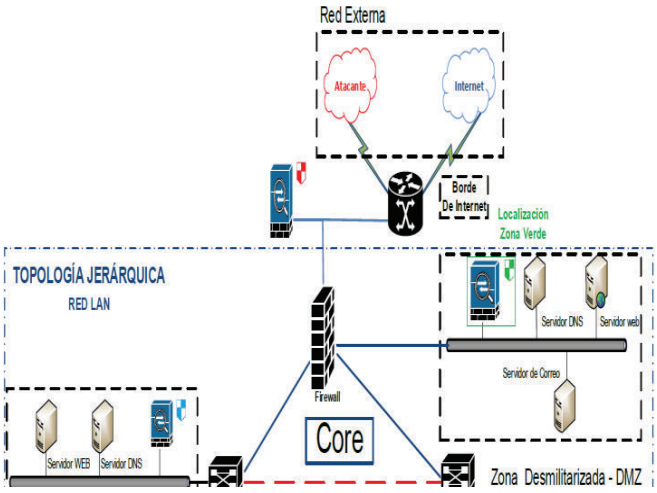


Figura 1.7 Localización zona verde

Se considera la localización del segundo sensor en la entrada de la red DMZ, como se puede apreciar en la Figura 1.7. La ventaja de esta localización es que podrá detectar ataques que provienen desde el interior y exterior de la red. Además podrá bloquearlos siempre y cuando se utilice un mecanismo de conexión entre el sensor y el firewall.

La cantidad de tráfico que se transmite en este segmento de red será menor al que se genera al exterior de la red, por lo tanto la cantidad de falsos negativos puede ser menor. La configuración del sensor podrá contar con una sensibilidad menos estricta que la configurada en la zona roja. El motor de análisis que se implementa para la detección de intrusiones en la DMZ debe estar en función de los servicios que allí se encuentren. De esta manera se disminuirá la cantidad de falsos positivos y el funcionamiento del IDS sea más eficiente.

1.4.1.2.3 Localización en zona azul

Esta localización propone un despliegue de sensores en el core de la red LAN y en la entrada de la granja de servidores, como se puede apreciar en la Figura 1.8.

De esta manera se puede monitorear el tráfico de la red interna y se analiza el tráfico que ingrese a la granja de servidores.

La sensibilidad con la cual se encuentre configurado este sensor será muy similar a la del sensor de la zona verde. Las políticas establecidas para el motor de análisis en esta zona dependerán de los servicios que se perciba de la granja de servidores. Se la denomina zona azul asumiendo que es una zona de confianza. Cualquier ataque generado desde el interior de la red dependerá en un gran porcentaje de las políticas de seguridad de la información definida y aplicada por la organización.

Los ataques internos no podrán ser detectados ni bloqueados por el firewall de borde; en este caso, se deberán implementar técnicas de defensa contra ataques, diferentes a las utilizadas para las zonas anteriores.

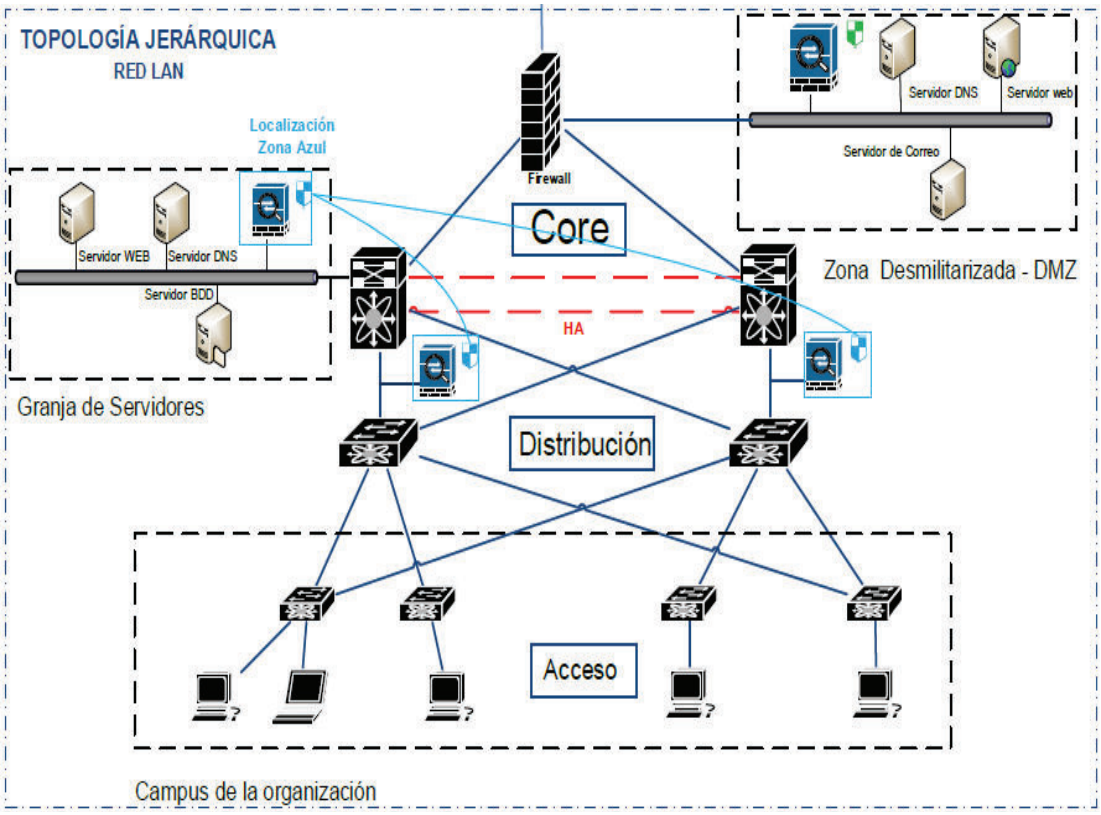


Figura 1.8 Localización zona azul

En la Figura 1.9 se presenta una topología de red jerárquica con distribución de IDS basada en criterios previos de localización.

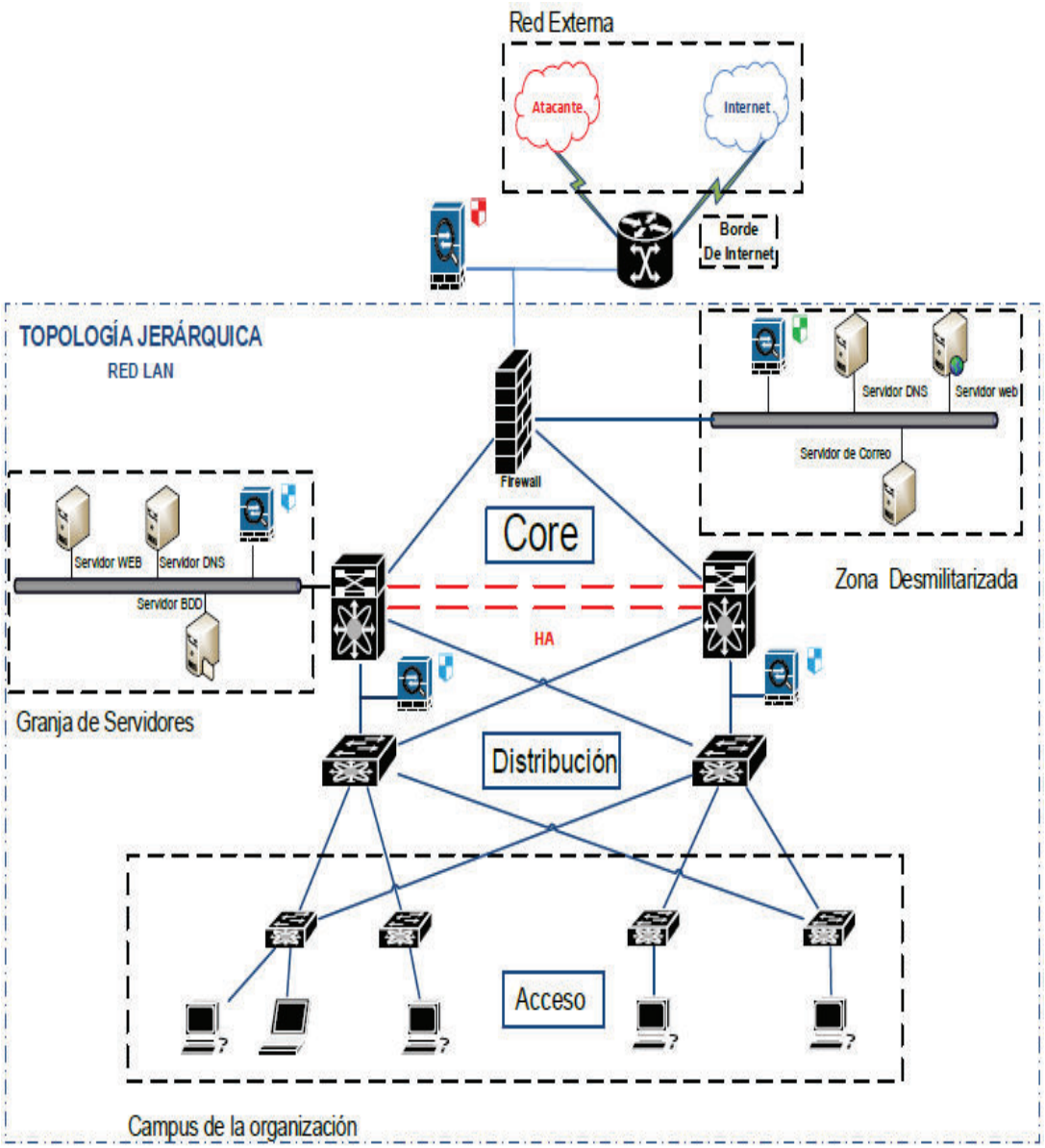


Figura 1.9 Localización Completa

Gracias a la técnica de localización distribuida de IDS en puntos estratégicos de la arquitectura utilizada, se puede monitorizar a una red con topología jerárquica completamente, independientemente del tamaño que tenga.

1.4.2 EL SALTO DESDE UNA DETECCIÓN PASIVA A CONTRAMEDIDAS ACTIVAS

Hace algunos años para ofrecer seguridad a una organización bastaba con el despliegue de reglas en un firewall o listas de control de acceso en un router, pero

debido al surgimiento de ataques más desarrollados, las soluciones tradicionales deben ser complementadas con herramientas que permitan un nivel de protección más avanzado. Los IDS son componentes que pueden ser utilizados para ayudar en una solución de defensa. Como se mencionó anteriormente, los IDSs son dispositivos que permiten monitorear la red en busca de comportamientos maliciosos y generar alertas en caso de detectar accesos no autorizados a una red o sistema informático.

La evolución del IDS es un IPS, *Intrusion Prevention Systems.* Los IPSs son sistemas de prevención de intrusiones, los cuales además de analizar la red en busca de comportamientos anómalos y generación de alertas, cuentan también con la capacidad de prevenir las intrusiones detectadas. La prevención puede ser llevada a cabo a través de un bloqueo o ejecución de acciones de respuesta básicas como: bloqueo de tráfico de la IP atacante, terminación de conexión, eliminar contenidos maliciosos de un host, reconfigurar controles de seguridad, etc. Una de las características principales de un IPS, es que debe estar ubicado en línea con el flujo de tráfico que se está monitoreando.

Una variación a los IPSs son los Sistemas de Respuesta a Intrusos o Sistemas de Respuesta Activa (IRSs). Un IRS no se encuentra necesariamente en línea con el flujo de tráfico monitoreado, sino que está en capacidad de interactuar con otros dispositivos de seguridad para disminuir el impacto de un ataque. El IRS está constituido de tres componentes.
El primero consiste en alertas que genera un IDS, el segundo tiene que ver con la selección de la respuesta y el tercero lo conforma la ejecución de la respuesta.

1.5 ATAQUES DE DENEGACIÓN DE SERVICIO [42], [43]

Se trata de un ataque informático el cual actúa con la intención de agotar los recursos disponibles para una red, una aplicación o un servicio; obstruyendo el acceso a usuarios legítimos.
A continuación se presentan de manera breve los tipos de ataques DoS que existen en la actualidad.

1.5.1 TIPOS DE ATAQUES DOS [42]

Los ataques DoS han evolucionado a través del tiempo, cada uno de ellos se generan a partir de la explotación de vulnerabilidades que se puedan encontrar en una red o sistema. Los ataques modernos llegan incluso a ser una combinación de varios tipos de ataques, volviéndose más fuertes a la hora de actuar.

1.5.1.1 Ataques DoS Distribuidos

También conocidos como DDoS, este tipo de ataque es una variante de los ataques DoS. En la actualidad resulta más rentable comercialmente producirlos debido a su efectividad y capacidad de denegación de Servicios Distribuida. La cual implica cientos o miles de estaciones que sincronizan una herramienta DoS y lanzan el ataque coordinando un destino en una hora específica. Las estaciones utilizadas para realizar el ataque pueden ser voluntarias o infectadas sin darse cuenta.

1.5.1.2 Volumétricos

Este tipo de ataque buscará consumir y saturar el ancho de banda de la red, generando exceso de tráfico, lo cual produce una congestión en la red. Puede ser dirigido a una la red (LAN) o una red WAN (ataques a ISP[27]).

1.5.1.3 Ataque DoS por consumo de recursos

El atacante intentará consumir los recursos asignados a un servidor, estos pueden ser: procesamiento, ancho de banda, memoria, almacenamiento. Se lo realiza a través de una generación excesiva de peticiones, las cuales intentará responder el servidor, llegando a tal punto que se sature.

1.5.1.4 DDoS Botnets

El ataque consiste en varios equipos que han sido infectados con un malware el cual generará gran cantidad de peticiones hacia un objetivo en común en un

27 ISP (Internet Service Providers).- Proveedores de Servicio de Internet

momento determinado, sin que los propietarios de los hosts infectados se den cuenta. A los hosts infectados también se los conoce como zombies. Para visualizar y entender de mejor manera este ataque se puede observar la Figura 1.10.

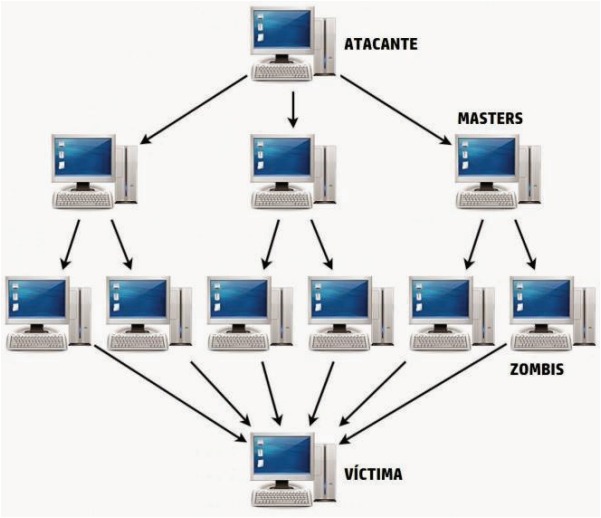


Figura 1.10 Ataque DDoS basado en Botnets OSI [45].

1.5.1.5 Reflexivo

Este tipo de ataque utiliza una técnica denominada spoofing, la cual consiste en ocultar la dirección IP del atacante en una dirección IP falsa, siendo esta la dirección IP víctima, por lo tanto si se realizan peticiones desde varias máquinas todas las respuestas tendrán como destino la IP víctima, de esta manera denegarán su servicio inundando su capacidad de transmisión.

1.5.1.6 Amplificación

Consiste en amplificar el tamaño de las tramas generadas por algún otro tipo de ataque, lo cual permitirá que a partir de pequeñas peticiones se puedan generar flujos de tráfico de mayor tamaño hacia la víctima.

1.5.1.7 Modificación de Configuración

El objetivo es modificar o borrar la configuración de un dispositivo o host crítico dentro de la organización. Una vez que se acceda ilegítimamente con privilegios de administrador a los archivos de configuración de un servidor, estos pueden ser

modificados paralizando la continuidad del negocio y produciendo una denegación de servicio.

1.5.1.8 Ataques a nivel de infraestructura

Esta clasificación abarca todos aquellos ataques que han enfocado su diseño basándose en las vulnerabilidades de los protocolos pertenecientes a las capas de red y transporte del modelo OSI (capa 3 y 4 respectivamente).
Por lo general estos son los protocolos que con mayor frecuencia se han utilizado para realizar ataques DoS, debido a su baja complejidad de construcción y ejecución.

1.5.1.9 Ataques a capa de Aplicación

Aquí se podrá encontrar aquellos ataques que han basado su diseño identificando y aprovechando los puntos débiles de la capa de aplicación, con la finalidad de producir una degradación o interrupción del servicio. Dentro de esta clasificación el protocolo mayormente explotado es HTTP, sin embargo protocolos como NTP SMTP o DNS son también ampliamente utilizados, tal como se observa en la Figura 1.11.

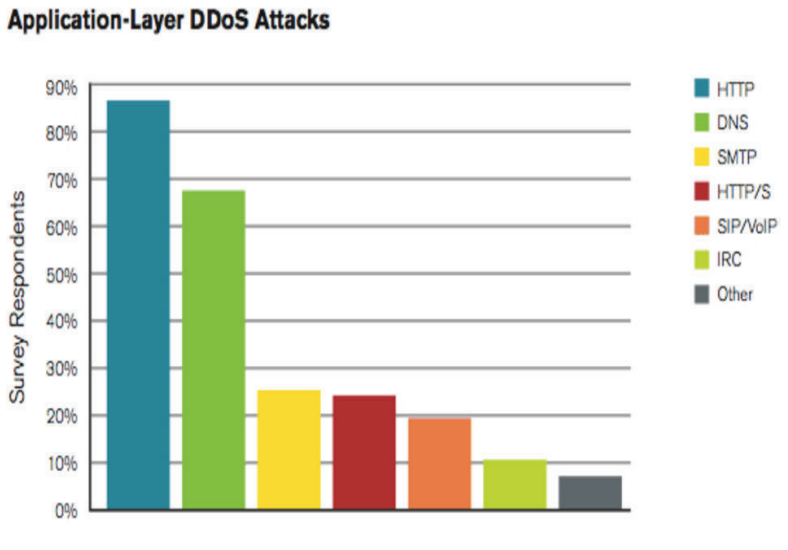


Figura 1.11 Protocolos más utilizados para ataques DoS a capa de aplicación [44].

1.5.2 ACTUALIDAD DE LOS ATAQUES DOS [42], [44], [45]

A continuación se realiza un recuento de las diferentes técnicas que se utilizan para generar ataques DoS. La ejecución de cada uno puede afectar en distintos puntos de una plataforma tecnológica, como se puede apreciar en la Tabla 1.3.

Nombre de Ataque	Basado en Protocolo	Tipo de Ataque / Tipo de Daño	Explicación
Echo request flood	ICMP	Volumétricos Saturación	También denominado Ping Flood. Envío masivo de paquetes ICMP falsos a alta velocidad, la víctima en el intento de responder todas las peticiones consume todos sus recursos de red y ancho de banda disponible hasta que se apaga. Utiliza IP Spoofed.
Packet Fragment	IP	Volumétricos Saturación	El servidor victima recibe gran cantidad de paquetes ICMP fragmentados a una tasa alta de paquetes entrantes, estos paquetes no podrán volver a ser armados,
SMURF	ICMP	Volumétricos Saturación	Ataque por saturación ICMP, envía peticiones a la dirección de broadcast ocultando su dirección de origen tras la dirección de la víctima, de tal manera que todas las respuestas estarán dirigidas hacia la víctima.
Ping of Death	ICMP	Amplificación Saturación	Envío de paquetes ICMP mayores a 65.535 bytes (normalmente son de 60 bytes) explotando la capacidad de canal asignada.
SYN Flood	TCP	Volumétricos Saturación	Envío masivo de solicitudes de conexión TCP, abrumando al servidor víctima, hasta agotar sus recursos (tabla de enlaces de memoria), donde se procesan estas peticiones.
SYN-ACK Flood	TCP	Volumétricos Saturación	Envío masivo de acuses de recibo de segmentos TCP en respuesta a peticiones SYN, que se enviarán a la IP víctima, pero fueron generados desde la IP del atacante. Es decir que se establece previamente una conexión

Tabla 1.3 Ataques DoS modernos. (1/2)

Nombre de Ataque	Basado en Protocolo	Tipo de Ataque / Tipo de Daño	Explicación
SYN ACK Reflection Flood	TCP	Reflexivo Saturación	Envío masivo de solicitudes de conexión TCP a un gran número de máquinas, cambiando la dirección de origen por la dirección de la víctima. peticiones.
Ataque de Sesión	TCP	Volumétricos	Se establece una conexión entre el atacante y la víctima, el atacante produce un atraso intencional para mantener activa la sesión. La sesión vacía agota los recursos del servidor (memoria, CPU, red) utilizados para calcular esta irregularidad.
Mal uso de Aplicación	TCP	Volumétricos Disruptivo	Afecta la disponibilidad en la capa 7 del modelo OSI. La idea aquí es hacer mal uso de los estándares de diseño o mal uso de la aplicación, de tal manera que ya sea el ataque intencional o no, se denegará el servicio a usuarios legítimos.
Flood	UDP	Volumétricos Saturación	Durante una inundación UDP, un servidor víctima recibe paquetes falsos UDP a una tasa alta. El servidor víctima consume sus recursos por el gran número de paquetes UDP entrantes.
Fragment Flood	UDP	Volumétricos Saturación	Una variación de la inundación UDP. El atacante utiliza paquetes grandes de 1500 bytes, con esto consumirá más ancho de banda con un menor número de paquetes.
DNS Flood	UDP	Volumétricos Saturación	Es este ataque el servidor que está siendo atacado no puede determinar qué paquete es verdadero, por lo tanto procede a responder a todas las peticiones.
HTTP GET/POST Flood	HTTP / TCP	Volumétricos Disruptivo	Un HTTP GET / POST es la inundación de un ataque volumétrico que no utiliza paquetes malformados, spoofing o técnicas de reflexión, es decir, utiliza una conexión legítima.

Tabla 1.3 Ataques DoS modernos. (2/2)

CAPÍTULO II

ESTADO DE LA TECNOLOGÍA

El desarrollo de este capítulo tiene la finalidad de realizar una revisión del estado de la tecnología en lo que respecta a las herramientas Open Source para detección y respuesta activa frente a intrusiones. De cada herramienta identificada se realiza una breve presentación de sus características principales, funcionalidad, tecnologías usadas y capacidad de interoperabilidad con otras herramientas.

2.1 CONTEXTO

La seguridad informática se ha vuelto imprescindible para cualquier organización, debido al riesgo de ser víctimas de ataques, los cuales usan técnicas cada vez más eficientes para llevar a cabo su objetivo. Esto ha hecho que se planteen nuevos modelos de seguridad [65].

Los modelos o soluciones de seguridad modernas, se diseñan e implementan basándose en requerimientos como: monitorización en tiempo real de la red, mitigación de ataques, detección de intrusiones, generación de reportes de intrusiones detectadas, interoperabilidad con herramientas externas de seguridad, etc. [75]. En muchos de los casos, las soluciones propuestas por un determinado fabricante o marca, se especializan en algún requerimiento específico, y para proveer de una solución completa se requiere de una integración de herramientas [45].

En el ámbito comercial, se manejan estándares de fabricación y operación, los cuales permiten una integración entre herramientas que no sean de los mismos fabricantes. La técnica que se utiliza con mayor frecuencia para la integración es el manejo de APIs [78]. Las libertades en las que se basan Open source presentan ventajas de implementación y diseño. La capacidad de gestión, estudio, modificación y mejora de herramientas que nos brinda el Open Source, sirve para adaptar e integrar distintas herramientas de seguridad, con el fin de proveer una solución única y completa [73].

2.2 HERRAMIENTAS OPEN SOURCE [50],[51],[52]

Una solución de seguridad para la respuesta activa frente a intrusiones puede basar su diseño en un IDS. A partir de un IDS se integran módulos que permitan proveer una respuesta activa (sobre varios dispositivos de seguridad), además de un almacenamiento y presentación de reportes de las intrusiones detectadas y respuestas ejecutadas.

2.2.1 IDS

Se han considerado varias características para la presentación de cada herramienta. Dichas características permiten hacer un análisis comparativo entre las herramientas.

2.2.1.1 IDS basados en Red

Las características representan parámetros de comparación, las cuales fueron tomadas en consideración partiendo de la funcionalidad básica que debe proveer un IDS. En la Tabla 2.1 se presenta los IDS basados en red.

Como se puede apreciar, existe gran variedad de IDS, los cuales pueden ser utilizados de acuerdo a los requerimientos de diseño que presente una solución. Las características que tienen Snort y Sagan son bastante similares, con excepción de la velocidad de análisis de paquetes, donde Snort es superior. Además, una ventaja de Snort es que su arquitectura es basada en plugins. Esta característica permite incorporar funcionalidad extra acorde a las necesidades de la organización. Suricata también presenta características de funcionamiento muy considerables cuando se las compara con Snort, llegando incluso a ser más solicitada ya que su instalación es más fácil; no obstante, es importante tener presente el escenario o solución que se busca, en este caso Suricata presenta funcionalidad de IPS, pero, pueden existir proyectos, como es el caso del presente, en donde se requiere la interacción con varios dispositivos de seguridad.

Sistemas de Detección de Intrusiones					
Características	Snort [58] [59] [60] [61] [69] [70]	Bro [58] [59] [60]	Suricata [58] [59] [60]	Shadow [61] [69] [70]	Sagan [61] [69] [70]
Fuente de información que se monitorea	Red	Red	Red	Red	Red
Principio de detección o Motor de análisis	Reglas	Anomalías	Reglas	Anomalías	Reglas
Personalización de reglas	SI	NO	SI	NO	SI
Estrategia de control	Distribuida	Distribuida	Distribuida	Central	Distribuido
Características de IPS	Snort Inline(componente interno) Snortsam (componente externo)	Basado en scripts	Opcional (al momento de iniciar el IDS)	NO	Snortsam (componente externo)
Interoperabilidad	UNIX Windows	UNIX	UNIX Windows	UNIX	UNIX
Dificultad de instalación y configuración	Alta	Alta	Baja	Baja	----
Integración con herramientas de análisis	SI (Sguil)	SI (scripts)	SI (Sguil)	NO	SI (Sagan)
Throughput (velocidad de análisis de paquetes)	Alto	Alto	Alto	Bajo (Mbps)	Bajo
Detección en tiempo real	SI	SI	SI	NO	SI
Características Avanzadas					
Proyectos Investigación para Generación de nuevas reglas	Talos[28]	No evaluado	Talos	---	---
Procesador Multi-Threaded	NO	NO	SI	SI	SI

Tabla 2.1 Herramientas NIDS Open Source (1/2)

[28] Talos.- Antes llamado VRT, es un grupo de expertos en seguridad quienes trabajan y aportan para la creación de nuevas reglas, las cuales permitan detectar ataques modernos como ataques de día 0.

Sistemas de Detección de Intrusiones					
Características	Snort [58] [59] [60] [61] [69] [70]	Bro [58] [59] [60]	Suricata [58] [59] [60]	Shadow [61] [69] [70]	Sagan [61] [69] [70]
Documentación	Documentación en sitio web oficial y otros sitios de Internet	Documentación en sitio web oficial y otros sitios de Internet	Pocos recursos en Internet	Pocos recursos en Internet	Documentación en sitio web oficial
Entornos visuales y manejos de Base de datos (BDD)	Frontends (Sguil, BASE, Aanval, prelude)	NO	Frontends (Sguil, BASE, Aanval, prelude)	Interfaz propia	SI
Comunidad de Usuarios	SI	NO	NO	SI (CIDER)	SI
Generación de alarmas (sometidos al mismo nivel de tráfico – generando un archivo tcpdump de 40GB)	Alta 408930 53m 19s	Media 95584 27 m 51 s	Baja 28243 > 1día	---	---
Sensibilidad (capacidad de generar alertas)	Alta (configurable)	Medio (genera registros por cada tipo de ataque)	Baja (poco configurable, Lento en escritura de logs)	---	Alta (configurable)

Tabla 2.1 Herramientas NIDS Open Source (2/2)

La personalización de reglas es también una característica compartida por los IDSs Snort, Suricata y Sagan. Esto permitirá definir un nivel de sensibilidad del IDS acorde a la realidad identificada por el administrador. Otro aspecto importante a considerar, es la documentación, y de ser posible el nivel de soporte que esta pueda brindar. En base a estos aspectos se podrá configurar la herramienta de manera apropiada.

A pesar de presentar una amplia gama de características, los IDS aún no perfeccionan ciertos aspectos que son críticos al momento de implementarlos. Por

ejemplo los IDSs basados en reglas esperan tiempos considerables hasta que se genere una nueva regla para un ataque moderno, generan falsos negativos, no ejecutan respuestas de forma automática, etc. Para solventar estos inconvenientes y ofrecer soluciones más robustas, los IDS deben integrarse a otras herramientas de seguridad.

2.2.1.2 IDS basados en Host

Los HIDS se enfocan en el análisis de una máquina en particular, por tal motivo, basan su diseño en características como, LIDS, PGP, detección rootkit, etc. La característica LIDS se la utiliza para detectar el mal uso de una computadora, políticas de violación y otras actividades inapropiadas. La detección de rootkit se logra a través del motor basado en anomalías con el que puede contar el HIDS. PGP que protege la integridad de los archivos basándose en técnicas de encriptación. Como se puede apreciar en la Tabla 2.2 los HIDS basan su funcionamiento en la detección por comportamientos anómalos.

Esto significa que requieren un entrenamiento previo con el funcionamiento normal del sistema. Esta es una característica importante ya que es posible detectar vulnerabilidades de día cero. Otro factor importante es que las herramientas descritas pueden monitorear sistemas operativos basados en Unix y Windows.
Un aspecto importante es que Ossec presenta una arquitectura flexible capaz de incorporar scripts externos para ejecutar respuestas ante la detección de intrusiones.

El análisis tanto de HIDS como de NIDS se enfoca en validar las soluciones basándose en una revisión de varias fuentes bibliográficas con sus análisis pertienentes, de tal manera que no habrá la necesidad de configurar cad una de las herramientas, debido que el hacerlo demandaría una gran cantidad de tiempo y no es el foco del presente proyecto. Como se puede apreciar en la Tabla 2.2 las características de un HIDS varían significativamente a un NIDS, esto se debe a que ambas soluciones tienen fuente de monitorización totalmente distinta.

Sistemas de Detección de Intrusiones				
Características	Ossec [62] [71] [72]	Samhain [62] [71]	Osiris [62] [71]	OpenDlp [72]
Principios de Detección o motor de análisis	Anomalías	Anomalías	Anomalías	Anomalías
Características de IPS	SI (al momento de instalar)	SI	NO	NO
Detección Rootkit29	SI	SI	NO	SI
Característica de LIDS30	SI	NO	NO	NO
Interoperabilidad	Unix - Windows	Unix – Windows (Servidores)	Unix - Windows	Unix - Windows
Dificultad de Instalación y Configuración	Baja	Media	Alta	Medio
Integración con herramientas de análisis	Herramientas propias del sistema	---	NO	NO
Características Avanzadas				
PGP signed31	SI	SI	SI	NO
Requerimientos (programas que se requieren para su funcionamiento)	---	GnuPG (solo si se utiliza seguridad en la base de datos)	OpenSSL	OpenSSL
Opciones de LOG	Log central	Syslog, Scripts Externos	Log central	Log central
Documentación	Documentación en sitio web oficial y otros sitios de Internet	Bajos recursos en Internet	Bajos recursos en Internet	Bajos recursos en Internet
Configuración Respuesta Activa	Tiempo de bloqueo, respuesta local o remota en la red	---	---	NO
Interfaz GUI y manejo de BDD	SI	SI	Herramientas externas	SI

Tabla 2.2 Herramientas HIDS Open Source

[29] Rootkit.- Conjunto de herramientas utilizadas para entrar ilegítimamente a un sistema

[30] LIDS (Log Analysis for intrusion detection).- Es el proceso o técnica usado para detectar ataques en un ambiente específico usando logs como primera fuente de información.

[31] PGP(Pretty good privacy).- Seguridad basada en encriptación y autenticación

2.2.2 IPS

En la Tabla 2.3 se muestra herramientas que proveen técnicas de prevención frente a intrusiones.

Sistemas de Detección de Intrusiones					
Características	Snortsam [49]	FWSnort [84]	Snortinline	Ossec [80]	OpenWIPS -NG
Fuente de información monitoreada	Red	Red	Red	Host	Wireless
Acción de respuesta activa	Bloqueo de tráfico	Bloqueo de tráfico	Bloqueo de tráfico	Bloqueo Firewall (Iptables ipfilter ipfw firewalld) Negación de Host	Bloqueo de tráfico
Acción de respuesta	Distribuida (IRS)	Local	Local	Local - Distribuida	Distribuida
Tecnología utilizada	Plugin de salida de Snort	Traduce reglas de Snort a Reglas de Netfilter	Versión modificada de Snort, crea reglas parta detección.	Creación de comandos Basada en scripts	Sensor Servidor Interface(GUI)
GUI de respuestas ejecutadas	No	No	No	No	Sí
GUI de respuestas ejecutadas	No	No	No	No	Sí
Interoperabilidad	Firewalls Open Source Y Comerciales	Linux	Linux	Windows y Linux	Linux
Línea con el tráfico	No	Si	Sí	No	No
Documentación	Documentación en Sitio web oficial y otros sitios de Internet	Sitios web en Internet	Documentación en Sitio web oficial y otros sitios de Internet	Documentación en Sitio web oficial y otros sitios de Internet	Documentación en Sitio web oficial y otros sitios de Internet

Tabla 2.3 Herramientas IPS Open Source

Como se puede apreciar tanto Ossec como Fail2ban trabajan a nivel de host, mientras que Snortsam, FWSnort, SnortInline y OpenWIPS usan el tráfico de red como fuente de información. Sería de suma importancia que una solución de seguridad incluya al menos una de cada tipo, de tal manera que la cobertura de protección sea mayor dentro de la organización.

En general, todas las herramientas Open Source descritas en la Tabla realizan bloqueo de tráfico ante una intrusión detectada. Dicho bloqueo requiere de una interacción con algún dispositivo de seguridad perimetral, tal como un firewall (reglas de filtrado) o un router (listas de control de acceso). No obstante, tanto Snortsam como Ossec, dada su arquitectura modular y extensible, brinda la posibilidad de implementar plugins para ejecutar otros tipos de respuestas.

Del estudio realizado también se determinó que Snortsam, Ossec y OpenWIPS son capaces de ejecutar acciones de respuesta de manera distribuida. Es decir que basan su funcionamiento en agentes remotos que son los que realmente ejecutan la respuesta. Esta es una característica de suma importancia dentro de un IRS, que como se conoce debe tener la posibilidad de interactuar con varios dispositivos de seguridad remotos.

Como se puede observar, una de las falencias que presentan la mayoría de IPSs, a excepción de OpenWISP, es que no proveen de una interfaz gráfica que permita observar las respuestas ejecutadas. En algunos casos, ni siquiera se almacenan logs de las mismas. Una posible solución a esta falencia, podría ser implementar un sistema que permita tomar los eventos generados por las respuestas, guardarlos en una base de datos, y posteriormente presentarlos a través de una interfaz web.

2.2.3 GESTIÓN DE EVENTOS DE SEGURIDAD A TRAVÉS DE UNA INTERFAZ GRÁFICA

Una vez que se ha logrado detectar un comportamiento anómalo en un host o una red, este debe registrarse, y posteriormente presentarse a través de una interfaz

gráfica, esto permitirá al encargado de seguridad revisar y analizar los eventos que han sido generados. Los historiales y estadísticas de los eventos generados son de suma importancia dentro una organización, ya que son insumos para realizar un análisis y evaluación de riesgos. Existen diferentes herramientas Open Source que permiten presentar los eventos generados por un IDS, las mismas se presentan en la Tabla 2.4

Gestión de eventos de seguridad				
Herramientas	IDS Soportados	Herramientas Gráficas	Dashboards personalizables	Documentación
SquertProject (mejora de Sguil) [86]	Snort	Si	Si	Sitio web oficial y otros sitios de Internet
Prelude [91]	Suricata Snort Ossec	Si	Si	Sitio web oficial y otros sitios de Internet
Security Onion	Snort Suricata Bro Ossec Sguil	Sí	No	Sitio web oficial y otros sitios de Internet
SnortReport[88]	Snort	Si	No	Sitio web oficial y otros sitios de Internet
Sguil[89]	IDS propio Snort	No	No	Sitios de Internet
Base[47] [72]	Snort	Si	No	Sitio web oficial y otros sitios de Internet
Ossec GUI[80]	Ossec	Si	No	Sitio web oficial y otros sitios de Internet
Aanval[90]	Snort Suricata Cisco Baracuda Emerging Therats	Si	No	Sitio web oficial y otros sitios de Internet

Tabla 2.4 Herramientas de presentación y gestión de eventos IDS Open Source

Lo que cabe destacar de los datos recopilados es que la mayoría de herramientas reportadas soportan Snort. Los dashboards personalizables están presentes en SquerProject y Prelude, en estas herramientas se puede elegir que contenido se quiere mostrar. A excepción de Squil, todas poseen herramientas gráficas que

muestrán estadísticos de las intrusiones detectadas. Para seleccionar una de estas herramientas se debe conocer previamente qué IDSs se emplearán.

2.3 SOLUCIONES COMERCIALES

Comercialmente no se hace una distinción entre tecnologías IDS, IPS o IRS. La mayoría de fabricantes adquiere el nombre de IPS involucrando características de las tres tecnologías [57]. La revisión de herramientas comerciales no será detallada como se hizo con las herramientas open source, debido a que no serán utilizadas en el proyecto, pero, servirán para tomar ciertos criterios al momento de diseñar y desplegar la solución de seguridad. La manera más práctica y simple de investigar una herramienta en particular y saber cómo esta se encuentra en el mercado laboral es revisando el cuadrante mágico de Gartner, el cual consiste en un análisis comparativo entre los diferentes productos correspondientes a diferentes áreas tecnológicas [55].
Los IPSs que se encuentran liderando el mercado en el último año son: IBM, Cisco, e Intel Security (McAfee) [56].

De igual manera, realizando una revisión al cuadrante de Gartner de SIEM[32] del 2015[77], se presentan como líderes a IBM, HP, Splunk, Intel y LogRhythm. Los SIEM permiten observar a través de un interfaz web las intrusiones detectadas por IDS. Dentro de las características principales que presentan las herramientas comerciales son [94]:

- Soporte técnico.
- Garantía de producto
- Capacitaciones y certificaciones de operación
- Corrección de bugs en producción

Existen soluciones comerciales muy avanzadas que brindan protección contra ataques DoS y DDoS, el inconveniente principal es el costo comercial que estos implican, de la misma manera sucede con la implementación y validación de estas soluciones sobre una infraestructura física.

[32] **SIEM (Security Information Event Manager).-** Correlacionador de eventos de seguridad.

CAPÍTULO III

DISEÑO DE LA SOLUCIÓN

El presente capítulo tiene la finalidad de presentar el diseño de la solución que contempla un sistema de respuesta activa frente a ataques de denegación de servicio. En primera instancia se presenta un conjunto de requerimientos a los que la presente solución pretende solventar. Posteriormente se muestra la arquitectura diseñada y se describen sus componentes. Finalmente, en base a la revisión del estado de la tecnología realizada en el capítulo anterior, se establecen criterios y se seleccionan las herramientas que conformarán la arquitectura antes diseñada.

3.1 PROBLEMÁTICA

Los ataques DoS han tenido un crecimiento importante en los últimos años [73], su éxito radica en la eficacia, potencialidad y hasta cierto punto facilidad de su ejecución. Una de las maneras básicas de abordar un ataque DoS, se trata de utilizar puertos que normalmente se encuentran abiertos en el firewall 80(HTTP), 25 (SMTP), 443 (SSH) o 53 (DNS). Las pérdidas que se puede llegar a tener una organización tras ser víctimas de un ataque DoS, pueden ser significativas [75]. Otro aspecto a tomar en consideración al intentar proteger la red de organización con un firewall, es el cuello de botella que se formaría al ser víctimas de un ataque DoS.

El firewall consumirá sus recursos de ancho de banda y procesamiento al recibir un número alto de peticiones, en intervalos de tiempo pequeño, y por ser la única puerta de entrada a la red, obstaculizará el flujo de tráfico entrante y saliente de la misma.

Es por tanto necesario pensar en una solución que complemente los servicios proporcionados por el firewall, ya que este no detecta ni protege a la red de manera adecuada cuando se enfrenta a un ataque de denegación de servicio simple o distribuido. Cualquier solución que se implemente, deberá tener la capacidad de configurar la detección y protección contra un ataque específico, tratando de cubrir la mayor cantidad de amenazas que este implique.

Las consideraciones que se debe tener para actuar ante ataques DoS son varias, una de ellas es tener la habilidad de no ser detectada. Esto ocurre con mayor frecuencia cuando se ejecuta un ataque DoS orientado a la capa de aplicación, en la cual se establece una conexión legítima y dentro de ella se lleva a cabo el ataque, a esta vulnerabilidad presente en las herramientas de detección se la conoce como falso positivo, el cual puede ser solventado mediante la creación o configuración de reglas específicas para el ataque.

Otra consideración es que un ataque DoS puede ser generado tanto desde el exterior de una red de organización, como del interior de la misma. Muchas de las soluciones no toman en cuenta los ataques generados desde el interior, vulnerabilidad aprovechada ampliamente por el atacante. Proveer una solución contra ataques DoS con herramientas comerciales, conlleva un análisis técnico y comercial. Debido a que la protección puede implicar la integración de varias herramientas de seguridad, pudiendo darse el caso que estas no sean del mismo fabricante. Los costos que representan la adquisición, configuración, implementación, capacitación para la operación de herramientas, muchas de las veces no pueden ser cubiertos por las organizaciones.

3.2 REQUERIMIENTOS DEL DISEÑO

Los requerimientos para diseñar la solución de seguridad se basan en el planteamiento de los objetivos del proyecto, en la problemática previamente descrita y otros que han surgido de la revisión del estado de la tecnología del capítulo II. Se plantea los requerimientos con la finalidad de tener un panorama claro y conciso, a través del cual se pueda diseñar una solución de seguridad completa, utilizando herramientas Open Source adecuadas. Los requerimientos iniciales son:

- Desplegar una red con topología jerárquica.
- Proveer una solución de seguridad sin incurrir en gastos de inversión por adquisición de herramientas comerciales.
- Proveer la detección tanto a nivel de red como a nivel de host.
- Disminuir la cantidad de falsos positivos.

- Manejo de falsos negativos
- Detección de ataques conocidos y ataques nuevos
- Ejecutar una respuesta activa ante la detección de intrusiones.
- Proteger la red contra ataques DoS provenientes desde el exterior e interior de la red con topología jerárquica.
- Almacenar los eventos de seguridad tales como intrusiones detectadas y respuestas ejecutadas.
- Integrar herramientas de gestión y presentación de eventos de seguridad.

3.3 ARQUITECTURA PROPUESTA PARA LA SOLUCIÓN

La arquitectura en la cual se basa el diseño de solución de seguridad se presenta en la Figura 3.1.

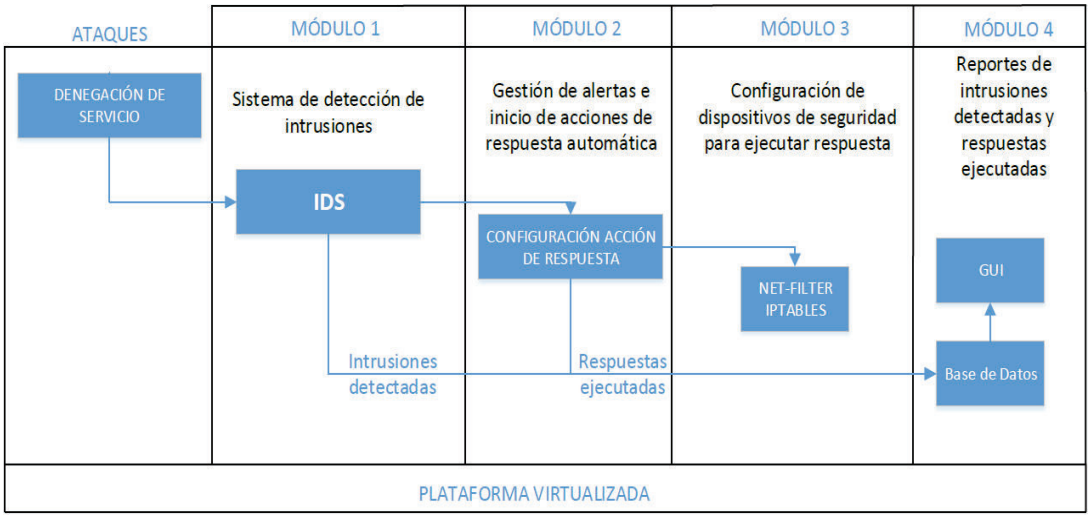


Figura 3.1 Arquitectura propuesta

Como se puede ver, esta se compone de 4 módulos. Para cada módulo se realizará la selección, configuración e integración de herramientas, de tal manera que se provea la ejecución de una respuesta activa al detectar un ataque DoS.

3.3.1 Modulo 1: IDS

Consiste en la implementación de sistemas de detección de intrusiones, lo cuales se localizan en lugares estratégicos, con la finalidad de proveer una protección en

toda la red jerárquica, monitoreando el flujo de la red y detectando comportamientos anómalos sobre la misma. Cada IDS implementado deberá tener la capacidad de generar logs los cuales puedan ser almacenados en una base de datos, así se podrá hacer uso de ellos para una presentación a través de una interfaz gráfica.

3.3.2 MÓDULO 2 – GESTIÓN DE ALERTAS, INICIO DE RESPUESTA AUTOMÁTICA

Este módulo provee la capacidad de configurar e iniciar una respuesta activa en función de la detección de una intrusión, ya sea habilitando el componente de respuesta activa presente en un IDS (Ossec) o configurando una herramienta externa que provea de esta funcionalidad (Snortsam). Las respuestas activas se inician a partir de la detección de una intrusión, por lo tanto deberán ser configuradas en el motor de análisis que utilice el IDS. Esta acción de respuesta deberá ser almacenada en logs o en una base de datos, de tal manera que pueda ser posteriormente presentada a través de una interfaz gráfica.

3.3.3 MÓDULO 3 – COMPONENTE DE SEGURIDAD

Este módulo llevará a cabo la ejecución de la respuesta activa. Una vez que se ha configurado el mecanismo de detección de intrusiones (módulo 1) y se haya configurado la respuesta activa (módulo 2), se enviará una orden o tarea a un componente de seguridad, el cual puede ser un firewall que se encarga de hacer efectiva la ejecución de la respuesta.

3.3.4 MÓDULO 4 – BDD[33] Y GUI[34]

Se compone de 2 elementos, el primero de ellos consiste en la configuración e implementación de una base de datos, la cual se integra a los módulos 1 y 2 para almacenar los eventos (detección de intrusiones y generación de respuestas) que aquí se generen.

---

33 BDD: Base de Datos
34 GUI: Interfaz gráfica de Usuario

El segundo elemento consta de una interfaz gráfica a través de la cual se presenta información los reportes generados por cada herramienta utilizada en la solución.

3.4 SELECCIÓN DE HERRAMIENTAS

Con fin de cumplir el requerimiento de no incurrir en gastos de inversión ni operativos de las herramientas seleccionadas, se tomaron en cuenta únicamente herramientas Open Source gratuitas. Al igual que software libre el open source maneja el término *free* el cual es mal interpretado como gratis, cuando en realidad se refiere a la libertad de ejecutar, distribuir, estudiar, modificar y mejorar el software [74].

De esta manera es como se promueve libertades de uso y de implementación de herramientas basada en esta tecnología. Una de las libertades hace referencia a la adaptación de la herramienta a las necesidades propias de la persona que la vaya a implementar. Esto permite que las herramientas seleccionadas se han configuradas de tal manera que cumplan con los requerimientos propuestos, y se integren a una solución común.

En las siguientes secciones se presentan los criterios utilizados para selección de las herramientas que conformarán la solución.

3.4.1 CRITERIOS PARA SELECCIÓN DE IDSS

El criterio para la selección de un IDS, se encuentra determinado en base a la revisión de estado de tecnología y fundamentación teórica. Básicamente debe cumplir los siguientes criterios.

- Criterio 1 (C1): Documentación para manejo y configuración de la herramienta.
- Criterio 2 (C2): Gestión de falsos negativos; es decir el IDS deberá presentar algún mecanismo que permita gestionar la generación de falsas alarmas.
- Criterio 3 (C3): Disminución de los falsos positivos; es decir que un IDS debe permitir la personalización de las reglas de detección.

- Criterio 4 (C4): Detección de ataques internos y externos; es decir que el IDS debe tener la capacidad de ser localizado en lugares estratégicos de la red.
- Criterio 5 (C5): Almacenamiento de las intrusiones detectadas, ya sea en logs o una base de datos, aquellas intrusiones que han sido detectadas.
- Criterio 6 (C6): Análisis de integridad de un host; para garantizar el correcto funcionamiento y disponibilidad de un host este deberá ser monitoreado en busca de alguna anomalía.

IDS						
Herramienta	C1	C2	C3	C4	C5	C6
Snort	X	X	X	X	X	
Bro	X		X	X	X	
Suricata			X	X	X	
Shadow			X		X	
Sagan	X	X	X	X	X	
Ossec	X		X	X	X	X
Samhain			X	X	X	X
Osiris				X	X	X

Tabla 3.1 Evaluación de IDS según criterios de selección.

3.4.2 CRITERIOS PARA SELECCIÓN DE IRSS

De igual manera que los IDS, se presenta a continuación los criterios básicos que debe presentar un IRS.

- Criterio 1 (C1): Interoperabilidad con IDS; ya sea un componente más o un plugin que pueda integrarse al motor de análisis del IDS, deberá contar con la capacidad de iniciar una respuesta activa al haber sido detectada una intrusión.
- Criterio 2 (C2): Configuración de respuesta; la cual permita configurar la acción más adecuada para ser llevada a cabo, de tal manera que la acción de protección no cause un mayor daño que el propio ataque, esto es posible si se trabaja directamente con el motor de detección, independientemente de cual sea este.

- Criterio 3 (C3): Interoperabilidad con un dispositivo de seguridad para ejecutar una respuesta. Una vez que se inicie la acción de respuesta esta busca un dispositivo o herramienta de seguridad donde pueda ser ejecutada.
- Criterio 4 (C4): Almacenar respuestas ejecutadas, una vez que se haya llevado a cabo la acción de respuesta, esta debe ser almacenada en un base de datos o como log en un archivo.

IRS				
Herramienta	C1	C2	C3	C4
Snortsam	X	X	X	
FWSnort	X		X	
Snort_in _Line	X		X	
Ossec	X		X	
OpenWIPS-NG	X	X	X	X
Fail2ban	X		X	

Tabla 3.2 Evaluación de IRS según criterios de selección.

3.4.3 CRITERIOS PARA SELECCIÓN DE DISPOSITIVOS DE SEGURIDAD.

- Criterio 1 (C1): Receptar respuestas enviadas por IRS; establecer comunicación con el IRS, de preferencia deberá hacerlo con una conexión cifrada.
- Criterio 2 (C2): Ejecutar la acción de respuesta, una vez que se ha establecido la comunicación entre en IRS y el dispositivo de seguridad.

Dispositivos de seguridad		
Herramienta	C1	C2
Iptables	X	X
ACLs		X

Tabla 3.3 Evaluación de Dispositivos de seguridad según criterios de selección.

3.4.4 BASE DE DATOS E GUI

- Criterio 1 (C1): Almacenar intrusiones detectadas en bases de datos, es decir que deberán integrarse al módulo 1.
- Criterio 2 (C2): Almacenar respuestas ejecutadas; integración con módulo 2.
- Criterio 3 (C3): Interfaz de presentación; el interfaz gráfico deberá conectarse a la base de datos y presentar la información almacenada por la misma.
- Criterio 4 (C4): Gestionar base de datos; el interfaz gráfico también deberá interactuar con la base de datos, de tal manera que las presentaciones de los valores que se requiera ver sean dinámicas.

BDD & GUI				
Herramienta	C1	C2	C3	C4
SquertProject (mejora de Sguil)	X		X	X
Security Onion	X		X	X
Prelude	**X**		**X**	**X**
SnortReport	X		X	
Base	X		X	X
Ossec GUI	X		X	X
Aanval	X		X	

Tabla 3.4 Evaluación de GUI según criterios de selección

Una vez presentado el estado de la tecnología y los criterios de selección, se realiza un emparejamiento de la matriz de requerimientos con herramientas que puedan cumplir cada uno de los items instanciados.

La Tabla 3.5 muestra la justificación de cada herramienta seleccionada para cumplir con el requerimiento propuesto en cada uno de los módulos, es decir IDS, IPS y la presentación de eventos a través de una interfaz gráfica.

Requerimientos	Herramienta Tecnología	Justificación
Desplegar una red con topología jerárquica.	VNX	Herramienta Open source que permite diseñar y desplegar escenarios de red basados en XML.
Proveer una solución de seguridad sin incurrir en gastos de inversión por adquisición de herramientas comerciales.	Herramientas Open Source	Aprovechando las libertades que nos ofrece el open source, se descargan las herramientas y se las configura de acuerdo a los requerimientos y objetivos del proyecto.
Proveer una detección a nivel de red: En base al monitoreo de tráfico de red y detección de comportamiento anómalo	Snort	Aprovechar la funcionalidad de IDS basado en red que ofrece Snort, herramienta de seguridad que puede ser localizada estratégicamente dentro de la topología de red jerárquica.
Proveer una detección a nivel de host: Analizando y comprobando la integridad de la configuración de un sistema operativo	Ossec	Tiene la capacidad de realizar un completo análisis de la configuración, comportamiento e integridad de un sistema operativo.
Reducir de generación de falsos positivos	Snort Ossec	Snort genera reglas específicas para cada ataque. OSSEC permite configurar un conjunto de reglas, mediante las cuales se puede configurar comportamientos más específicos.
Controlar y gestionar falsos negativos	Snort	Creación de listas blancas, de tal manera que habrá servidores en los cuales no se apliquen las reglas definidas en Snort.
Detección de ataques conocidos, ataques modernos y rootkit35	Snort Ossec	Snort basado en reglas personalízales. Ossec basado en detección de anomalías y un análisis al file system de un sistema operativo.
Ejecutar una respuesta activa ante la detección de intrusiones.	Snortsam Respuesta activa de Ossec	Snortsam es un plugin de salida de Snort, permite que este deje de ser IDS y pase a ser un IRS, es decir que inicia una acción de respuesta ante una intrusión detectada. De la misma manera sucede con la activación del componente de respuesta activa de Ossec.

Tabla 3.5 Selección de herramientas para cumplimiento de Requerimientos. (1/2)

35 Conjunto de herramientas utilizadas por un atacante para ocultar aquellos procesos y archivos que l permiten mantener el acceso al sistema.

Requerimientos	Herramienta Tecnología	Justificación
Proteger la red contra ataques DoS provenientes desde el exterior e interior de la red con topología jerárquica.	Snort Ossec	Snort permite la localización estratégica de sensores en puntos sensibles de la red. OSSEC presenta una arquitectura cliente servidor, la cual permite configurar agentes en cada máquina virtual que se requiere monitorear. Estas características que presentan ambos IDS proveen una defensa contra ataques internos y externos.
Almacenar eventos de seguridad tales como intrusiones detectadas y respuestas ejecutadas	Barnyard2 Base de datos de Ossec	Barnyard2 permite dar un formato a las alerta generadas por Snort para que puedan ser almacenadas en un BDD de manera rápida.La base de datos de Ossec se configura con la finalidad de almacenar todos los eventos de seguridad, para posteriormente hacer uso de ellos según lo que se requiera.
Integrar herramientas de gestión y presentación de eventos de seguridad	BASE Ossec GUI	BASE y Ossec GUI se integran a la BDD de Snort y Ossec, para administrarla y presentar los datos tras una interfaz gráfica.

Tabla 3.5 Selección de herramientas para cumplimiento de Requerimientos. (2/2)

3.5 HERRAMIENTAS SELECCIONADAS PARA CONFORMAR LA ARQUITECTURA

3.5.1 MÓDULO DE IDS

En la Figura 3.2 se puede ver cada uno de los módulos de la solución. El primer módulo se encuentra oonformado por Snort y Ossec, seleccionados con la finalidad de detectar amenazas de un ataque DoS. Las actividades que se realizan en este módulo son:

- Generación y configuración de reglas para detección de ataques DoS.
- Monitoreo de comportamiento de hosts y de red.
- Generación de alertas en caso de detectar comportamientos anómalos.
- Envío de eventos de seguridad a bases de datos correspondientes.
- Despliegue de IDS en lugares estratégicos en la red, de tal manera que la protección sea completa en cuanto a cobertura de la red.

3.5.2 MÓDULO DE INICIO DE RESPUESTA ACTIVA

En este módulo se encuentran herramientas que permiten iniciar una respuesta activa. Para la solución se ha optado por Snortsam, y el componente de respuesta activa de Ossec. Las tareas principales que se realizarán en este módulo son:

- Configuración de respuesta activa en reglas de Snort a través del plugin de Snortsam.
- Habilitación de respuesta activa en Ossec.
- Configuración de respuesta activa en alertas, nivel de alertas o conjunto de reglas de Ossec.
- Ejecución de script que permite almacenar las respuestas ejecutadas por Snortsam y Ossec, en base de datos correspondiente.

3.5.3 MÓDULO DE DISPOSITIVOS DE SEGURIDAD

Consistirá en la configuración de la conexión adecuada entre Snortsam y el firewall perimetral, además de la conexión entre los agentes Ossec y servidor Ossec a través de las siguientes tareas.

- Configuración de Snortsam para que recepte las acciones de respuesta que provienen desde los sensores de Snort, para lo cual se crea una conexión tcp cifrada a través del puerto 898.
- Ejecución de respuesta activa en el firewall perimetral.
- Añadir los agentes de Ossec en el servidor Ossec, esta tarea se le realiza tras la configuración de un enlace cifrado entre el cliente y el servidor a través del puerto 1514.
- Ejecución de respuesta activa a través de los Iptables de cada host donde se encuentra instalado el agente de Ossec.

3.5.4 MÓDULO DE BASES DE DATOS E INTERFAZ GRÁFICA

La base de datos contiene dos instancias de base de datos, la primera de ellas es asignada para Snortsam, dentro de la cual se crea una tabla denominada logs conformada por los siguientes campos, Fecha (date), Hora(varchar), DireccionIP(varchar), Herramienta e Informe.

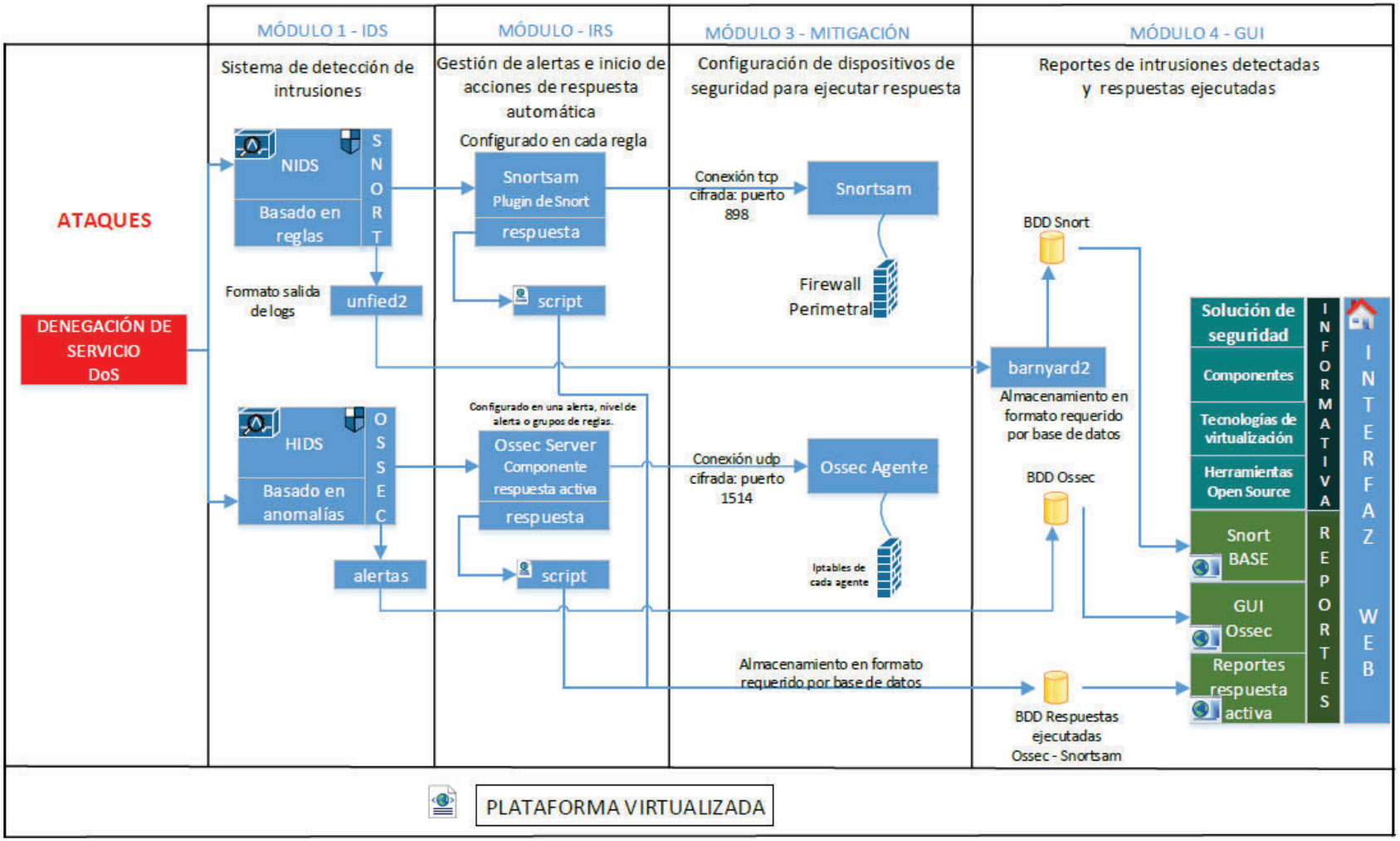


Figura 3.2 Diagrama de despliegue de arquitectura

La segunda instancia es asignada a Ossec, la tabla creada también se la llama logs, pero a diferencia de la tabla de Snortsam, esta tiene los siguientes campos, Diasemana, Mes, IPorige, diames, hora, Meridiano, año, path, AD, espacio, IPblock, Proceso, IDIntrussion.

Los campos provienen de los logs generados por Ossec al ejecutar una respuesta activa.

- Bases de Datos:
    - Gestión de BDD Ossec: Se encuentra en el servidor Ossec, y es configurada cuando se levantan los servicios de interfaz gráfica. A esta base de datos llegarán todos los eventos detectados por los agentes Ossec.
    - Gestión de BDD Snort: Los datos que se registran en la base de datos de Snort serán ingresados con la ayuda de la herramienta barnyard2, la cual provee de un mecanismo que optimiza la capacidad de almacenar alertas generadas por Snort en una base de datos.
    - Gestión de BDD de respuestas ejecutadas: Base de datos creada con la finalidad de almacenar las respuestas ejecutadas por Snortsam y Ossec.

En lo que respecta la interfaz gráfica, es creada a partir de la necesidad de presentar todos los GUIs configurados para cada herramienta desde un mismo portal, a su vez presenta una revisión general de las características principales del proyecto. En la Figura 3.3, se presenta el prototipo de las interfaces y su navegabilidad.

- Interfaz Gráfica:
    - Creación de una interfaz gráfica basada en lenguaje php, por medio de la cual se provee de un portal para presentar información del proyecto, y también los reportes generados en base a las intrusiones detectadas y respuestas ejecutadas.
    - La interfaz gráfica permitirá una interrelación entre herramientas de presentación de reportes como los es Base, y Ossec web UI, pero también constará de un módulo (verde) que permitirá conectarse a

lasa base de datos de Snortsam y Ossec, de tal manera que mediante una consulta basada en una fecha, se puedan extraer los datos detectados y respuestas ejecutadas, generados por las herramientas respectivas.

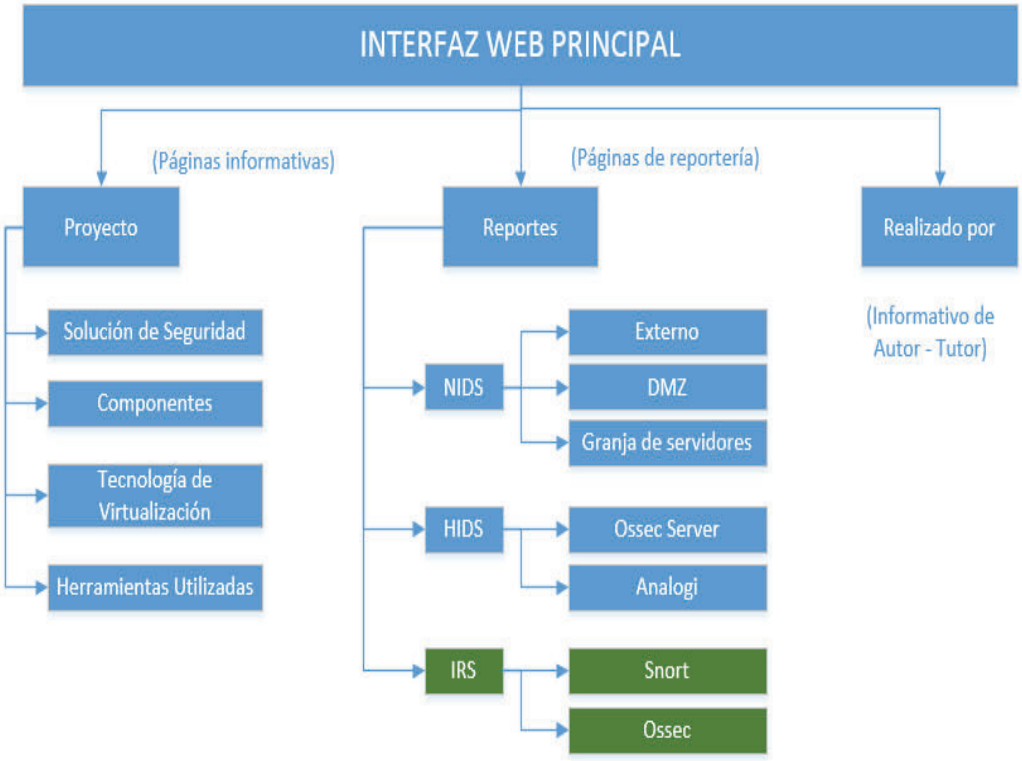


Figura 3.3 Prototipo de Interfaz web principal

CAPÍTULO IV

IMPLEMENTACIÓN Y PRUEBAS

La propuesta de la solución de seguridad, como se mencionó en el capítulo anterior, contempla la integración de varias herramientas Open Source. El presente capítulo muestra el procedimiento que integra los módulos para conseguir la solución de seguridad. Además se realiza la validación de la solución a través de ataques DoS.

4.1 DESPLIEGUE DE LA SOLUCIÓN

El escenario de red virtualizado sobre el cual se despliega la solución de seguridad se realiza a través de VNX. Dicho despliegue se basa en la escritura y programación de archivos XML, dentro de los cuales se instancia las máquinas virtuales, segmentos de red y equipos de conectividad a utilizar en la red.

4.1.1 Presentación del escenario de red virtual

La topología que se implementó para el despliegue de la solución, se puede observar en la Figura 4.1, en la cual se muestran las máquinas virtuales, dispositivos de conexión y dispositivos de seguridad. Los números colocados sobre los dispositivos son identificadores, por medio de los cuales se presentan las características principales y funcionalidad de cada uno de ellos en la Tabla 4.1.

Los recursos de cada una de las máquinas virtuales dependen de su funcionalidad y servicios que provean dentro de la solución, por ejemplo, Snort será una máquina que requiera más recursos de memoria RAM ya que será la encargada de monitorear el tráfico que se transporta por la red a gran velocidad. Pero máquinas que no provean servicios como Metasploitable que es la máquina víctima, en la cual se ejecutarán los ataques, tendrá una menor cantidad de recursos, de la misma manera se asigna poca cantidad de recursos a las imágenes dynamips, ya que estas así lo demandan.

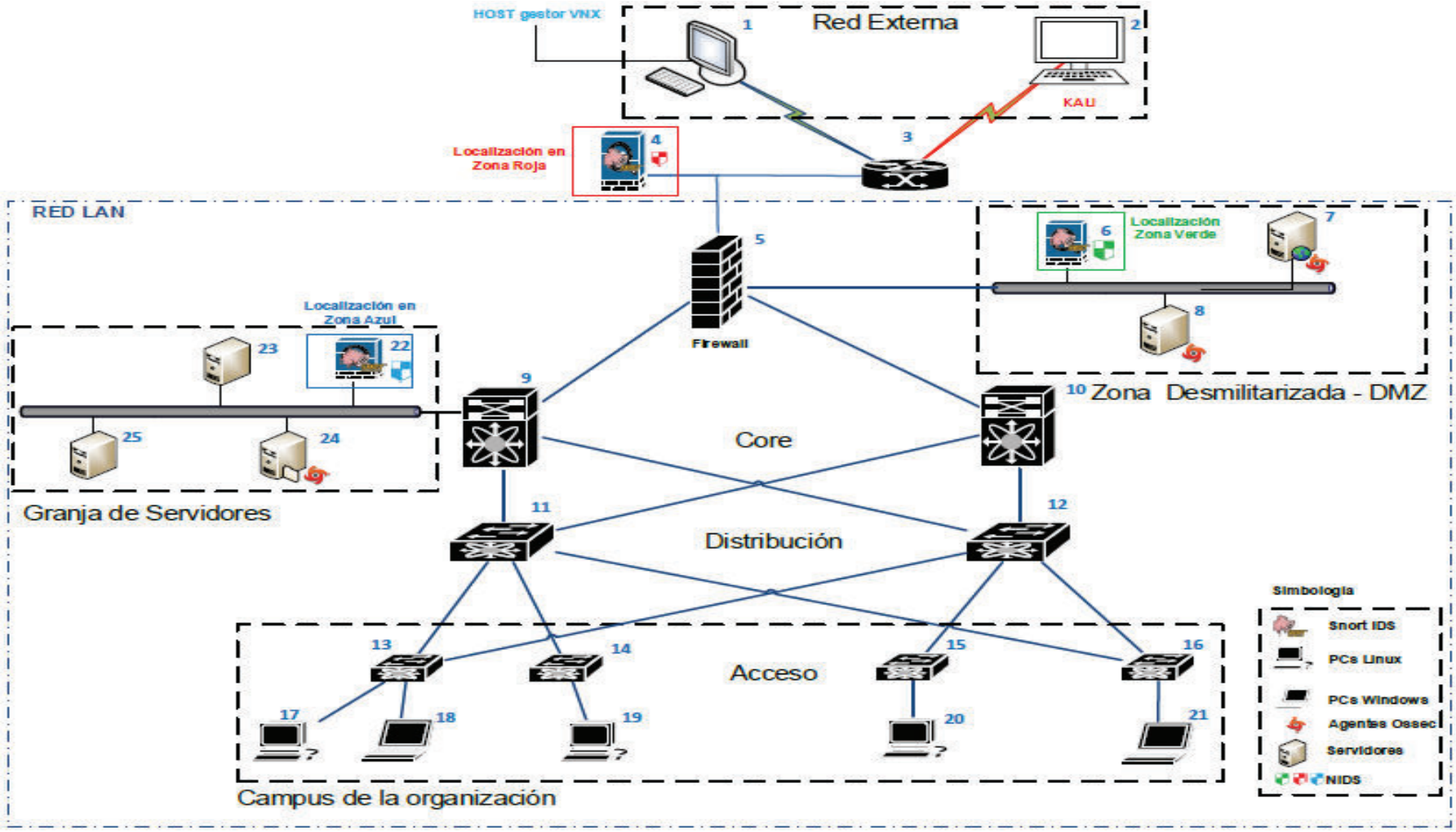


Figura 4.1 Despliegue de solución sobre topología jerárquica

ID	Nombre Máquina Virtual	Tipo de Máquina	Dirección IP	Recursos Memoria Storage vCPU	Herramientas Instaladas	Sistema Operativo	Funcionalidad
1	---	Hypervisor	10.1.2.2	Mem = 8 GB Storage = 1TB Cpu = 4 Cores/cpu = 2	VNX KVM	Ubuntu 14.04	Es el Host físico donde se instala el software necesario para crear los escenarios de red.
2	kali_e (e: externo)	Atacante	10.1.0.2	Mem = 256 MB Storage = 40 GB vCPU = 2	Slowloris Hping3 Fudp	Kali - Debian 3.18	Sistema operativo desde el cual se generan diferentes tipos de ataques DoS, utilizando varias herramientas para la generación de los mismos.
3	router_gw	Redes	10.1.0.1 10.1.1.1 10.1.2.1	vCPU = X mem = 96M (default)	Configuraciones para zona de frontera	Cisco Internetwork Operating System (2007)	Router localizado en la zona de borde de la red, el cual permitirá conectar la red LAN con la red externa (Kali externo - Host físico (laptop Sony Vaio) que administra la plataforma).
4	snort_r (red)	Seguridad	10.1.1.10	Mem = 512 MB Storage = 40 GB vCPU = 2	Snort Plugin de Snortsam Barnyard2 Base	Ubuntu 15.04	Localizado en la parte exterior de la red, denominado zona roja, se encargará de revisar y monitorear los paquetes que ingresen a la red. Inicia una acción de respuesta ante una intrusión detectada. Aquí se cumple el módulo 1 y 2 de la arquitectura propuesta.

Tabla 4.1 Dispositivos implementados en el despliegue de la solución (1/4)

5	firewall	Seguridad	10.1.1.2 10.1.1.3 10.1.1.6 10.1.1.4	Mem = 256 MB Storage = 20 GB vCPU = 2	Iptables Snortsam	Ubuntu 15.04	Se encarga de permitir o denegar el acceso a la red basándose en reglas. También llevará a cabo la ejecución de la respuesta activa que se inició en el IDS.
6	snort_g (green)	Seguridad	10.1.1.20	Mem= 512 MB Storage = 40 GB vCPU = 2	Snort Plugin de Snortsam Barnyard2 Base	Ubuntu 15.04	Localizado a la entrada de la red DMZ, protegiendo los servicios que aquí se levanten. Al igual que el IDS localizado en la zona roja, inicia una acción de respuesta cuando detecta un determinado ataque.
7	osseca_dmz	Servidor	10.1.1.11	Mem = 128 MB Storage = 40 GB vCPU =1	Ossec agente	Ubuntu 15.04	Servidor Ubuntu en el cual ha sido instalado el agente de Ossec, que se encargará de monitorear y revisar la integridad del sistema operativo.
8	metas_dmz	Servidor	10.1.1.12	Mem = 128 MB Storage = 20 GB vCPU = 2	Agente de Ossec	Ubuntu modificado	Se trata de un sistema operativo Ubuntu modificado, el cual ha sido configurado para ser vulnerable.
9	core_1	Redes	10.1.3.5 10.1.4.5 10.1.1.7	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Router que se encarga de interconectar la capa de distribución y conectividad a la granja de servidores.
10	core_2	Redes	10.1.3.6 10.1.4.6 10.1.1.8	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Es el backup de router core_1, cumple las mismas características, por lo tanto debe tener las mismas prestaciones que el router principal.

Tabla 4.1 Dispositivos implementados en el despliegue de la solución (2/4)

11	switch_d1	Redes	10.1.3.3 10.1.4.3	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Switch de distribución 1, enruta el tráfico que ingresa a la capa de acceso y lo envía hacia la capa núcleo o core
12	switch_d2	Redes	10.1.3.4 10.1.4.4	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Switch de distribución 2,(backup de 1) enruta el tráfico que ingresa a la capa de acceso y lo envía hacia la capa núcleo o core
13	switch_a1	Redes	10.1.3.1	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Brinda el acceso a la red a todos los dispositivos que se conectan a esta capa.(manejo de vlan1)
14	switch_a2	Redes	10.1.4.1	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Brinda el acceso a la red a todos los dispositivos que se conectan a esta capa.(manejo de vlan2)
15	switch_a3	Redes	10.1.3.2	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Brinda el acceso a la red a todos los dispositivos que se conectan a esta capa.(manejo de vlan1)
16	switch_a4	Redes	10.1.4.2	vCPU = X mem = 96M (default)	Vlans	Cisco Internetwork Operating System	Brinda el acceso a la red a todos los dispositivos que se conectan a esta capa.(manejo de vlan2)
17	linux_a1	Máquina Virtual	10.1.3.7	Mem = 128 MB Storage = 20 GB vCPU =1	Agente de Ossec	Ubuntu 15.04	Host en la capa de acceso que emula un dispositivo de usuario final dentro de una organización. Sistema operativo Linux.

Tabla 4.1 Dispositivos implementados en el despliegue de la solución (3/4)

18	wind_a1	Máquina Virtual	10.1.3.8	Mem = 512 MB Storage = 30 GB vCPU =1	Agente de Ossec	Windows XP Update 3	Host en la capa de acceso que emula un dispositivo de usuario final dentro de una organización. Sistema operativo Windows.
19	kali_i (i: interno)	Atacante	10.1.4.7	Mem = 256 MB Storage = 40 GB vCPU = 2	Slowloris Hping3 Fudp	Kali - Debian 3.18	Host que iniciará ataques desde el interior de la red.
20	metas_gs	Servidor	10.1.1.17	Mem = 128 MB Storage = 20 GB vCPU = 2	Servicios por defecto levantados Agente de Ossec	Ubuntu modificado	Se trata de un sistema operativo Ubuntu modificado, el cual ha sido configurado para ser vulnerable. Sobre el cual se probará la solución de seguridad.
21	ossecs_gs	Seguridad	10.1.1.15	Mem = 256 MB Storage = 40GB vCPU =2	Servidor de Ossec Ossec GUI Ossec analogi MySql Apache	Ubuntu 15.04	Es el servidor mas importante de Ossec, ya que este analiza los comportamientos de los agentes e inicia una acción de respuesta en caso de encontrar un comportamiento anómalo. Sobre este servidor también se configura la Interfaz web de Ossec y Analogi.
22	xampp_gs	Servidor	10.1.1.14	Mem=256 MB Storage = 40 GB vCPU = 2	PHP MySQL Apache	Ubuntu 15.04	Es el servidor donde ha sido configurado la base de datos que recibe las respuesta ejecutadas, y las presenta a través de un interfaz web. A su vez este servidor abarca un portal que incluye enlaces hacia los reportes de todas las detecciones.y respuestas

Tabla 4.1 Dispositivos implementados en el despliegue de la solución $(4/4)$

Para conseguir el escenario de red virtual con todas las máquinas previamente presentadas, se hace uso de dos tecnologías. La primera es VNX, la cual crea el escenario de red a través de un archivo de configuración XML. La segunda es KVM, la cual es utilizada para crear y administrar las máquinas virtuales. Cada una de estas máquinas virtuales es instanciada desde el archivo XML.

4.1.1.1 Escritura del escenario

En la Figura 4.2 se presenta el proceso que se debe llevar a cabo para la creación de un escenario de red virtual.

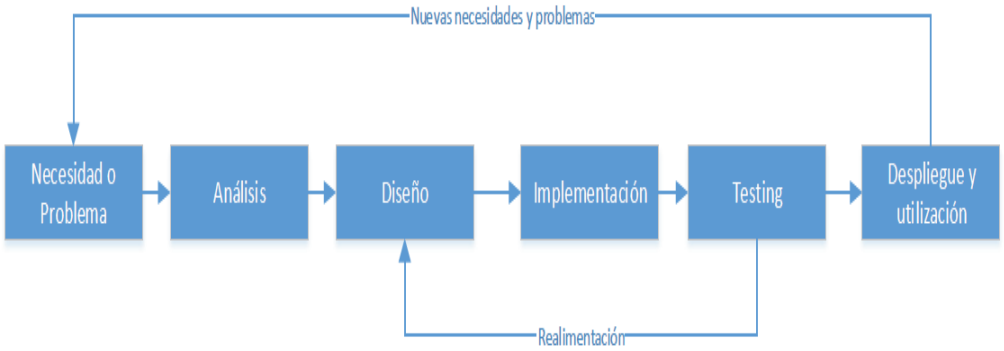


Figura 4.2 Ciclo de vida de desarrollo de un sistema [9].

Necesidad o problema: Creación de un escenario de red con topología jerárquica

Análisis: Cuántas y qué máquinas virtuales correrán sobre el escenario de red. Estos datos se obtienen de la Tabla 4.1.

Diseño el escenario: Basándose en el diseño presentado en el capítulo 3 y el lenguaje de referencia presentado en el capítulo 1, el escenario se encuentra constituido de las siguientes secciones: introducción, definiciones globales, redes virtuales, máquinas virtuales y configuraciones del host.

4.1.1.1.1Introducción o Cabecera

Conformada por la especificación del lenguaje XML, y XSD la cual es una descripción de la semántica de las etiquetas a utilizar, la misma que será especificada con el path absoluto, como se puede ver en la Figura 4.3:

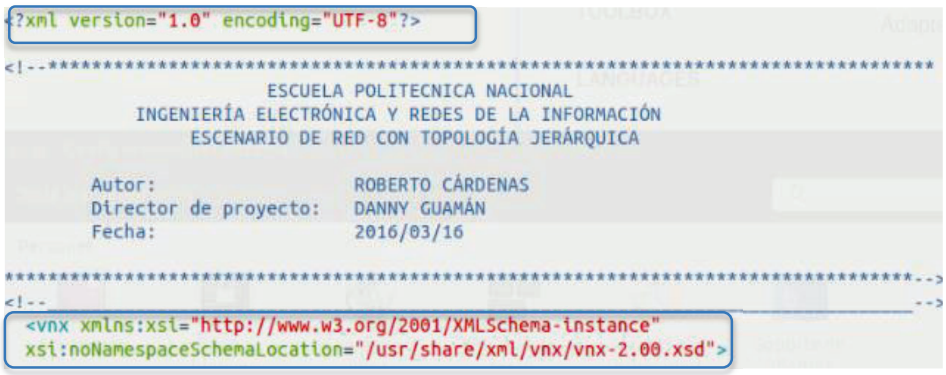


Figura 4.3 Cabecera archivo XML

4.1.1.1.2Definiciones Globales

En esta sección se define el nombre del escenario, las etiquetas que brindan funcionalidad a todos los elementos del escenario, y la especificación del archivo que contiene los dispositivos **Cisco** con los cuales se va a trabajar, la configuración del mismo es presentado en la Figura 4.4.

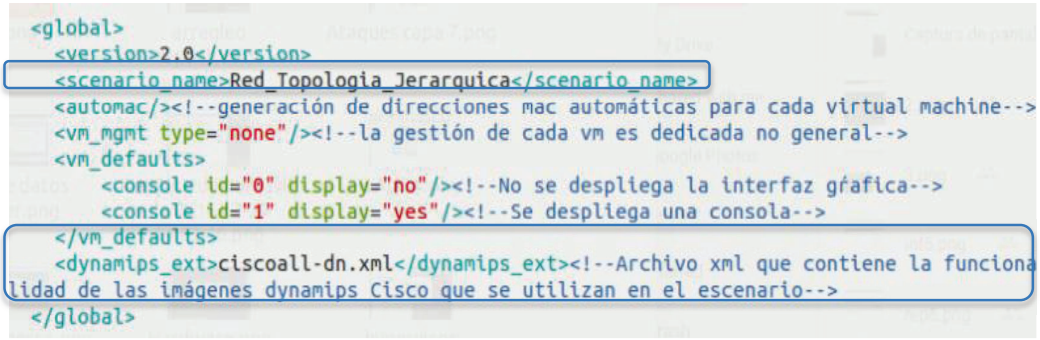


Figura 4.4 Definiciones globales

4.1.1.1.3Redes Virtuales

En la Figura 4.5 se presentan las redes que forman parte de la topología jerárquica utilizada en el proyecto.

Figura 4.5 Redes Virtuales

Cada red corresponde a un segmento de la topología jerárquica. El primer segmento es la red externa donde se encuentra situado el host desde el cual se ejecutarán ataques hacia la red interna. El segundo segmento corresponde a la red interna, dentro de la cual se despliegan todas las capas de la topología jerárquica. El tercer segmento corresponde a la conexión del host físico con el escenario de red. El cuarto y quinto segmento lo constituyen las VLANs configuradas dentro de la red interna. El sexto segmento es un enlace tipo puente con salida a internet. De esta manera se provee conectividad a todas las máquinas virtuales.

4.1.1.1.4 Máquinas Virtuales

En la Figura 4.6 se puede ver configurada una máquina virtual.

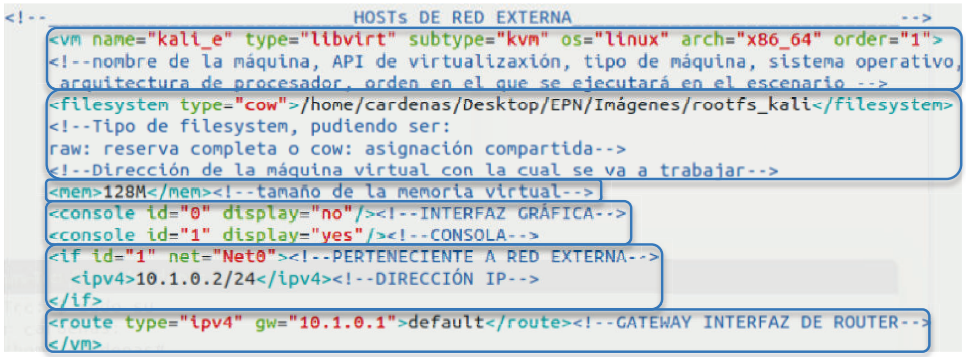


Figura 4.6 Configuración de máquina virtual en XML

En la figura 4.7 se muestra la configuración de una máquina virtual con imagen Dynamips. La configuración de estas máquinas muestra ciertas consideraciones, las mismas que se muestran en la figura 4.8 para un mayor entendimiento.

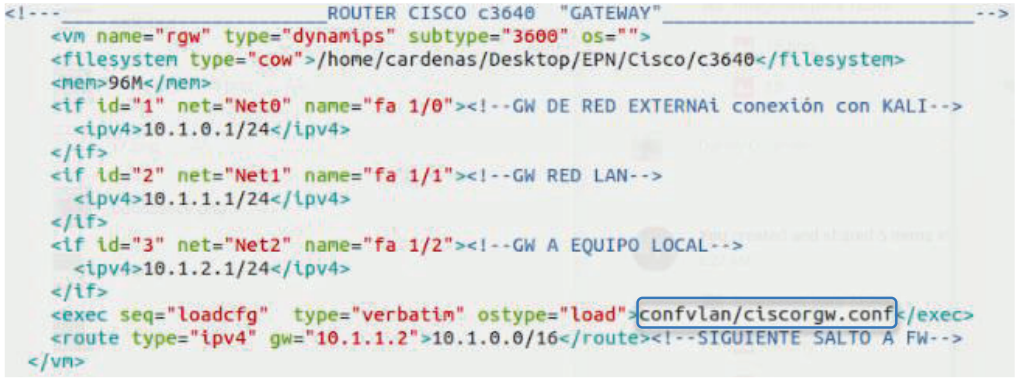


Figura 4.7 Configuración de máquina virtual con Dynamips en XML

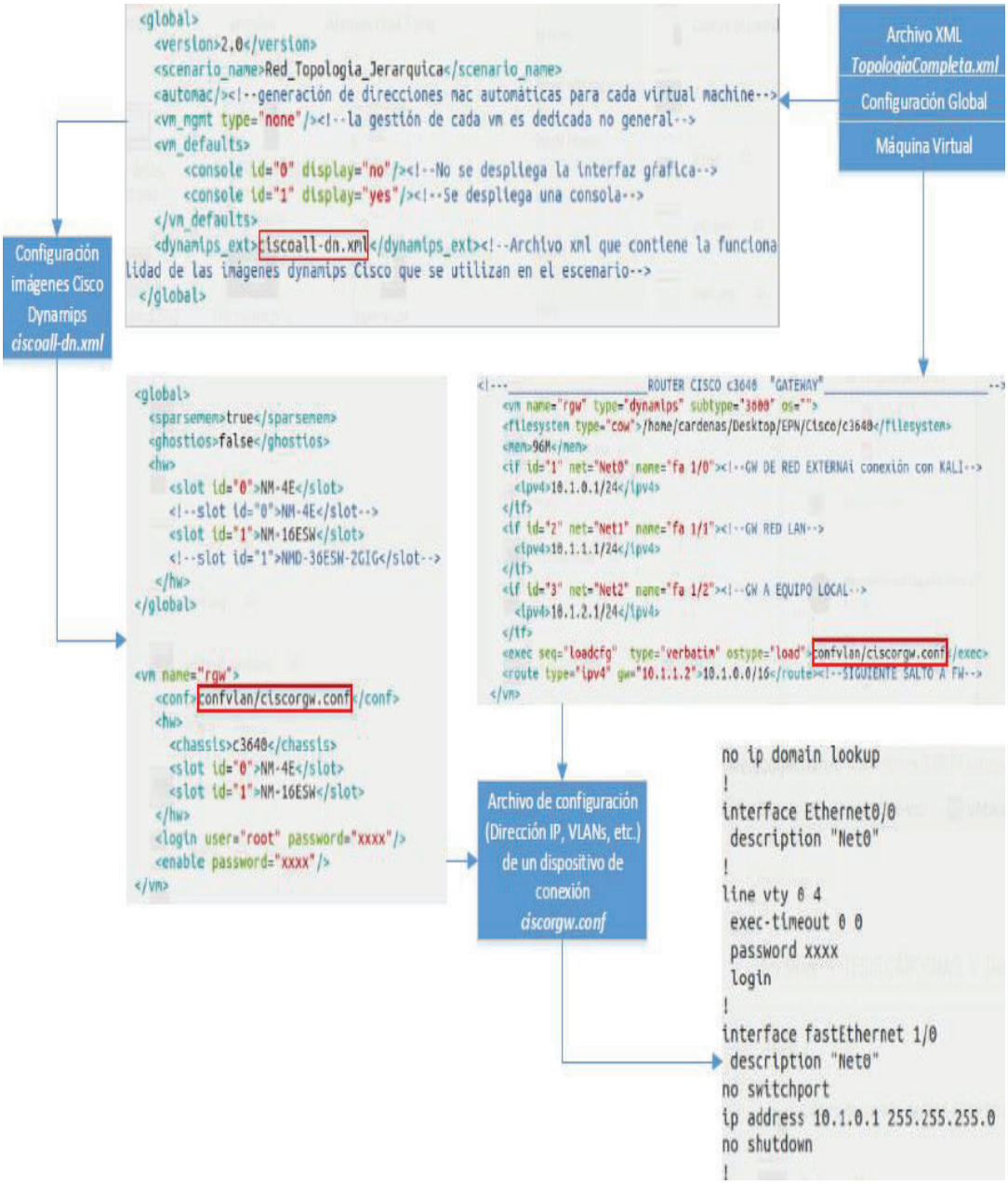


Figura 4.8 Configuración de imágenes Dynamips en archivos XML.

4.1.1.1.5Configuración del host

Esta sección configura una consola virtual, la cual permite acceder desde el Host físico hacia el escenario de red virtual, con la finalidad de interactuar con las máquinas virtuales. La configuración de esta consola es muy parecida a la configuración de una máquina virtual, como se puede ver en la figura 4.9. La configuración del escenario completo se encuentra en el Anexo 1.

```
<!--                              H-O-S-T                                    -->
  <host>
    <hostif net="Net2"><!--RED DE HOST FÍSICO-->
       <ipv4>10.1.2.2/24</ipv4>
    </hostif>
    <route type="ipv4" gw="10.1.2.1">10.1.0.0/16</route><!--GW INTERFAZ DE ROUTER-->
  </host>

</vnx>
```

Figura 4.9 Configuración de host..

Implementar el escenario: el escenario se implementa mediante el siguiente comando: *vnx –f Nombre_del_escenario.xml –v --create*

Probar el escenario (testing): Las principales pruebas que se realizan para observar el funcionamiento del escenario son:

- Pruebas de conectividad: Esto implica, revisar direcciones IP en máquinas virtuales, configuración de direccionamiento en equipos de conexión.
- Gestión de máquinas virtuales.
- Revisión de recursos de hardware físico.

Despliegue y utilización: Una vez que las pruebas del escenario han respondido correctamente, se procede a configurar las herramientas de seguridad. La configuración e integración de cada uno de los módulos se presenta en las siguientes secciones.

4.1.1.2 Generación de máquinas virtuales:

Los sistemas operativos implementados a través de root filesystem, utilizados para el despliegue de la solución de seguridad, son instanciados en la sección de máquinas virtuales del archivo de configuración XML, como se puede ver en la figura 4.10.

```
<filesystem type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_kali</filesystem>
<!--Tipo de filesystem, pudiendo ser:
raw: reserva completa o cow: asignación compartida-->
<!--Dirección de la máquina virtual con la cual se va a trabajar-->
```

Figura 4.10 Llamada a rootfile system desde el archivo de configuración XML.

VNX es una tecnología que permite generar root file system utilizando el siguiente proceso:

- Descargar una imagen iso del sistema operativo con el cual se desea trabajar, (pudiendo ser Windows o Linux ya que se emplea el mismo procedimiento).
- Crear la imagen ejecutando el siguiente comando: qemu-*img create -f qcow2 vnx_rootfs_kvm_ejemploWindows.qcow2 20G*
- Se crea el root file system utilizando la imagen iso previamente descargada, también se asignan los recursos básicos con los cuales se creará el root file system. Esto se lo lleva a cabo través de la ejecución de : *vnx --create-rootfs vnx_rootfs_kvm_wejemploWindows.qcow2 --install-media /root/imagenWindows.iso --mem 2G --arch x86_64*
- Una vez ejecutado el comando se inicia la instalación del sistema operativo, tal como se puede ver en la figura 4.11.
- Cuando la instalación termine de ejecutarse y se hayan configurado los parámetros importantes del sistema operativo, el siguiente paso es actualizar aced. La cual es una herramienta que permite acoplar el root file system creado con la tecnología VNX, de tal manera que se pueda optimizar su configuración y gestión.

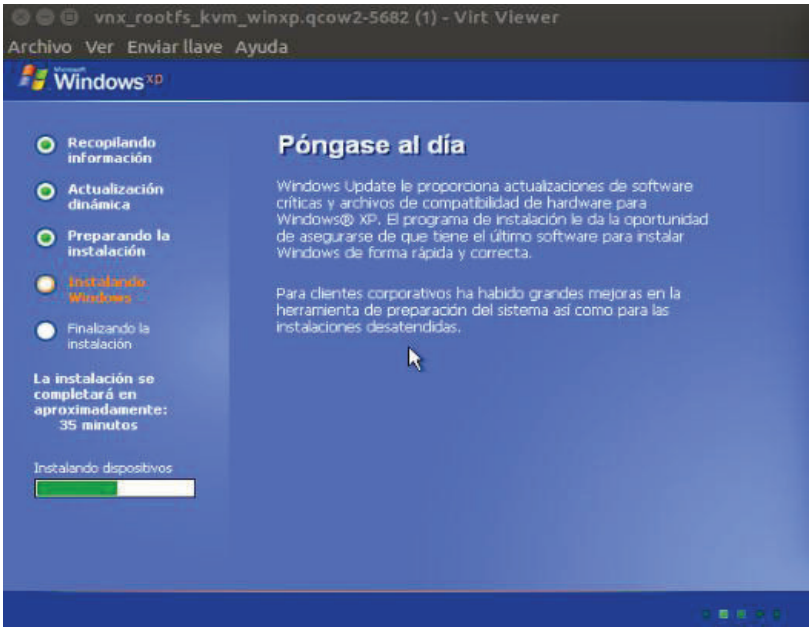


Figura 4.11 Instalación de Windows XP como rootfile system.

- Si se trabaja con una máquina Windows el comando que se deberá ejecutar es: /usr/share/vnx/bin/vnx_update_aced <vm_number> /usr/share/vnx/aced/vnx-aced-win-0.1b.exe
  Una vez que se descarga esta herramienta, se la ejecuta de manera normal.
- En caso de trabajar con una máquina Ubuntu el procedimiento que se deberá llevar a cabo para actualizar aced, es el siguiente:
  - Reiniciar el sistema operativo utilizan el siguiente comando: *vnx --modify-rootfs vnx_rootfs_kvm_ubuntu.qcow2* ***--update-aced*** *--mem 512M*
  - Se actualiza el sistema Linux: *apt-get update; apt-get dist-upgrade*
  - Se instalan dependencias de aced: *apt-get install libxml-libxml-perl libnetaddr-ip-perl acpid*
  - Se crea un punto de montaje para esta nueva actualización y se instala el paquete: *mount /dev/sdb /mnt/; perl /mnt/vnxaced-lf/install_vnxaced*

De esta manera se crea un root file system y se actualiza aced, optimizando la gestión y configuración del escenario virtual.

4.1.2 Configuración de IDS (Módulo 1 arquitectura)

En esta sección se describe la configuración y despliegue de la arquitectura sobre el escenario y máquinas virtuales creadas. Como primer punto se presentan los IDS, es decir el módulo 1 de la arquitectura propuesta.

4.1.2.1 Snort

Snort como se presentó en secciones anteriores, es un sniffer[36] de paquetes que monitorea y captura el tráfico que circula por la red con el objetivo de detectar intrusiones, aprovechando al máximo los recursos de procesamiento y minimizando la pérdida de paquetes. Por tal razón, la localización del mismo juega un papel importante en el diseño de la solución de seguridad.

36 Software que absorbe o captura paquetes de una red.

4.1.2.1.1 Localización

La localización se encuentra en función de la topología de red utilizada y las zonas que se pretende proteger. Como se revisó en el capítulo 1, existen tres zonas en las que se recomienda colocar los IDS. Es posible implementar esta recomendación basándose en la arquitectura o estrategia de control distribuida con la que cuenta Snort, de esta manera la localización de los mismos es la siguiente. Cada una de las zonas, requiere una configuración de Snort diferente.

- **Zona Roja**: La localización de Snort en esta zona, permitirá detectar ataques que vienen desde el exterior de la red.

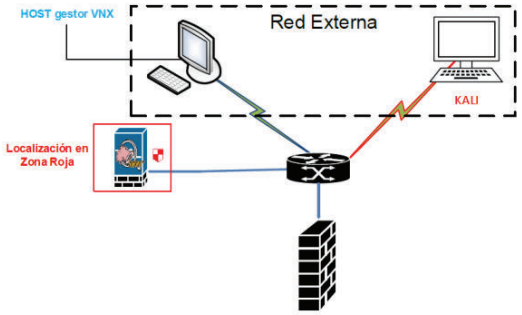


Figura 4.12 Localización Snort en zona roja.

- **Zona Verde**: La localización de Snort en esta zona, se encuentra adecuada para proteger los servidores que aquí se encuentren, es decir que las reglas con las que se configure el IDS deberán estar ser diseñadas en función de los servicios provistos por esta zona. En la Figura 4.14 se puede ver la localización de Snort para esta zona.

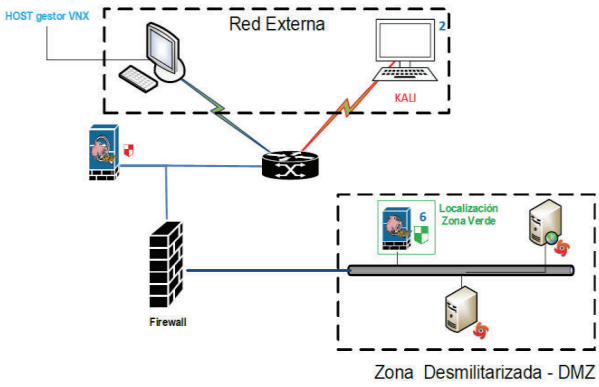


Figura 4.13 Localización de Snort en zona verde.

- **Zona azul**: La localización de Snort en esta zona tiene la finalidad de analizar los paquetes que ingresen a la granja de servidores, y generar alertas en el caso de detectar un comportamiento anómalo o intentos de intrusión, entendiendo como intrusión a aquel flujo de tráfico o paquetes que intentan poner en riesgo la integridad, disponibilidad y confidencialidad de las máquinas que se encuentren en esta zona. En la Figura 4.14 se puede observar la localización de Snort en la zona azul.

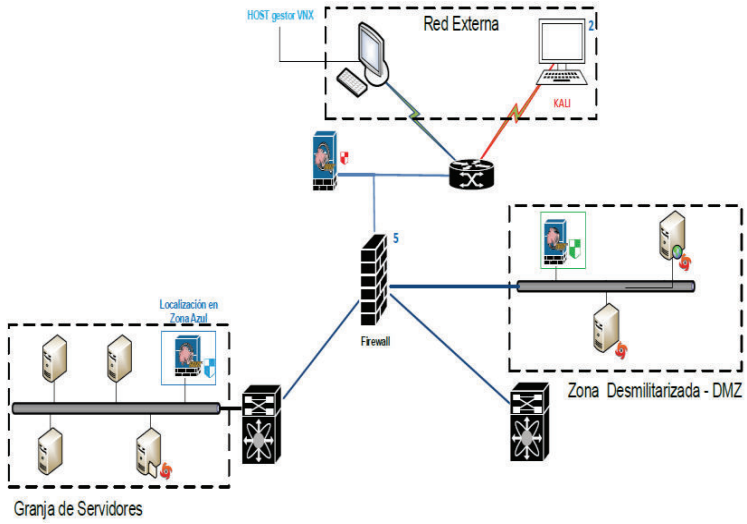


Figura 4.14 Localización de Snort en zona azul.

4.1.2.1.2 Configuración

Para el uso correcto de Snort se deben conocer y considerar los siguientes aspectos. Snort tiene 4 modos de ejecución: modo de sniffer, modo de Packet logger, modo NIDS y modo in-line, cada uno de ellos se los podrá habilitar en línea de comandos tras ejecutar las opciones que se presentan el la Tabla 4.2. Para el presente proyecto se seleccionan dos manera de Snort, la primera será en modo snifer, de tal manera que se podrá analizar el tráfico que pase a través de Snorty la segunda es en modo NIDS.

Snort adquiere el comportamiento de un IDS basado en red integrando algunos archivos y directorios con permisos adecuados. Esto se consigue siguiendo el procedimiento presentado en la tabla 4.3.

Comando	Opción	Modo de ejecución	Descripción
snort	-v	Sniffer	Inicia a Snort en modo sniffer de paquetes, mostrando en pantalla la dirección IP y cabeceras tcp/udp/icmp.
snort	-l	Parcket logger	Especifica el directorio donde almacenará los logs generados.
snort	-c	NIDS	Especifica el archivo de configuración *snort.conf* con la configuración necesaria para que Snort adquiera las funcionalidades de NIDS
snort		Inline	Personalización en las reglas que utilice snort

Tabla 4.2 Modos de ejecución de Snort

Snort NIDS
Crear un usuario y grupo snort
groupadd snort sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
Crear subdirectorios
sudo mkdir /etc/snort sudo mkdir /etc/snort/rules sudo mkdir /etc/snort/rules/iplists sudo mkdir /etc/snort/preproc_rules sudo mkdir /usr/local/lib/snort_dynamicrules sudo mkdir /etc/snort/so_rules
Crear archivos para almacenar reglas y listas de IPS
sudo touch /etc/snort/rules/iplists/black_list.rules sudo touch /etc/snort/rules/iplists/white_list.rules sudo touch /etc/snort/rules/local.rules sudo touch /etc/snort/sid-msg.map

Tabla 4.3 Configuración de Snort en modo NIDS (1/2).

Snort NIDS
Crear archivos para logs
sudo mkdir /var/log/snort
sudo mkdir /var/log/snort/archived_logs
Otorgar permisos
sudo chmod -R 5775 /etc/snort
sudo chmod -R 5775 /var/log/snort
sudo chmod -R 5775 /etc/snort/so_rules
sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
Cambiar el propietario de las carpetas
sudo chown -R snort:snort /etc/snort
sudo chown -R snort:snort /var/log/snort
sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules

Tabla 4.4 Configuración de Snort en modo NIDS (2/2).

Cada uno de los directorios y archivos creados con sus respectivos permisos, permiten que Snort trabaje integralmente analizando la red en busca de intrusiones. Para iniciar Snort en modo NIDS se debe configurar el archivo principal de snort "snort.conf" desde el cual se integran los demás archivos, como reglas, listas blancas, threshold, etc. Las modificaciones que se deben realizar en el documento *snort.conf* son:

- Configuración de la dirección de red y direcciones a las cuales se va a proteger.

```
# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.122.0/24
```

Figura 4.15 Configuración de red que protegerá Snort.

- Configuración de red externa: como se puede ver en la Figura 4.16, se considera como red externa, a todas las direcciones que no están dentro de !$HOME_NET, es decir de la red interna.

```
#ipvar EXTERNAL_NET any
ipvar EXTERNAL_NET !$HOME_NET
```

Figura 4.16 Configuración de red externa.

- La creación de variables para los servidores se presenta en la Figura 4.17. Las variables serán utilizadas en la creación y configuración de las reglas.

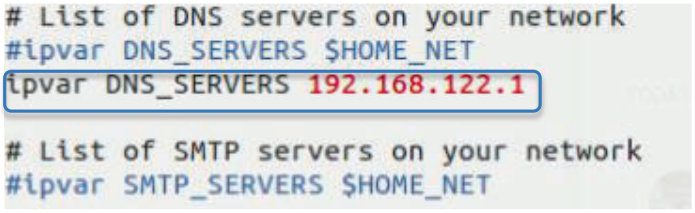


Figura 4.17 Configuración de variables para direcciones de Servidores.

- Descripción de la ruta donde se encuentra las reglas que se requiere implementar, dentro de cada archivo .rules se configuran las reglas personalidas para un ataque.

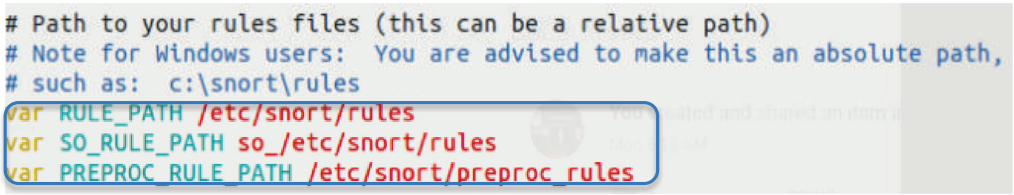


Figura 4.18 Rutas absolutas de reglas de Snort.

- Incluir las reglas a utilizar. A pesar de que previamente se haya configurado el path del directorio donde se encuentran las reglas, estas deben ser incluidas para que el motor de análisis de Snort haga uso de ellas para detectar intrusiones.

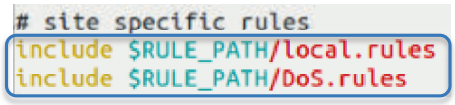


Figura 4.19 Incluir reglas dentro de snort.conf.

4.1.2.1.3 Configuración de Reglas

El motor de detección que utiliza Snort se encuentra basado en reglas, las cuales son patrones que se buscan dentro de los paquetes que Snort monitorea, y genera una alerta en caso de encontrar una similitud. La estructura de una regla de Snort se la puede ver en la Figura 4.20.

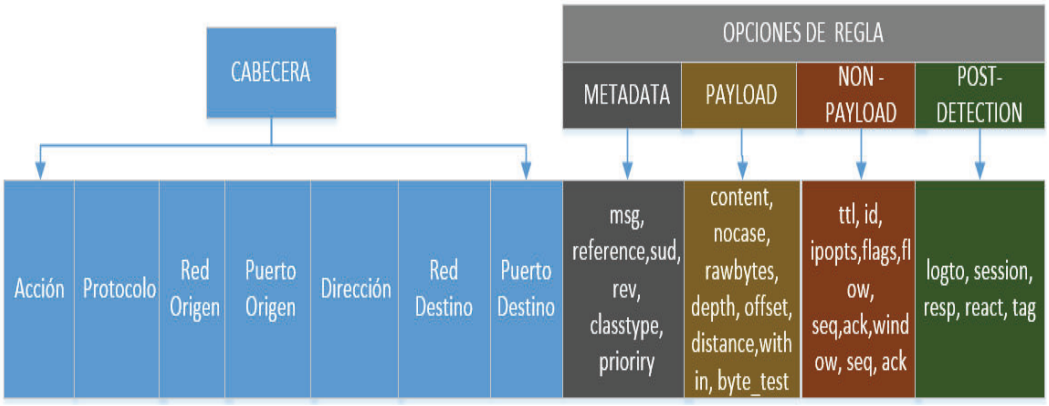


Figura 4.20 Estructura de regla de Snort [92]

La cabecera se encuentra conformada por los siguientes campos:

CABECERA						
Acción	Protocolo	Red origen	Puerto origen	Dirección	Red destino	Puerto destino
alert log Pass drop reject	tcp Udp Icmp Ip	$EXTERNAL_NET	Any	->	$HOME_NET	Any

Tabla 4.5 Cabecera de regla de Snort.

La construcción de la cabecera de una regla, consiste en una combinación de acciones, protocolos, direcciones ip y puertos, adaptables para el diseño de una solución de seguridad. Los siguientes campos de una regla de snort se componen por las opciones de una regla, las cuales se especifican en la tabla 4.5.

Cada regla de Snort depende especifícamente del ataque que se requiera detectar, es decir, en caso de detectar un ataque DoS y dependiendo de la naturaleza del mismo se creará una regla que analice los los portocoles que usualmente utiliza este ataque como, TCP o UDP. Pero en caso de detectar un malware o ataques que lleven en su payload código malicioso, Snort trabajará de manera distinta.

Opciones de una regla	Característica
Metadata	Son campos netamente informativos, es decir que no aportan para la detección de una intrusión.
Payload	Busca patrones o firmas dentro del payload de un paquete
Non-payload	Buscan patrones en los demás campos que no sean payload de un paquete, por ejemplo cabecera.
Post Dtection	Activa reglas específicas, que ocurren después de que se ejecute una regla.

Tabla 4.6 Opciones de reglas de Snort. [93]

Ejemplo de regla de Snort

Sw requiere crear una regla que detecte un ataque DoS basado en la capa de aplicación. El ataque seleccionado es *Slowloris*, el cual es un ataque escrito en perl que aprovecha una vulnerabilidad del protocolo http para provocar una denegación de servicio agotando los recursos disponibles. Como se puede ver en la Figura 4.22, dentro del archivo que contiene el código del ataque, existe un método que intenta establecer comunicación con un servidor web:

```
my $primarypayload =
    "GET /$rand HTTP/1.1\r\n"
  . "Host: $sendhost\r\n"
  . "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR 1.
1.4322; .NET CLR 2.0.50313; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; MSOffice 12)\r\n"
  . "Content-Length: 42\r\n";
if ( print $sock $primarypayload ) {
    print "Connection successful, now comes the waiting game...\n";
}
else {
    print
"That's odd - I connected but couldn't send the data to $host:$port.\n";
    print "Is something wrong?\nDying.\n";
    exit;
}
}
```

Figura 4.21 Código fuente ataque slowloris

Sabiendo lo que el paquete trae en su payload, se escribirá una regla que analice los paquetes de red y detecte este patrón, a su vez, genere una alerta de seguridad.

La regla escrita para este ataque es:

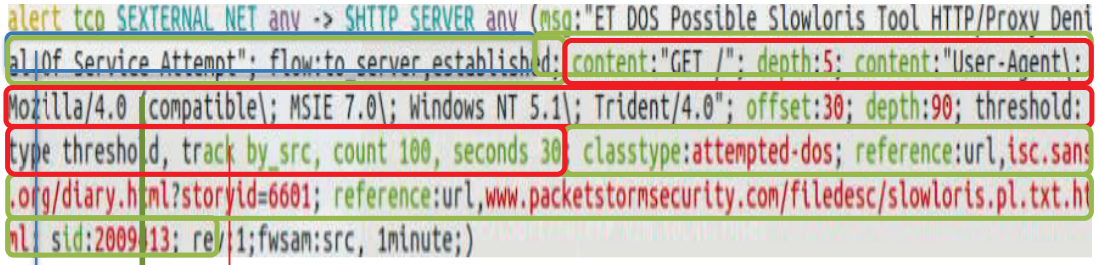


Figura 4.22 Análisis de Ataque slowloris

Payload - Content: Snort buscará esta secuencia dentro del payload paquete.
Depth: El contenido: "GET/ "debe encontrarse en los primeros 5 bytes del payload.
Offset: El contenido: "User-Agent\: Mozilla/4.0(compatible; MSIE 7.0\; Windows NT)" debe estar a partir de los primeros 30 bytes de payload.

Metadato, contiene solo la parte informativa del ataque DoS, en este caso se trata de una mensaje que se mostrará cuando se detecte un paquete que coincida con esta firma.
Classtype: qué tipo de ataque intentó el paquete.
Reference: permite referenciar las reglas a través de formatos predefinidos.

Cabecera de la regla- la cual se encuentra conformada por: el protocolo tcp, tipo de alerta, direcciones IP y puertos desde los cuales se espera los paquetes, y las direcciones IP y puertos que se va a proteger.

Figura 4.23 Regla para detectar ataque slowloris

4.1.2.2 Ossec

Ossec es un IDS basado en host, cuyo funcionamiento se basa en el análisis de la integridad de un sistema operativo, ya sea este Linux o Windows.

4.1.2.2.1 Localización

Ossec es una herramienta que basa su funcionamiento en un formato cliente servidor. El servidor o servidores, lo conforman aquellas máquinas en las cuales se haya instalado el agente de Ossec.

Los agentes de Ossec son instaladas en máquinas que se requiera un monitoreo del sistema operativo, por lo tanto la localización de los agentes es independiente de la topología de red. Para el proyecto, se identifica las maquinas que cuenten con el agente de Ossec, a través del símbolo , tal como se puede observar en la Figura 4.24.

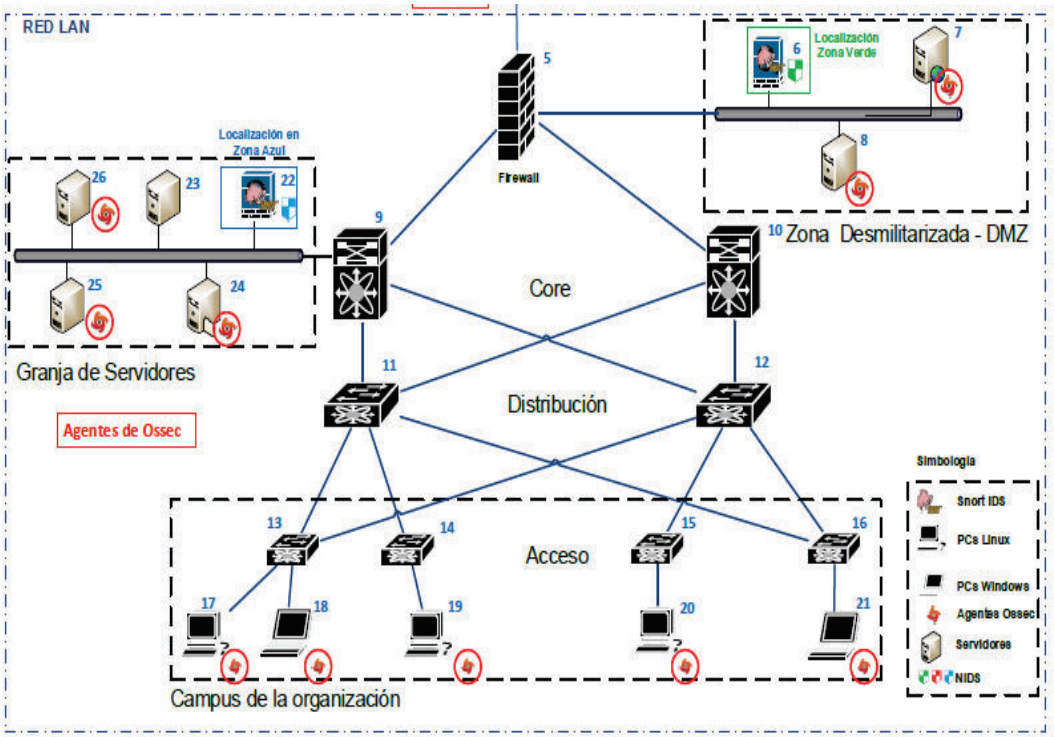


Figura 4.24 Agentes de Ossec desplegados sobre topología de red jerárquica.

4.1.2.2.2 Configuración

La configuración ya sea de un agente o un servidor, se lo realiza al momento de instalar Ossec. Al momento de ejecutar el instalador, la primera pregunta que Ossec presenta es: ¿Qué tipo de instalación desea? Tal como se aprecia en la Figura 4.25

```
  - System: Linux vps1.sparklingclouds.nl 3.2.0-042stab076.8
  - User: root
  - Host: vps1.sparklingclouds.nl
  -- Press ENTER to continue or Ctrl-C to abort. --
1- What kind of installation do you want (server, agent, local, hybrid or help)? server
```

Figura 4.25 Instalación de Ossec.

Por lo general se escoge cliente o servidor, y dependiendo de esta selección, se puede tener las opciones que se presenta a continuación en la tabla 4.6.

Servidor	Agente
Configurando las variables de entono: Elija donde instalar OSSEC HIDS [/var/ossec]	Configurando las variables de entono: Elija donde instalar OSSEC HIDS [/var/ossec]
Configurando el sistema OSSEC HIDS: Desea recibir notificación por correo electrónico Desea agregar el servidor de integridad del sistema Desea agregar detección de rootkit ***¿Desea Habilitar respuesta activa?*** Desea habilitar syslog remoto	Configurando el sistema OSSEC HIDS: Dirección o nombre del servidor Desea agregar el servidor de integridad del sistema Desea agregar detección de rootkit ***¿Desea Habilitar respuesta activa?***

Tabla 4.7 Opciones de reglas de instalación de Ossec.

Una vez que se ha instalado el agente y el servidor, el siguiente paso será integrarlos, realizando el siguiente procedimiento.

a) Ejecutar en el servidor: /var/ossec/bin/manage_agents (donde aparecerá las opciones presentadas en la Figura 4.26. Se selecciona (A)

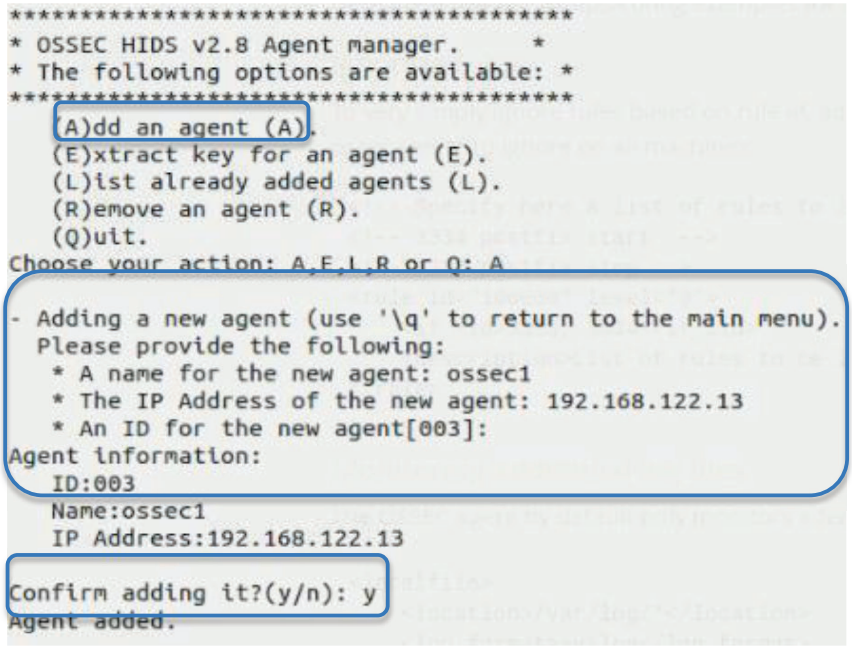


Figura 4.26 Añadir un agente en Ossec server.

b) Extraer clave para el agente: Dentro del mismo menú ahora se selecciona (E) como se puede ver en la Figura 4.27

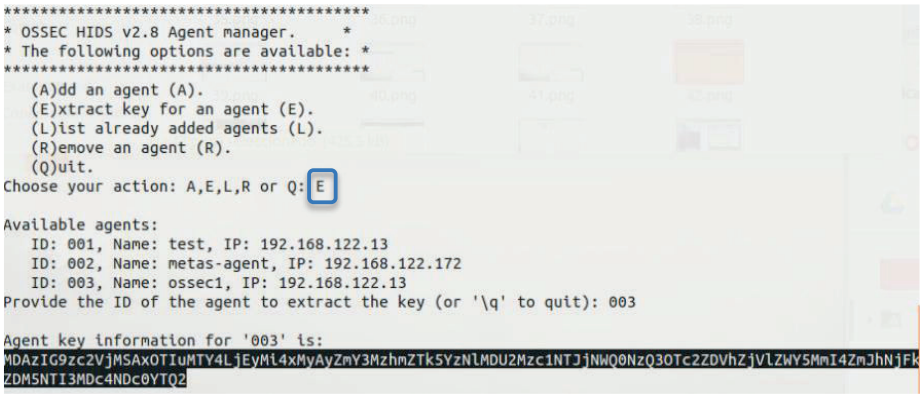


Figura 4.27 Extraer clave de comunicación agente – servidor en Ossec

c) Importar Clave desde servidor: La siguiente configuración se lleva a cabo en la máquina que se ha instalado como agente de Ossec, ejecutando /var/ossec/bin/manage_agents. Para establecer una comunicación encriptada, el servidor Ossec genera una clave, la cual fue generada en el punto anterior, y deberá ser copiada como se puede ver en la figura 4.28.

Figura 4.28 Copiar la clave de comunicación en el agente de Ossec

d) Verificar la dirección del servidor Ossec: Cambiar o validar la dirección del servidor de Ossec en los archivos de configuración del agente de Ossec, de la forma como se presenta la Figura 4.29, el cual se encuentra en /var/ossec/etc/ossec.conf

```
<ossec_config>
  <client>
    <server-ip>192.168.122.246</server-ip>
  </client>
```

Figura 4.29 Dirección de servidor de Ossec en archivos de configuración de agente Ossec

e) Iniciar Control de Ossec: Tanto en el agente como en el servidor de debe iniciar el control de Ossec a través del comando /var/ossec/bin/Ossec-control start. Se inicia una revisión de los archivos de configuración.

4.1.3 Configuración de respuesta activa (Módulos 2 y 3 de la arquitectura)

4.1.3.1 Snortsam

Es una herramienta que consta de un agente que corre sobre un Iptables o ciertos firewalls comerciales y un parche implementado sobre Snort, basados en una arquitectura cliente servidor, nos permiten contar con un NIDS con respuesta activa tal como se puede ver en la figura 4.30.

Figura 4.30 Arquitectura cliente-servidor Snortsam

Es importante manejar bien la compatibilidad de versiones entre Snort, Snortsam (agente) y parche de Snortsam, no necesariamente las más actualizadas son compatibles entre sí. Las utilizadas para este proyecto son:

Herramienta	Versión	Link de descarga
Snort	2.9.7.3	https://www.snort.org/
Parche de Snortsam	snortsam-2.6.1.3.diff	ftp://ftp.snortsam.net/
Snortsam Agente	Snortsam-src-2.60	http://www.snortsam.net/files/snortsam/

Tabla 4.8 Versiones de herramientas Snort y Snortsam.

Para que un despliegue de arquitectura cliente servidor es ejecute de manera efectiva utilizando Snortsam se debe realizar el siguiente procedimiento:

a) Instalación parche Snortsam en Snort:

Descargar el paquete: snortsam-2.6.1.3.diff

Aplicar el parche sobre snort: patch –l < snortsam-2.6.1.3.diff

Al aplicar el parche, se obtiene la siguiente salida:

patching file autojunk.sh

patching file src/Makefile.am

patching file src/output-plugins/Makefile.am

patching file src/output-plugins/spo_alert_fwsam.c

patching file src/plugbase.c

patching file src/plugin_enum.h

patching file src/twofish.c

patching file src/twofish.h

Despues de haberse modificado el archivo principal de Snort se dan permisos al archivo de ejecución: chmod +x autojunk.sh

Se compila y ejecuta: ./configure –with-mysql; make; make install

b) Instalación de agente Snortsam

Descarga del paquete: snortsam-src-2.60.tar.gz

Descomprimir el paquete: tar xvzf snortsam-src-2.60

Otorgar permisos: chmod 777 makesnortsam.sh

Copiar los ejecutables al directorio: /usr/local/bin: cp

/usr/local/src/snortsam/snortsam* /usr/local/bin

/usr/local/src/snortsam/conf/snortsam.conf.sample /etc/

Renombrar el archivo de configuración: snortsam.conf

c) Configuración agente Snortsam

Para que Snort pueda enviar una petición de bloqueo, se requiere que un agente se encuentre escuchando por un puerto específico en el firewall, el puerto por defecto es 898. Para usar una comunicación encriptada se requiere configurar una clave, tanto la clave como el puerto a utilizar se presentan en la figura 4.31.

```
#   Set's the default key for ALL allowed hosts to <key>.
#   The default key is used when no other key is specified in an ACCEPT option.
#   You have to use the same key in the snort.conf file in the
#   "output alert_fwsam line". If the keys, or passwords if you will, don't
#   match, SnortSam can not decrypt the request from Snort and ignore it.
#
#   Example:    defaultkey mydefaultpassword
defaultkey password
#   If omitted, SnortSam will use a default key (in which case it would have to
#   omitted in snort.conf as well).
#
# port <port>
port 898
```

Figura 4.31 Configuración de puerto y clave de Snortsam

Lo siguiente que se debe configurar es: una lista de sensores Snort desde los cuales se aceptarán las peticiones de bloqueo, opcionalmente se puede adicionar una clave de encripción propia para un host, tal como se puede ver en la figura 4.32.

```
#
accept 192.168.122.0/24, password
# accept <host>/<mask>,<key>
#
#   This option lists Snort sensors that SnortSam is accepting packets from.
#   You can specify hostname, IP address, IP address and network mask, and
#   optionally an encryption key used configured for that host or network.
```

Figura 4.32 Sensores Snort aceptados en Snortsam

Snortsam permite la generación de listas blancas, dentro de las cuales se añade host o redes que no serán bloqueadas, la configuración de listas blancas se presenta en la Figura 4.33

```
# dontblock <host>/<mask>
#
#   This adds the host or network to the white-list of hosts/networks that will
#   never be blocked. Blocking request for hosts on this list are ignored.
#
#   Examples:  dontblock a.root-servers.net
#              dontblock 192.168.10.0/24
```

Figura 4.33 Listas blancas en Snortsam

Snortsam permite especificar el archivo en el cual se crearán logs de los eventos de Snortsam, cada vez que se inicie el programa. Las acciones que se almacenan son: bloqueos, desbloqueos, acciones de rendimiento y eventos de error.

La configuración de la ruta de este archivo de logs, se presenta en la figura 4.34

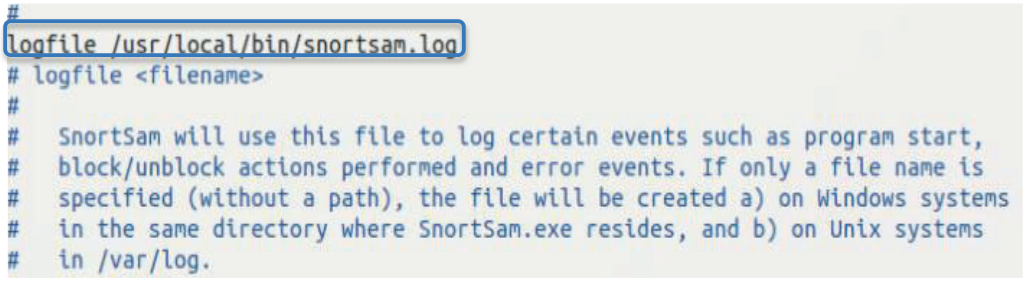


Figura 4.34 Localización de archivo de logs

Snortsam puede utilizar diferentes mecanismos para iniciar una respuesta, cada uno de ellos requiere de la configuración de un plugin. Los plugins con los que cuenta Snortsam son: wexec, fwsam, fwsamipflip, opsec, plugin pix, cisco acl, cisconullroute, email, Iptables, forward. El plugin implementado en el proyecto es Iptables, la configuración del mismo consta del nombre del plugin y la interfaz de red por la cual recibe los paquetes, tal como se puede ver en la Figura 4.35.

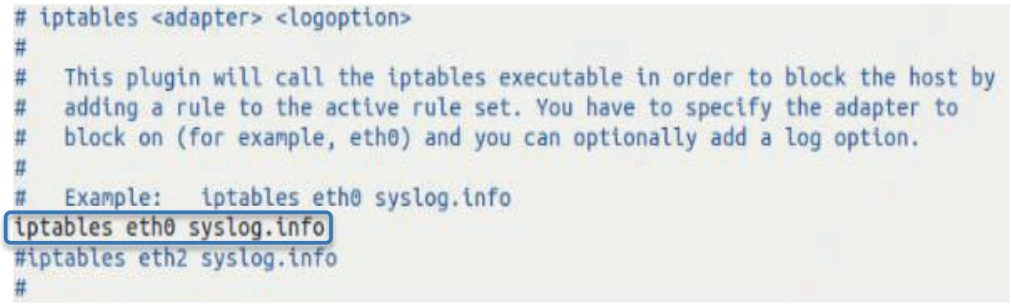


Figura 4.35 Implementación de plugin Iptables en Snortsam

d) Configuración Snort

Una vez que se ejecuta el parche de Snortsam, y este ha modificado el código fuente de Snort, se puede configurar el archivo snort.conf , como se muestra en la Figura 4.36

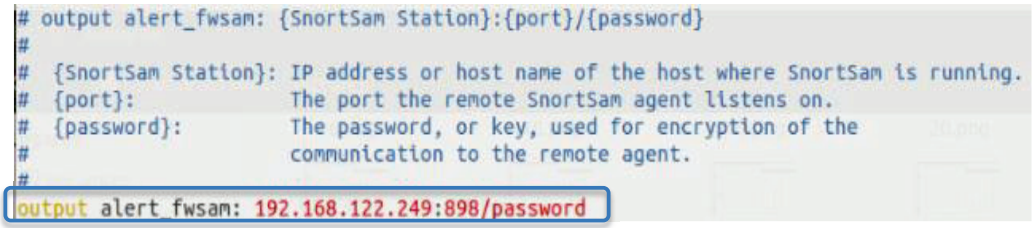


Figura 4.36 Configuración de conexión Snortsam en snort.conf

e) Configuración de reglas con Snortsam

Toda regla que requiera iniciar una respuesta activa al coincidir con su firma, deberá agregar al final de la regla los siguientes parámetros, tal como se muestra en la Figura 4.37.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVER any (msg:"ET DOS Possible Slowloris Tool HTTP/Proxy Deni
al Of Service Attempt"; flow:to_server,established; content:"GET /"; depth:5; content:"User-Agent\:
Mozilla/4.0 (compatible\; MSIE 7.0\; Windows NT 5.1\; Trident/4.0"; offset:30; depth:90; threshold:
type threshold, track by_src, count 100, seconds 30; classtype:attempted-dos; reference:url,isc.sans
.org/diary.html?storyid=6601; reference:url,www.packetstormsecurity.com/filedesc/slowloris.pl.txt.ht
ml; sid:2009413; rev:1;fwsam:src, 1minute;)
```

Figura 4.37 Reglas con plugin Snortsam

4.1.3.2 Agente - Servidor Ossec

Los agentes de Ossec se encuentran monitoreando el sistema operativo, generando eventos de cualquier comportamiento, los mismo que son enviados hacia el servidor Ossec, el cual analiza los eventos y genera alertas ante un posible comportamiento anómalo, las cuales permiten generar patrones de comportamiento. En la Figura 4.38 se presenta gráficamente el funcionamiento del modelo agente servidor de Ossec.

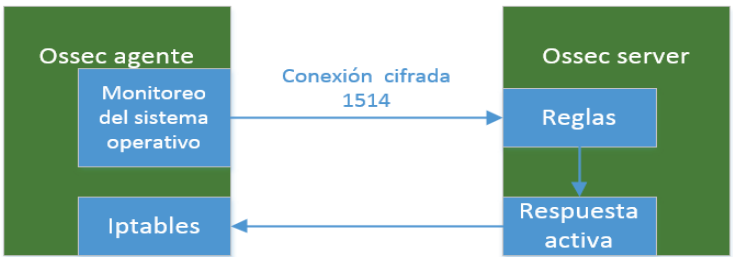


Figura 4.38 Modelo agente servidor Ossec

La configuración de la respuesta activa de Ossec se la lleva a cabo en el archivo de configuración "*ossec.conf*", localizado en el servidor de Ossec. Este archivo básicamente consta de las siguientes secciones:

- Reglas: Hace referencia a archivos xml, en los cuales se ha definido reglas, las mismas que representan un comportamiento específico en un host, y han sido agrupadas de acuerdo a su afinidad, como se lo puede ver en la Figura 4.39.

- Syschek: Ejecutado bajo demanda, permitiendo ignorar directorios que se consideren no deban ser parte del chequeo.
- Rootcheck: Análisis que requiere privilegios de administrador para su ejecución.

```
<group name="apache,">
  <rule id="30100" level="0">
    <decoded_as>apache-errorlog</decoded_as>
    <description>Apache messages grouped.</description>
  </rule>

  <rule id="30101" level="0">
    <if_sid>30100</if_sid>
    <match>^[error] </match>
    <description>Apache error messages grouped.</description>
  </rule>

  <rule id="30102" level="0">
    <if_sid>30100</if_sid>
    <match>^[warn] </match>
    <description>Apache warn messages grouped.</description>
  </rule>
```

Figura 4.39 Archivo xml para reglas apache de Ossec

- Comandos: Scripts definidos por Ossec, los cuales llevarán a cabo una acción de respuesta activa. Ossec provee de los siguiente comandos
  - Host-deny
  - Firewall-drop
  - Disable –account
  - Restart Ossec
  - Route-null
- Repuesta activa: Compuesta por los siguientes campos:
  - Comando: Script a utilizar
  - Localización: Donde se ejecutará el comando, pudiendo ser local o remota.
  - Reglas: ID de la regla utilizada para detectar un comportamiento anómalo. El conjunto de reglas provee patrones de comportamiento.
  - Timeout: tiempo de duración del comando ejecutado.
- Localfile: Path del directorio donde serán almacenados los logs.

A continuación en la Figura 4.40 se presenta un ejemplo de configuración de una respuesta activa en *ossec.conf.*

```
<active-response>
  <!-- Firewall Drop response. Block the IP for
    - 600 seconds on the firewall (iptables,
    - ipfilter, etc).
   -->
  <command>firewall-drop</command>
  <location>all</location>
  <rules_group>ids, </rules_group>
  <rules_id>100002, 100003, 100004, 100031, 5503, 5716, 1002, 2502, 5716, 5503, 131105, 131105,900
000, 900001, 900002, 900003, 900010, 900011, 900012, 900013, 31151  </rules_id>
  <level>2</level>
  <timeout>60</timeout>
</active-response>
```

Figura 4.40 Configuración respuesta activa Ossec

4.1.4 Herramientas GUI (Módulo 4 arquitectura)

4.1.4.1 BASE como Frot-end

Es una herramienta que permite recolectar las intrusiones detectadas por Snort y presentarlas a través de una interfaz web, para lo cual requiere la instalación y configuración de un servidor web que soporte PHP y una base de datos. El procedimiento que se utilizó para el despliegue de BASE, se debe seguir el siguiente procedimiento.

a) Plataforma XAMPP

Para la implementación de BASE se ha seleccionado la herramienta XAMPP (versión actualizada de LAMP (Linux- apache-mysql-php)), la cual es una versión de apache que ya viene integrada con MySQL y lenguajes de programación Perl y PHP. La configuración de XAMPP básicamente consiste en seguir un wizard el cual configura cada una de las herramientas. Una vez que XAMPP se haya instalado correctamente se inicializan los servicios ejecutando el comando:
/opt/lampp/lampp/ start

b) Barnyard2

Se hace uso de esta herramienta como medio para optimizar la capacidad de almacenamiento de alertas generadas por Snort en una base de datos. Barnyard2

es un intérprete de ficheros de salida en formato unified2, el cual permite la escritura de las alertas generadas por Snort de manera más rápida en un base de datos. La configuración de la salida de alertas de Snort se lo realiza en el archivo *snort.conf* tal como se muestra en la figura 4.41

```
# unified2
# Recommended for most installs
# output unified2: filename merged.log, limit 128, nostamp, mpls_event_types,
output unified2: filename snort.u2, limit 128
#vlan_event_types, mpls_event_types

# Additional configuration for specific types of installs
output alert_unified2: filename snort.alert, limit 128, nostamp
output log_unified2: filename snort.log, limit 128, nostamp

# syslog
output alert_syslog: LOG_AUTH LOG_ALERT
```

Figura 4.41 Configuración de formato de salida de alertas de Snort

c) Base de datos Snort

Creación de base de datos para almacenar los eventos generados por Snort con la ayuda de barnyard2.

mysql> create database snort; use snort;

mysql> source ~/snort_src/barnyard2-2-1.14-336/schemas/create_mysql;

mysql> CREATE USER‘snor’ @‘localhost’ IDENTIFIED BY ‘password’;

mysql> grant create, insert, select, delete, update on snort.* to ‘snort@’localhost’;

mysql> exit;

El siguiente paso será, conectar Barnyard2 a la base de datos creada, esto es posible configurando el archivo *barnyard2.conf* como se muestra en la Figura 4.42

```
output database: log, mysql, user=root password=snortreport dbname=snort host=localhost
```

Figura 4.42 Conexión de barnyard2 a la base de datos de Snort

d) Configuración BASE (Basic Analysis and Security Engine

Requiere de 4 paquetes de instalación ñps cuales se presentan junto a sus versiones en la tabla 4.8:

Herramienta	Versión	Link de descarga
Barnyard2	barnyar2-2.1.13	https://github.com/firnsy/barnyard2
Adodb	Adodb462	http://adodb.org/dokuwiki/doku.php
Image_graph	Image_Color-1.0.2. Image_Canvas-0.3.0 Image_Graph-0.7.2.tgz	http://pear.php.net/package/pearweb_gopear

Tabla 4.9 Versiones de herramientas requeridas por BASE.

Una vez que se han instalado las herramientas se configura BASE en el archivo de configuración: *base_conf.php,* como se puede ver en la Figura 4.43

```
/* Archive DB connection parameters */
$archive_exists   = 0; # Set this to 1 if you have an archive DB
$archive_dbname   = 'snort_archive';
$archive_host     = 'localhost';
$archive_port     = '';
$archive_user     = 'snort';
$archive_password = 'snortreport';
```

Figura 4.43 Configuración de BASE

La apariencia de la interfaz web final de BASE se la presenta en la Figura 4.44

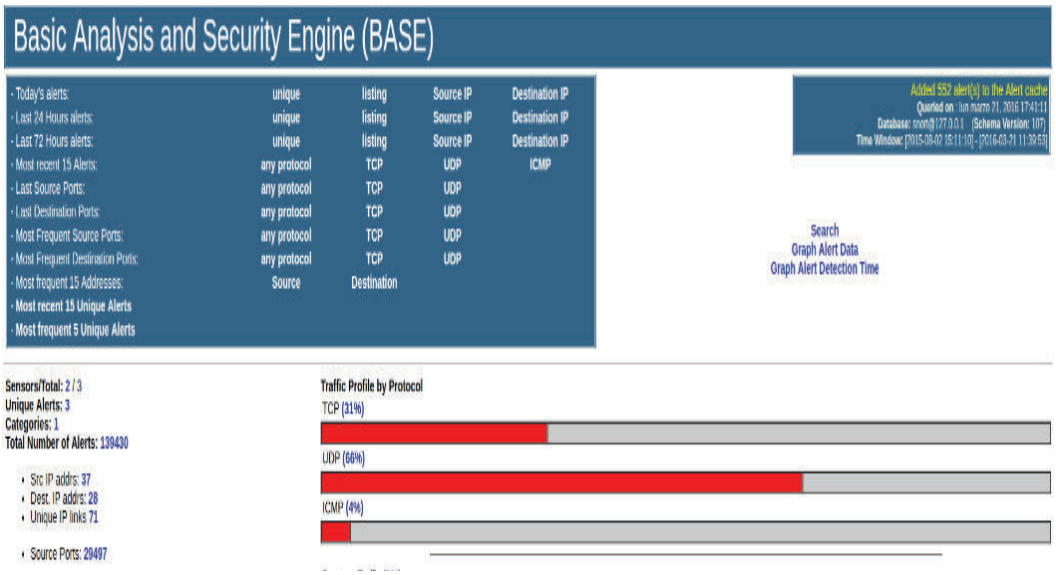


Figura 4.44 Interfaz web front end BASE

4.1.4.2 Ossec GUI

A diferencia de BASE utilizado como fornt-end de Snort, Ossec cuenta con su propia interfaz web, la cual se la implementa siguiendo el procedimiento.

a) Herramientas Adicionales

Ossec GUI requiere de un servidor web que interprete PHP y una base de datos. Las herramientas cumplen estos requerimientos son: Apache, y MySQL.

b) Configuración de MySQL

Creación de base de datos para almacenar los eventos generados por Ossec.
mysql> create database ossec;
mysql> grant INSERT, SELECT, UPDATE,CREATE, DELETE, EXECUTE on Ossec.* to ossec_u;
mysql> set password for ossec_u = PASSWORD ('password');
mysql> flush priviligies; exit

c) Configuración MySQL en Ossec

Una vez que se creó la base de datos de Ossec, se requiere la conexión la misma, esto hace a través del archivo *ossec.config, como en* en la Figura 4.45

```
<ossec_config>
    <database_output>
        <hostname>127.0.0.1</hostname>
        <username>ossec_u</username>
        <password>Passw0rd</password>
        <database>ossec</database>
        <type>mysql</type>
    </database_output>
</ossec_config>
```

Figura 4.45 Conexión de Ossec con la base de datos

d) Ossec Web UI

La instalación de Ossec Web UI se lo realiza ejecutando los siguientes comandos:
sudo wget http://www.ossec.net/files/ossec-wui-0.8-beta-1.tar.gz

sudo tar -xf ossec-wui-0.8-beta-1.tar.gz
sudo mkdir /var/www/ossec/
sudo mv ossec-wui-0.8-beta-1/* /var/www/ossec/
sudo chown www-data:www-data /var/www/ossec/tmp/
sudo chmod 666 /var/www/ossec/tmp

Una vez que la instalación ha sido finalizada sin inconvenientes, se inician los servicios de apache y mysql mediante el comando *"var/ossec/bin/ossec-control start*". La interfaz que presenta Ossec se muestra en la Figura 4.46

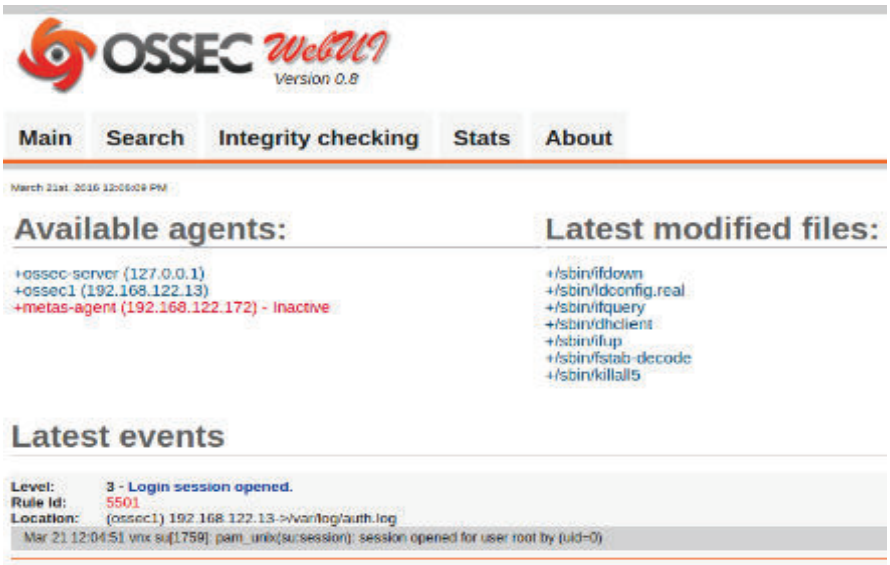


Figura 4.46 Interfaz Web UI Ossec

4.1.4.3 Interfaz General

Existen 2 razones esenciales para diseñar e implementar una interfaz web central, para el proyecto: La primera consiste en que, debido a la diversidad de herramientas que se utilizan para diseñar la solución de seguridad, y la localización distribuida de las mismas en la topología de red jerárquica, se requiere contar con interfaz central.

La segunda tiene que ver con el almacenamiento y presentación de las respuestas ejecutadas por un IRS, ya que en la actualidad no existe herramienta Open Source que provea de esta funcionalidad.

Por estas razones se han diseñado 2 instancias de bases de datos, las cuales almacenarán las acciones de respuesta ejecutadas por Snortsam y Ossec, y una Interfaz Web desde la cual se presenta la información general del proyecto, y un módulo para la reportería. La descripción del procedimiento que se siguió para implementar la Interfaz web general, se lo presenta a continuación.

a) Creación de bases de datos

A pesar que los logs generados por Snortsam y Ossec al ejecutar una respuesta son diferentes, el procedimiento para almacenarlos y presentarlos en una interfaz web es el mismo. Los campos de las tablas de las bases de datos creadas, han sido creados en función de los campos que contiene cada uno de los logs.

Snortsam:

mysql> create database snortsam;
mysql> create table logs (Fecha DATE, Hora VARCHAR(20), DireccionIP VARCHAR(20), IoD VARCHAR(10), Herramienta VARCHAR(15), Informe VARCHAR(200));

Ossec:

mysql> create database ossec;
mysql> create table logs (diasemana VARCHAR(5), mes VARCHAR(5), IPorigen VARCHAR(20), diames VARCHAR(5), hora VARCHAR(20), Meridiano VARCHAR(10), Año VARCHAR(10), path VARCHAR(100),AD VARCHAR(100),espacio VARCHAR(3), IPblock VARCHAR(20), Proceso VARCHAR(30), IDIntrusion VARCHAR(50));

b) Creación de Scripts

Los scripts, sirven para dar formato a los logs que se generaron por la ejecución de las respuestas, que inició Snortsam y Ossec, de tal manera que puedan ser almacenados en las bases de datos. Los scripts se muestran en la figura 4.47 y 4.48, ejecutados por Snortsam y Ossec respectivamente.

```
#!/bin/bash
cat /usr/local/bin/snortsam.log | grep -v 'trying to re-sync' >> updateBDDirs.txt
scp updateBDDirs.txt 192.168.122.44:/home/vnx/Desktop/BDDSnortsam/.
#scp updateBDDirs.txt 10.1.1.14:/home/vnx/Desktop/BDDSnortsam/.
```

Figura 4.47 Script para dar formato a logs de Snortsam

```
#!/bin/bash
sed -e 's/  / 192.168.122.13 /' /var/ossec/logs/active-responses.log > /var/ossec/logs/active-respon
sesbdd.log
cat /var/ossec/logs/active-responsesbdd.log | grep -v '(no_rule_id)' > updateBDDirsOSSEC.txt
scp updateBDDirsOSSEC.txt 192.168.122.44:/home/vnx/Desktop/BDDOssec/.
```

Figura 4.48 Script para dar formato a logs de Ossec

El script utilizado para insertar los datos previamente modificados es el siguiente:.

#!/bin/sh

mysql -u root -p 'snortsam' -D snortsam < automateossec.sql

LOAD DATA INFILE '/home/vnx/Desktop/BDDSnortsam/updateBDDirs.txt' INTO TABLE logs FIELDS TERMINATED BY ',';

c) Creación de Interfaz web

El interfaz web fue realizado en lenguaje PHP, el resultado del mismo se lo puede ver en la figura 4.49.

Figura 4.49 Módulos de Interfaz Web

4.2 PRUEBAS DE FUNCIONAMIENTO Y RESULTADOS

Las pruebas que se presentan a continuación, se enfocan en validar la solución de seguridad frente a ataques DoS. Las pruebas de funcionamiento consisten en desplegar la topología de red jerárquica virtualizada, haciendo uso de VNX. Una vez que se consiguió los escenarios adecuados se cumple el siguiente protocolo presentado a continuación.

4.2.1 Protocolo de pruebas

Para llevar a cabo las pruebas que validen el funcionamiento de la solución, se cumplió el procedimiento presentado en la Tabla 4.9

Orden	Actividades	Herramientas
A	Generación y ejecución de ataques DoS	KALI Linux
B	Detección de ataques	Snort (NIDS) Ossec (HIDS)
C	Generación de respuesta activa	Snortsam Ossec
D	Almacenamiento de ataques detectados	Barnyard2 BDD
E	Ejecución de respuesta activa	Iptables
F	Almacenamiento de respuesta ejecuta	Scripts (c) BDD Snortsam BDD Ossec
G	Presentación de respuestas	BASE Ossec web UI Interfaz General

Tabla 4.10 Protocolo de validación de solución *configurados manualmente

4.2.2 Validación de Solución

La solución fue validada mediante el cumplimiento de cada uno de los puntos presentados en el cuadro anterior. Para las pruebas se implementaron los siguientes escenarios.

4.2.2.1 Ataques Externos

Desde una máquina localizada en el exterior de la red, específicamente con sistema operativo Kali Linux, se generan 4 diferentes tipos de ataques, los cuales serán presentados con más detalle, en las siguientes secciones. Estos serán dirigidos hacia la DMZ y granja de servidores, como se puede ver en la Figura 4.50

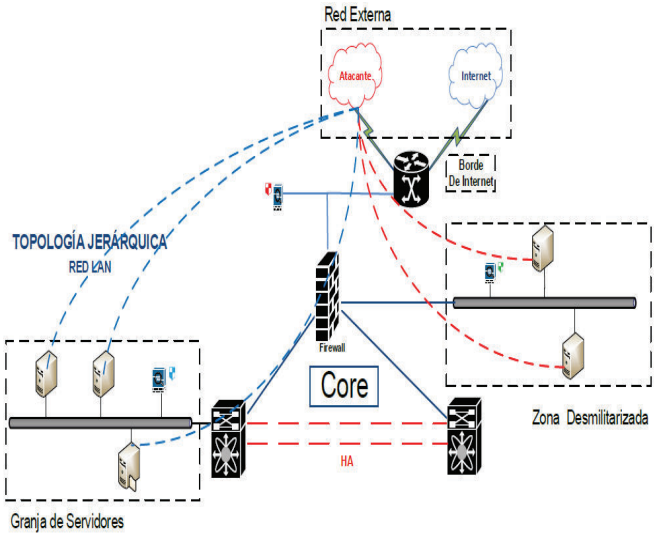


Figura 4.50 Ataques Externos

4.2.2.2 Ataques Internos

Los ataques Internos son generados desde un host que se encuentra en la capa de acceso de la topología de red jerárquica, el atacante y su localización son presentados en la Figura 4.51.

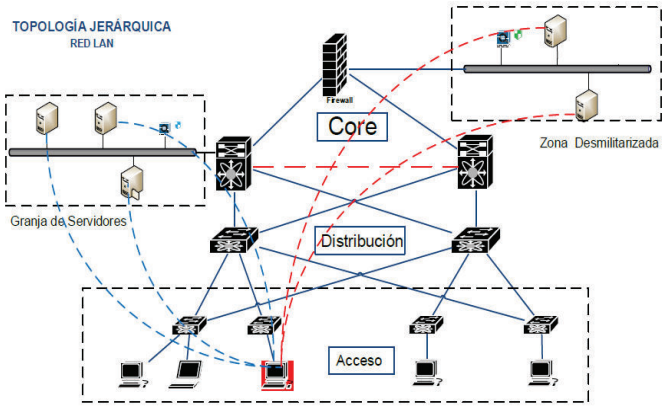


Figura 4.51 Ataques Internos

4.2.2.3 Herramientas para generar ataques.

Los ataques con los cuales se realizó la validación de la solución fueron:

- Slowrloris: Funciona mediante la apertura de varias conexiones al servidor web víctima, manteniéndolos abiertos el mayor tiempo posible. Esto se lleva a cabo mediante el envío de peticiones HTTP de forma continua.
    - Tipo de Ataque: Volumétrico / HTTP GET/POST Flood.
    - Tipo de daño: Disruptivo
    - Enfoque: Esta herramienta puede ser ejecutada tanto desde el interior como el exterior de la red, buscando establecer una comunicación http en un servidor web.

- Hping3 (Herramienta de Kali): Es una herramienta que genera inundación de paquetes ICMP, TCP y UDP mediante la generación de una gran cantidad de paquetes en intervalos pequeños de tiempo, ejecutada en línea de comandos de sistemas operativos Linux.
    - Tipo de Ataque: Volumétrico / SYN Flood.
    - Tipo de Daño: Saturación.
    - Enfoque: La herramienta es configurada para utilizar cualquier puerto que requiera del protocolo TCP, esto sirve para atacar servicios como Telnet, FTP, SMTP, HTTP.

- Fudp: Es una herramienta que permite generar inundación UDP, utilizando por defecto puertos aleatorios de la IP víctima, pero también se puede especificar un puerto en particular. Puede también ser utilizado con tecnología spoofing.
    - Tipo de Ataque: Volumétrico / Fragment Flood
    - Tipo de Daño: Saturación
    - Enfoque: El ataque genera una inundación de paquetes fragmentados udp, los cuales puedes ser ejecutados sobre un servidor de la granja de servidores o uno localizado en la zona desmilitarizada, tanto desde el interior de la red como el exterior de la misma,

- Hydra: Es una aplicación que permite generar una gran cantidad de paquetes de conexión ssh, de tal manera que se obstaculicen las peticiones legítimas.
    - Tipo de Ataque: Volumétrico / SYN Flood
    - Tipo de Daño: Consumo de Recursos
    - Enfoque: La herramienta puede ser utilizada para validar la configuración interna de un host, ya que el ataque tiene como objetivo el servicio de conexión segura SSH, utilizando el puerto 22.

- Ettercap: Es una suite completa, que permite generar varios tipos de ataques. Para el proyecto será orientado a la generación de un ataque DoS mediante la técnica “man in the midle”.
    - Tipo de Ataque: Reflexivo / Syn Ack Reflection Flood.
    - Tipo de Daño: Saturación
    - Enfoque: Este ataque será ejecutado únicamente desde el interior de la red, ya que requiere direcciones IP de una red LAN para realizar un envenenamiento ARP, ocultando una dirección MAC (física) tras una dirección IP (lógica) la cual es la víctima, posteriormente se realiza una inundación de paquetes TCP desde la máquina atacante, pero las respuestas serán enviadas hacia la IP víctima.

4.2.2.4 Análisis de ataques y configuración de IDSs

En la siguiente sección se presenta el análisis realizado para cada uno de los ataques, los cuales permitieron configurar las herramientas IDSs, con la finalidad de diseñar una solución efectiva frente a ataques DoS.

Slowloris

Ataque ejecutado desde el exterior de la red, como se puede ver en la Figura 4.52 cuyo objetivo es el servidor web que se encuentra en la granja de servidores, En la figura 4.53 se puede observar el análisis del script utilizado por el atacante y la regla que detectará un paquete que contenga este payload.

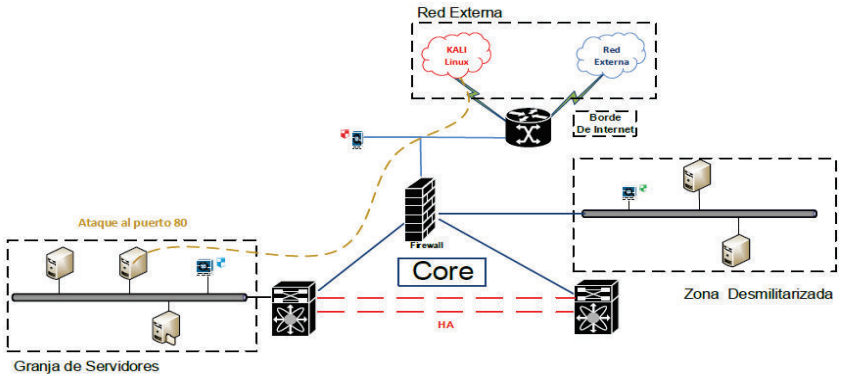


Figura 4.52 Ataque Slowloris

Regla utilizada para detección: alert tcp any any -> any any (msg: "DoS by slowloris Tool"; content:"GET /"; depth:10;content:"User-Agent\: Mozilla/4.0 (compatible\; MSIE 7.0\; Windows NT 5.1\; Trindent/4.0"; offset:10;depth:100;sid:887799; rev:1; fwsam: src, 10 minutes;)

```
my $primarypayload =
    . "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trid
ent/4.0; .NET CLR 1.1.4322; .NET CLR 2.0.50313; .NET CLR 3.0.4506.2152; .NET CLR
 3.5.30729; MSOffice 12)\r\n"
    . "Content-Length: 42\r\n";
if ( print $sock $primarypayload ) {
    print "Connection successful, now comes the waiting game...\n";
}
```

Figura 4.53 Análisis de ataque Slowloris

Hping3

Utilizado para generar una gran cantidad de paquetes TCP utilizando el puerto 25 (SMTP) como se puede ver en la figura 4.54. En la Figura 4.55 se puede observar la trama detectada por Snort, y su análisis respectivo con la trama TCP /IP, lo cual permite generar una regla que permita detectar este tipo de ataques.

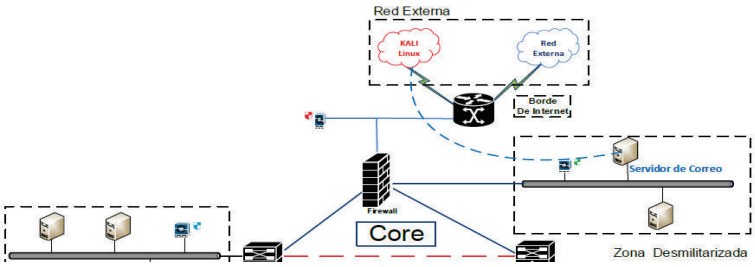


Figura 4.54 Ataque Hping3

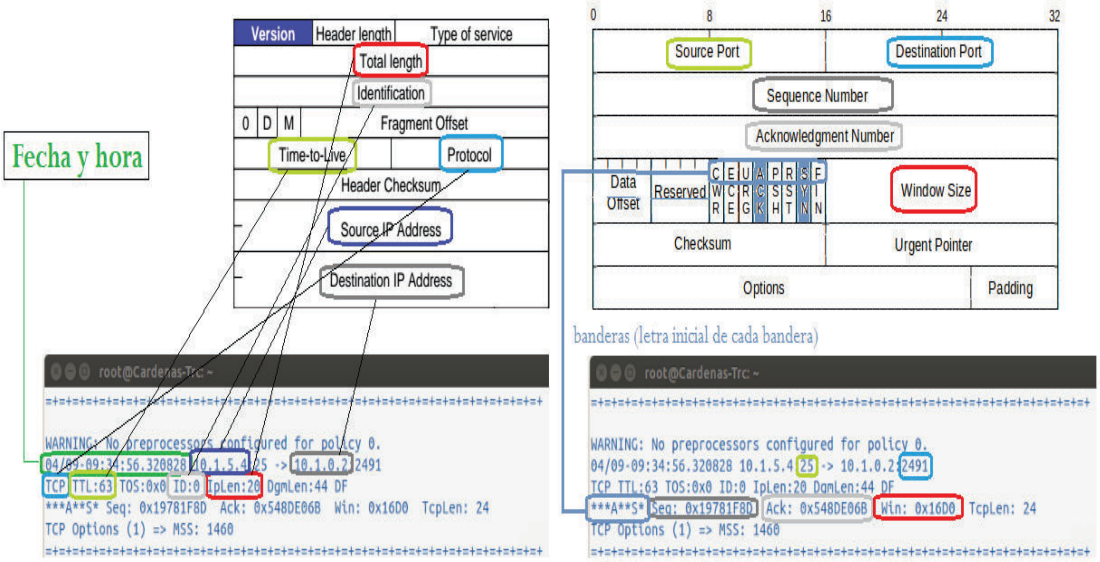


Figura 4.55 Análisis de ataque Hping3

FUDP

El ataque fudp consiste en generar varios paquetes UDP fragmentados, los cuales serán enviados a un servidor localizado en la zona desmilitarizada, con la intención de denegar sus servicio, el escenario se lo puede apreciar en la Figura 4.56

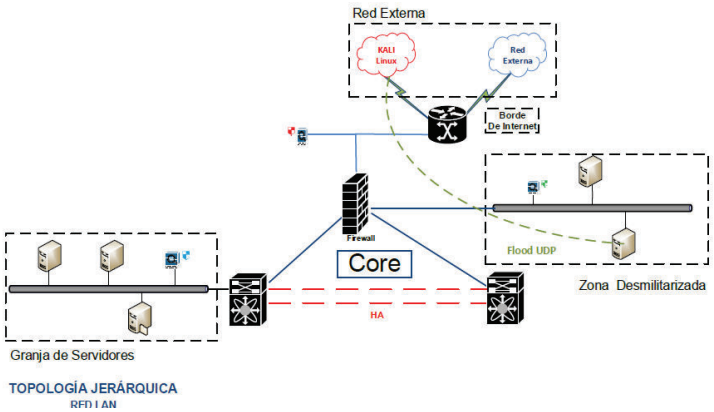


Figura 4.56 Ataque FUDP

Para este tipo de ataques en los que se detecta paquetes fragmentados, NO solamente es necesaria la configuración una regla de Snort, sino también intervienen los preprocesadores de Snort, los cuales definen un comportamiento específico que Snort deberá detectar al momento de ser ejecutado como un NIDS. El preprocesador es independiente de las reglas de Snort, pero estas a su

vez pueden hacer uso de ellos con motivo de crear una regla más específica. Uno de los preprocesadores más utilizados es: **preprocessor frag3_global: prealloc_frags 8192.** El cual indica a Snort que se debe detectar paquetes fragmentados. El análisis de la trama detectada se presenta em la Figura 4.57.

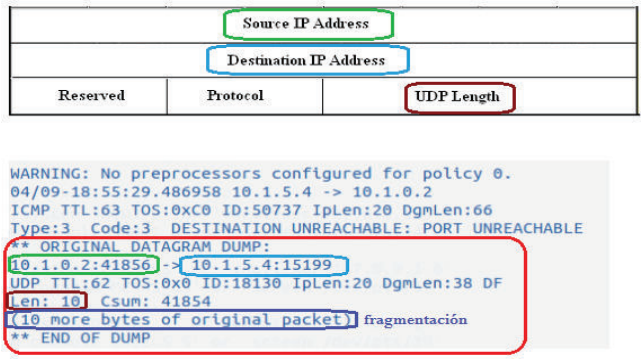


Figura 4.57 Análisis de ataque FUDP

Hydra

Este tipo de ataques, difícilmente será detectado por un NIDS, ya que no genera flujos de tráfico inusuales, o lleva patrones sospechosos dentro de su payload. Para solventar este inconveniente se hace uso de un IDS basado en Host, el cual se encuentra en constante monitoreo de la integridad de una determinada máquina. El ataque Hydra tiene como objetivo específico acceder a un servidor ssh tras la generación de varias peticiones, como se puede ver en la Figura 4.58

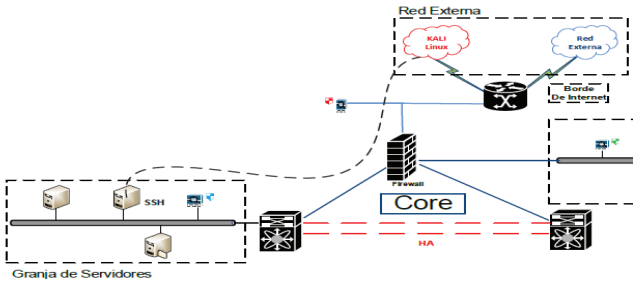


Figura 4.58 Ataque Hydra

En la figura 4.59 podemos ver como se genera una respuesta activa tras la información otorgada por los agentes HIDS Ossec, la cual consiste en indicar aquellas reglas que serán tomadas en consideración para iniciar el script que ejecute una respuesta activa.

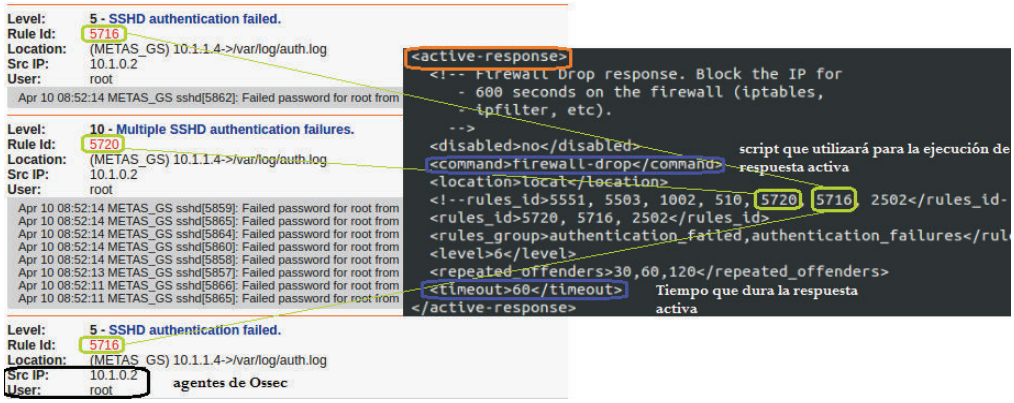


Figura 4.59 Análisis de ataque Hydra

Ettercap

Este ataque a diferencia de los anteriores es ejecutado únicamente desde el interior de la red tal como se observa en la Figura 4.60, ya que su funcionamiento se divide en dos actividades. En la primera el atacante intentará ocultar su IP realizando un ARP poisoning (envenenamiento). La siguiente actividad será enviar una gran cantidad de ataques TCP, ejecutadas desde el atacante pero ocultándose tras la IP víctima.

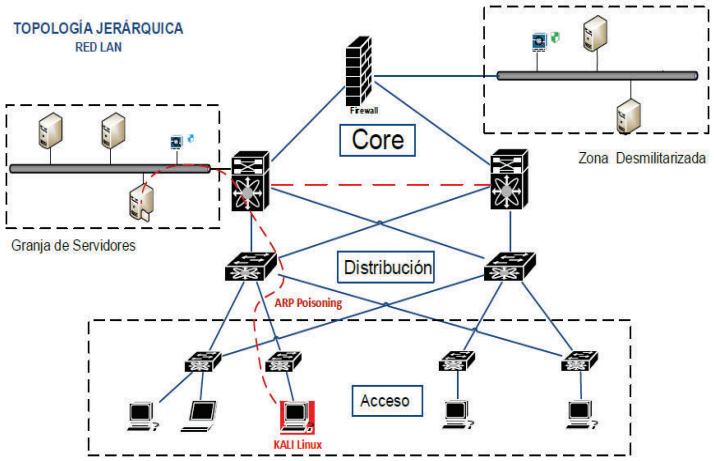


Figura 4.60 Ataque Ettercap

Para este caso también se requiere de la configuración de los preprocesadores de Snort, los cuales permiten indicar la dirección IP y MAC de un determinado host, y en caso de que este host intente realizar un envenenamiento ARP se genera una alerta. La configuración del preprocesador es la siguiente:

preprocessor arpspoof_detect_host: 10.1.4.4 f0:0f:00:f0:0f:00.

4.2.2.5 Demostración de Solución

A continuación se describe en imágenes el protocolo que se llevó a cabo para validar la solución. La Figura 4.61 muestra un funcionamiento normal de un servidor web accedido desde una red externa.

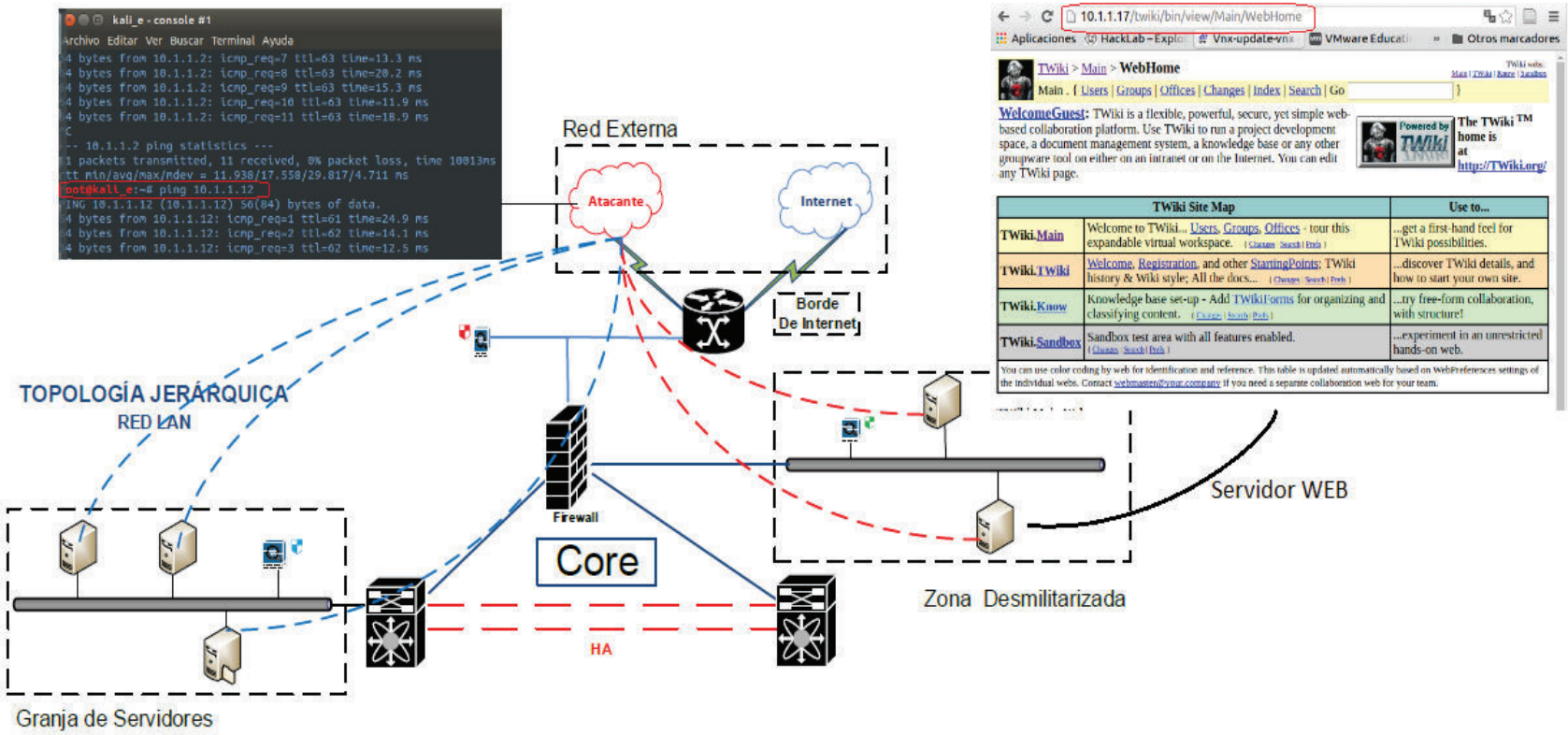


Figura 4.61 Funcionamiento normal de servidor web

A. Generación de Ataque Dos con Kali Linux (Figura 4.62)

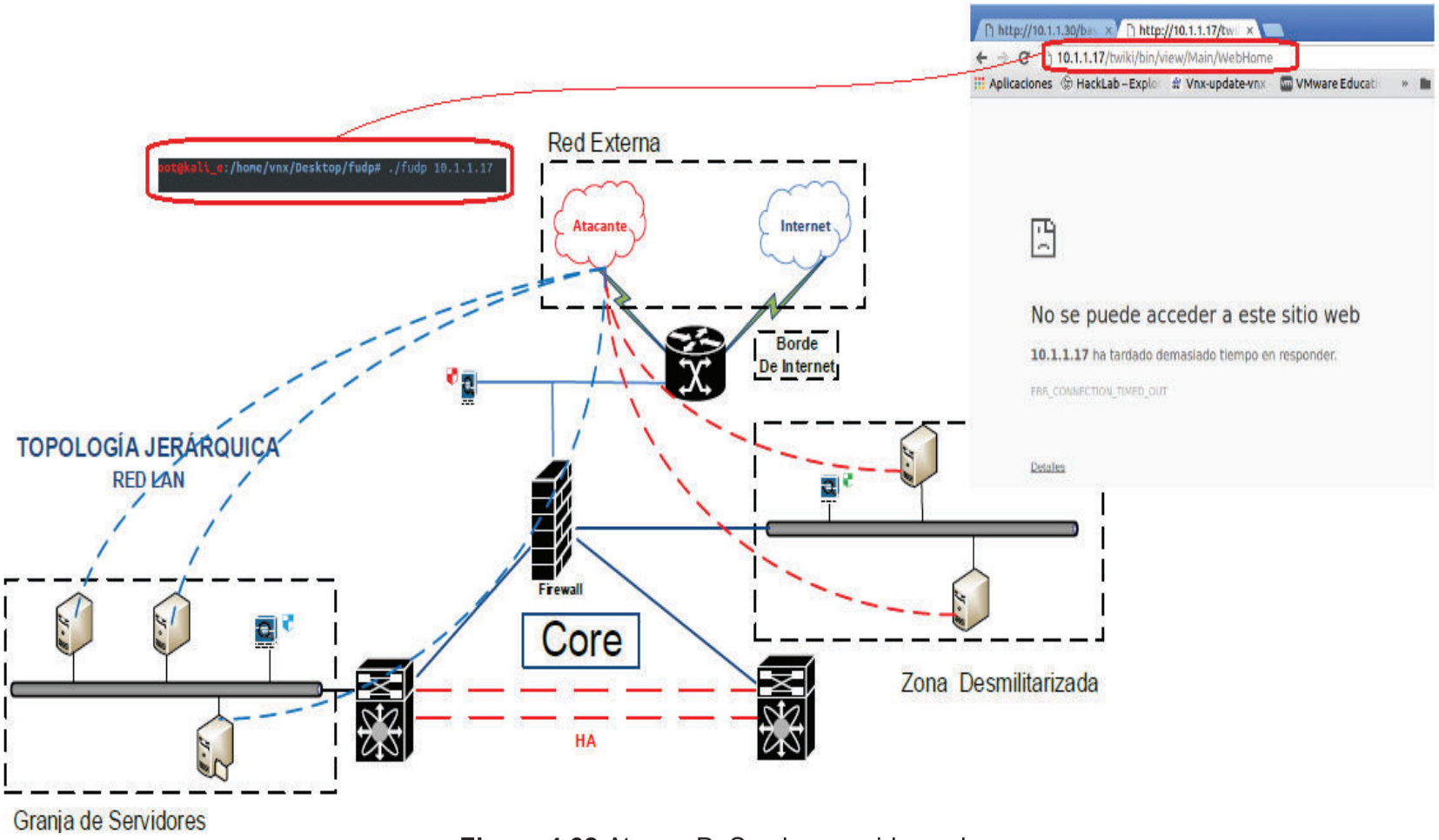


Figura 4.62 Ataque DoS sobre servidor web

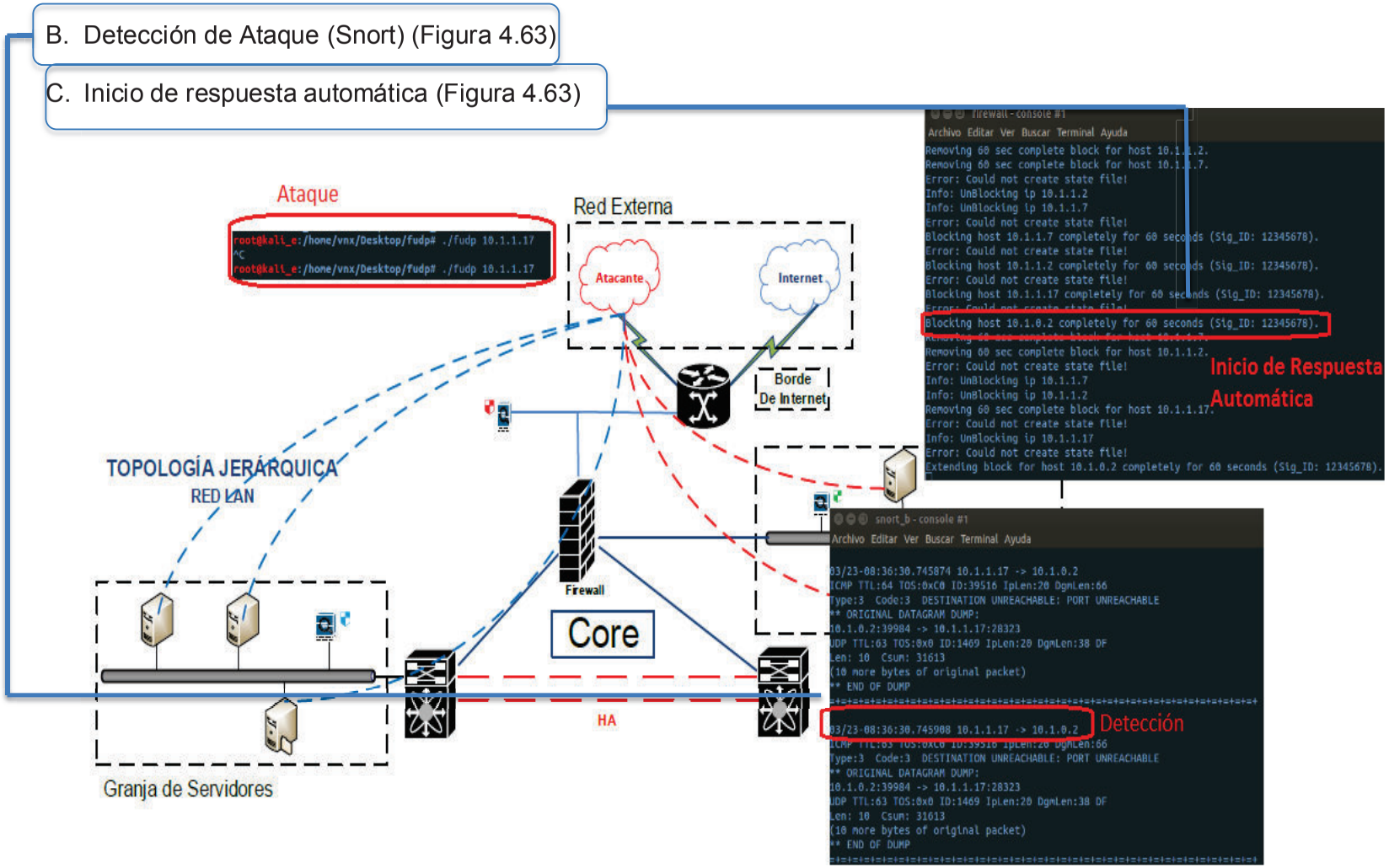


Figura 4.63 Detección e inicio de respuesta automática

D. Almacenamiento de Intrusión detectada (Figura 4.64)

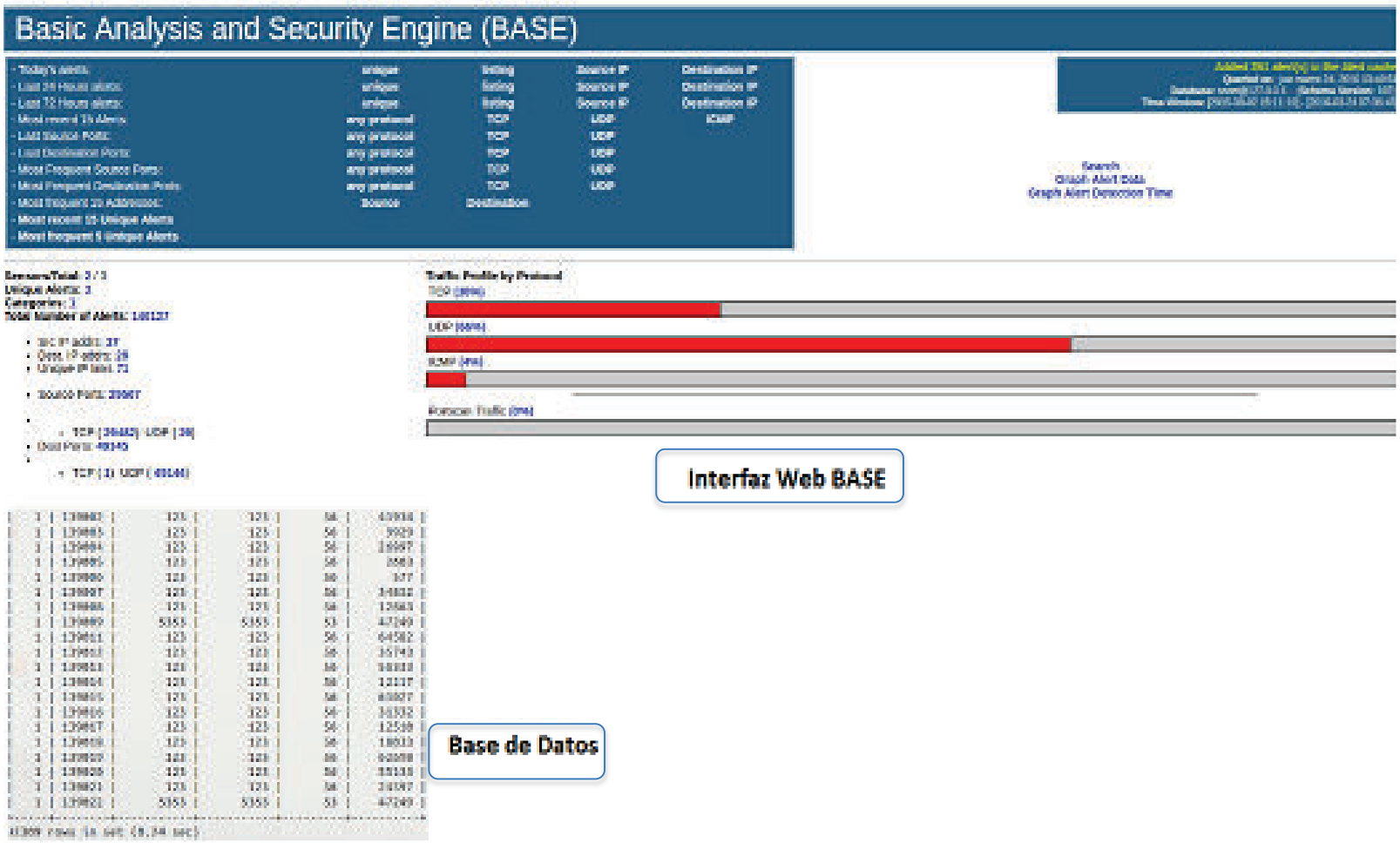


Figura 4.64 Almacenamiento y presentación de detecciones

E. Ejecución de Respuesta Activa (Snortsam Figura 4.65)

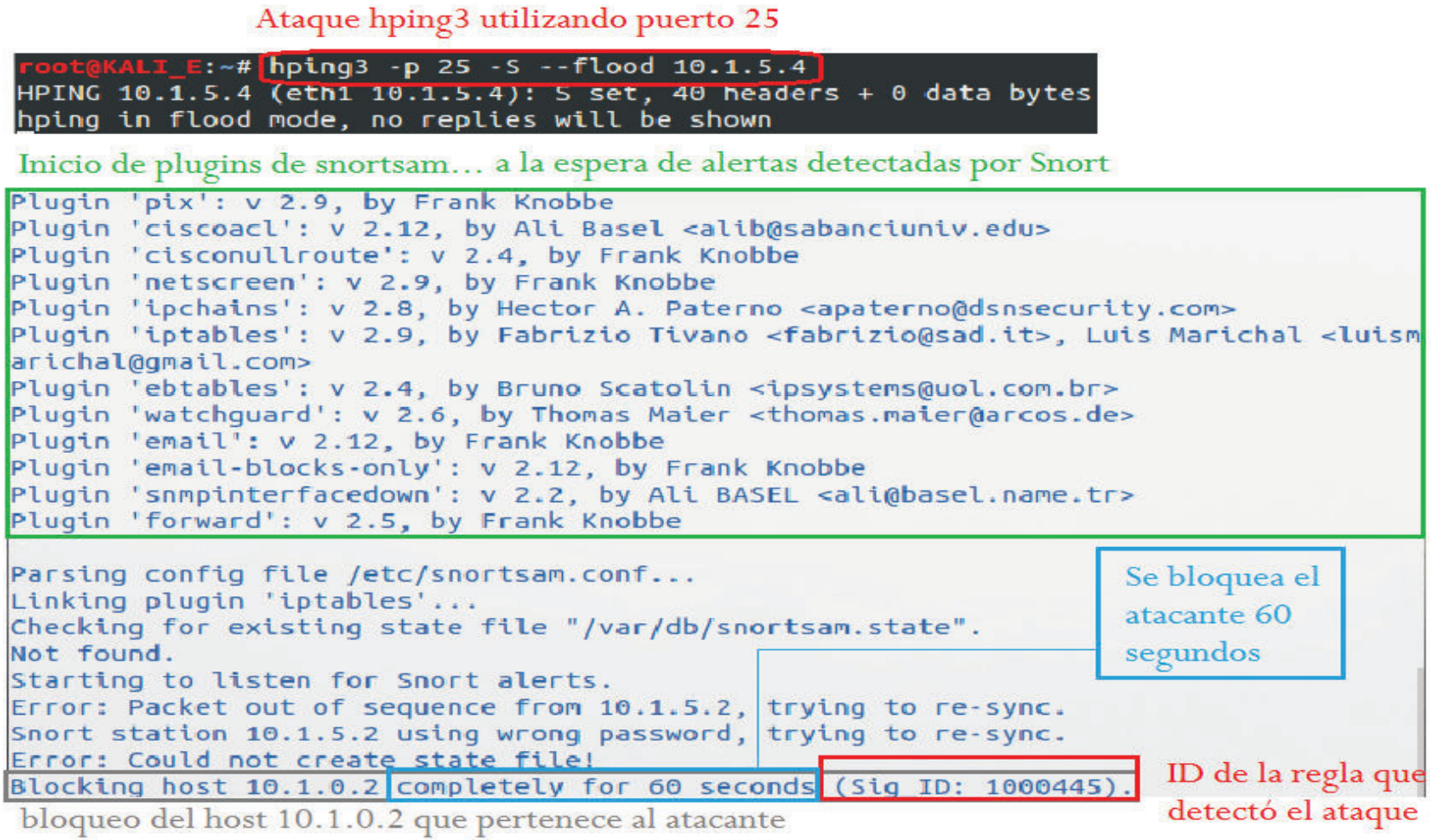


Figura 4.65 Ejecución de respuesta activa

F. Almacenamiento en base de datos (se consigue tras ejecutar un script que asigna la forma de una entrada a base de datos a los logs que se generan de la detección) (Figura 4.66)

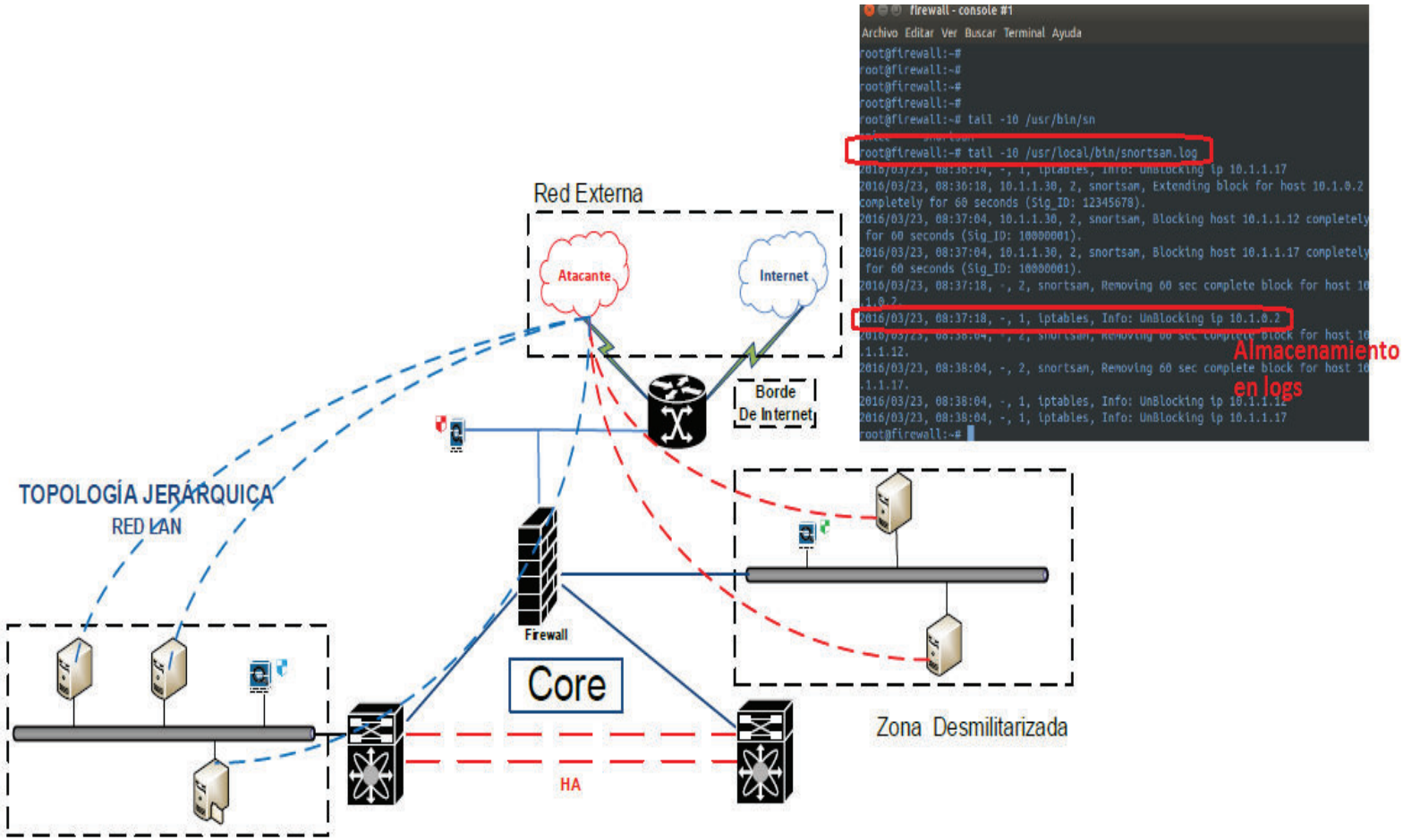


Figura 4.66 Almacenamiento de respuestas ejecutadas

G. Presentación de Respuesta Activa (Figura 4.67)

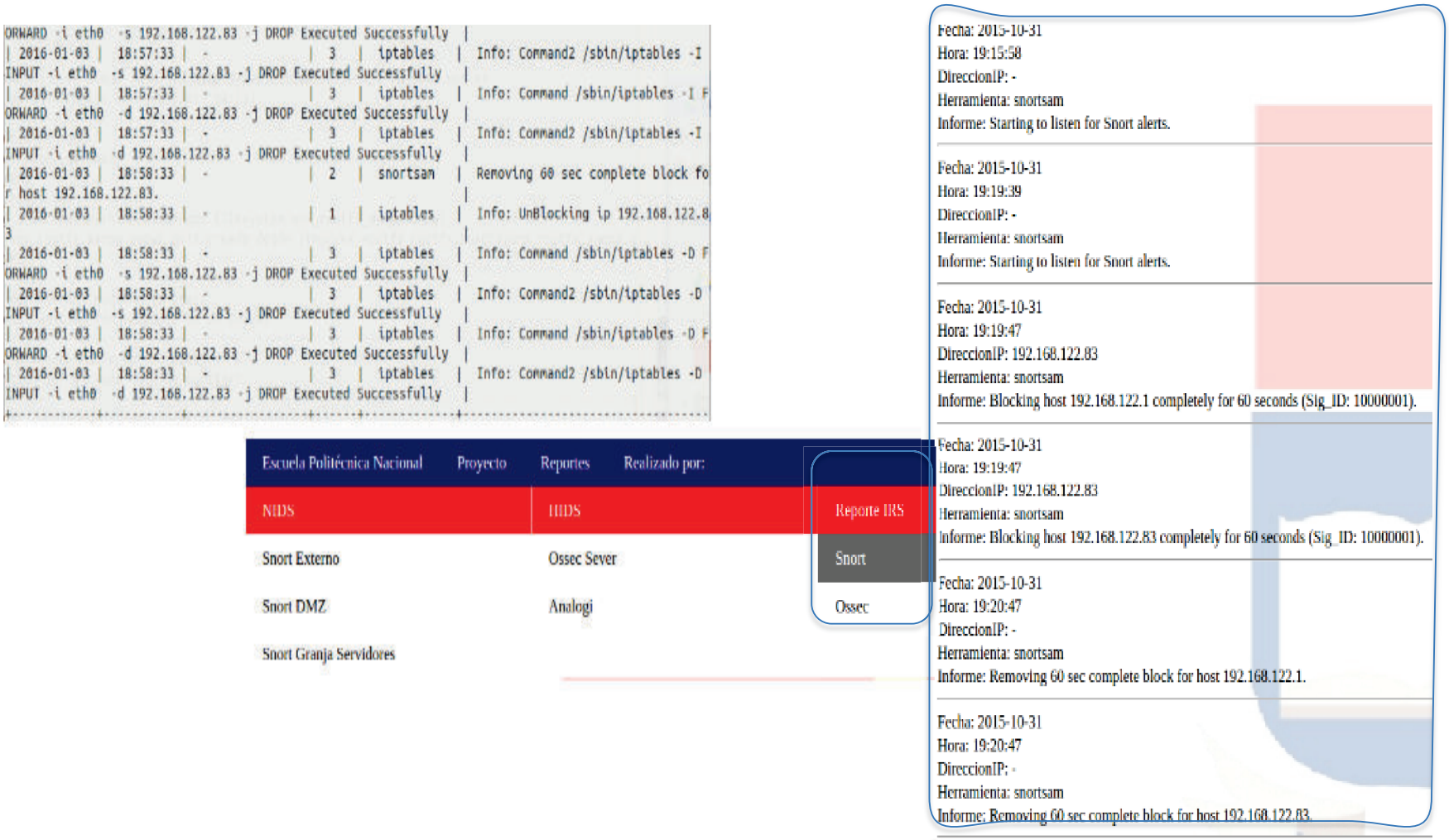


Figura 4.67 Presentación de respuestas ejecutadas.

4.2.2.6 Matrices de validación de solución

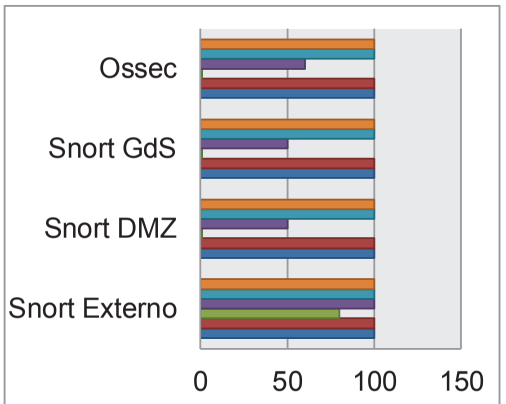
Las siguientes matrices, presentan la validación de la solución después de haber ejecutado los ataques DoS, con las cuales se puede tener una percepción más real sobre la efectividad de la misma. A su vez da un valor agregado al análisis que se pueda realizar sobre una solución de seguridad basada en un sistema de respuesta activa.

Ataques Ejecutados	Ataques Detectados (IDS)				Ataques Mitigados (IRS)		
Ataques Externos							
Herramienta	IDS	Reglas Efectivas	Dificultad de detección	Almacebamiento/GUI	IRS	Dispositivo de Seguridad	Almacebamiento/GUI
Slowloris	Snort	1/3	Alta	Ok	Ok	Ok	Ok
Hping3 (TCP-25)	Snort	1/2	Alta	Ok	Ok	Ok	Ok
Hydra	Ossec	1/6	Media	Ok	Ok	Ok	Ok
FUDP	Snort	2/3	Alta	Ok	Ok	Ok	Ok
Ataques Internos							
Ettercap	Snort /Ossec	1/3	Alta	Ok	0	0	0
Hping3 (UDP-53)	Snort	1/2	Alta	Ok	0	0	0
Slowloris	Snort/Ossec	1/3	Alta	Ok	0	0	0
Hydra	Ossec	2/5	Alta	Ok	Ok	Ok	Ok

Tabla 4.11 Matriz de validación de solución

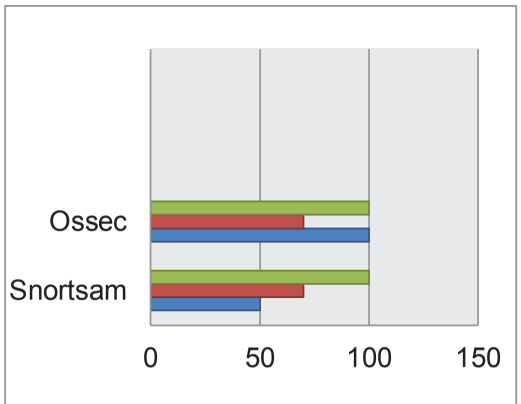
Conclusión de solución mediante gráficas

IDS



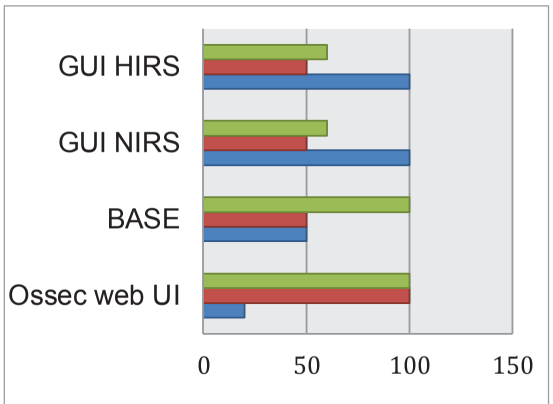
- Ataques Ext: Relación entre ataques y detecciones (4/4).
- Ataques Int: Relación entre ataques y detecciones (4/4).
- Reglas Efectivas:Reglas efectivas de Snort (4/8), Ossec (12/20).
- Falsos Negativos: Generación de falsas alarmas,(Alto en zona roja debido a la sensibilidad del IDS)

IRS



- Integración con dispositivos de seguridad: Snort con Firewall perimetral, Ossec con Iptables de cada host
- Almacenamiento de respuestas ejecutadas: Nivel de automatización de almacenamiento (ejecución de scripts).
- Efectividad de respuesta: Ejecución de cada de respuestas ejecutadas desde el interior y exterior de la red.

GUI



- Presentación de GUI: Interfaz gráfico que se muestra al usuario final.
- Automatización de almacenamiento y presentación: Nivel de automatización para almacenar y presentar los eventos de seguridad.
- Dif. Implementación: Nivel de dificultad en la implementación de la interfaz gráfica.

Tabla 4.12 Análisis de solución mediante gráficas

Ataque			Detección	Respuesta Activa			Almacenamiento
Ataques Ejecutados (Herramienta)	Descripción	Herramienta	Regla Efectiva	Herramienta	Dispositivo de Seguridad	IRS	Base de Datos
Slowloris	Externo – Interno	Snort / Ossec	#Regla Snort alert tcp any 80 -> any any (msg:"Alert Getting TCP Flood Messag";sid:1000005;flags: A;rev:1;fwsam: dst, 1 minutes;) #Regla Ossec <rule id=”100002″ level=”5″> <if_sid>31108</if_sid> <id>^206$</id> <description>Web Server 206 Error Code</description> </rule> <rule id=”100003″ level=”10″ frequency=”8″ timeframe=”5″> <if_matched_sid>100002</if_matched_sid> <same_location /> <same_source_ip /> <description>Multiple Web Server 206 Error Codes </description> <description>from Same Source IP</description> <group>web_scan,recon,</group> </rule>	Snortsam	Firewall	OK	Snort Barnyard2

Tabla 4.13 Resumen de Ataques y herramientas (1/3)

Hping3	Externo – Interno	Snort	http://10.1.5.4/index.phphttp://10.1.5.4/index.php alert tcp any any -> any 25 (flags: S; msg:"Ataque Hping3"; flow: stateless; thrr eshold: type both, track by_src, count 70, seconds 10; sid:1000444;rev:1;fwsam: src, 1 minutes;) alert tcp any 25 -> any any (flags: AS; msg:"Ataque Hping3"; flow: stateless; thh reshold: type both, track by_src, count 70, seconds 10; sid:1000445;rev:1;fwsam:: dst, 1 minutes;) **Preprocesador:** preprocessor frag3_global: prealloc_frags 8192	Snortsam	Firewall	OK	Snort Barnyard2
Fudp	Externo	Snort	alert udp any any -> any any (msg:"DOS flood denial of service attempt";flow:to_server;detection_filter:track by_dst,count 50, seconds 1; metadata:service syslog; classtype:attempted- dos;sid:25102;rev:1;fwsam: src, 1 minutes;) alert udp any any -> any any (msg:"DOS flood denial of service attempt";flow:to_server;detection_filter:track by_dst,count 50, seconds 1; metadata:service syslog; classtype:attempted- dos;sid:251144;rev:1;fwsam: src, 1 minutes;) alert icmp any any -> any any (msg :"ICMP test"; itype: 3; sid:10000441; rev:001; fwsam:dst, 2minutes)	Snortsam	Firewall	OK	Snort Barnyard2

Tabla 4.14 Resumen de Ataques y herramientas (2/3)

Hydra	Externo – Interno	Ossec – Snort	alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"ET SCAN Hydra User-Agent"; flow: established,to_server; content:"User-Agent|3A| Mozilla/4.0 (Hydra)"; nocase; http_header; fast_pattern:23,8; threshold: type limit, track by_src,count 1, seconds 60; reference:url,freeworld.thc.org/thc-hydra; classtype:attempted-recon; sid:2011497; rev:3;) Reglas Ossec: Ssh: 2502, 5716, 5503, 1002	Snortsam Ossec (respuesta activa)	Firewall Iptables (de cada host)	OK	Snort Barnyard2 Ossec
Ettercap	Interno	Snort	preprocessor arpspoof_detect_host: 10.1.4.4 f0:0f:00:f0:0f:00. Arpwatch_rules.xml <group name="syslog,arpwatch,"> <rule id="7200" level="0" noalert="1"> <decoded_as>arpwatch</decoded_as> <description>Grouping of the arpwatch rules.</description> </rule> <rule id="7201" level="4"> <if_sid>7200</if_sid> <options>alert_by_email</options> <if_fts /> <description>Arpwatch new host detected.</description> <group>new_host,</group> </rule>	Snortsam Ossec (respuesta activa)	Iptables	--	Snort Barnyard2

Tabla 4.15 Resumen de Ataques y herramientas (3/3)

4.2.2.7 Análisis de la solución.

Después de haber realizado las pruebas mediante ataques DoS, se pudo apreciar la efectividad de la solución para 4 de los 5 ataques DoS llevados a cabo. Es importante tener presente que la detección de las intrusiones está supeditada a la creación de reglas de Snort y la ejecución de una respuesta por parte de Snortsam y Ossec depende de la información contenida en la alerta. El ataque Ettercap si bien fue detectado, no se pudo ejecutar una acción de respuesta ya que no se contaba con un plugin para ello. Una posible solución a este inconveniente es la implementación de plugins de Snortsam y Ossec más sofisticados, que tengan la inteligencia necesaria para determinar sobre qué dispositivo actuar.

El ataque efectuado con Ettercap, puede generar varios falsos negativos, esto se debe a que el ataque puede ocultarse tras una dirección IP legítima. Es importante analizar, que pasaría si el atacante logra suplantar una dirección importante como la del servidor DNS, la naturaleza de la solución crearía una regla en el firewall, y esta quedaría inaccesible para los demás host que requieren sus servicios. Es por esta razón que existen servidores que han sido localizados en listas blancas, las cuales quedarán excluidas de cualquier respuesta activa que se haya configurado llevar a cabo.

Tanto los falsos positivos como negativos, han sido disminuidos al crear reglas específicas para cada ataque ejecutado. Tanto las reglas configuradas para Ossec como Snort fueron basados en los análisis posteriores a los ataques ejecutados, el siguiente paso será adecuar las reglas creadas para que puedan ser estandarizadas y optimizadas, es decir que, fruto de este trabajo se pueda potencializar la solución, y que esta pueda proteger ataques DoS o cualquier otro tipo de ataque de manera más automatizada.

Los preprocesadores fueron de gran ayuda para la generación de reglas, las cuales a su vez permitieron detectar los ataques. Esto se debe a que cada preprocesador extiende una funcionalidad de un IDS otorgando la capacidad de ordenar la información, para que esta pueda ser comparada por cada una de las

reglas configuradas. Además permite iniciar el chequeo de comportamientos como: el análisis de paquetes fragmentados, el inicio de un escaneo de puertos, o el intento de un envenenamiento ARP.

El número de eventos que se generan tras haber detectado un ataque Dos dependerá de la configuración de la regla que permitió su detección, es decir que si una regla detecta un ataque en un trama específica, se generan alertas por cada uno de los paquetes generados por el atacante, por tal razón y tratándose de un ataque DoS se generan una gran cantidad de alertas, las cuales son almacenadas en la base de datos correspondiente y posteriormente presentadas a través de una interfaz gráfica.

Pero en caso de utilizar una regla, que detecta un ataque tras analizar un conjunto de paquetes en un determinado intervalo de tiempo, las alertas generadas se verán notablemente disminuidas, esto optimizará el almacenamiento de las mismas, y por ende, el rendimiento del IDS. Por esta razón se pudo observar que una regla que sea efectiva para detectar ataques DoS no necesariamente es eficiente, ya que los recursos que utiliza para detectar el ataque son mayores a los que se utilizaría al implementar una regla más óptima. Para cada uno de los ataques se configuraron varias reglas, de las cuales no todas fueron efectivas para detectar el ataque, sin embargo se logró configurar y obtener al menos una por cada ataque.

Los ataques DoS que provienen desde el exterior de la red pueden ser detectados por el IDS localizado en la zona roja. La sensibilidad de este IDS es baja, ya que la cantidad de falsos negativos que se pueden generar en esta área es alta. Para el proyecto se ha manejado en nivel de sensibilidad mediante la configuración de reglas en cada uno de los IDSs.

Tanto el IDS que protege a la red DMZ como a la granja de servidores, manejan sensibilidades más altas. Cada regla configurada en estos IDSs dependió de los servidores que se encuentran en cada una de las áreas respectivamente, este permite a que el IDS trabaje de manera óptima sin estar sobrecargado de reglas o preprocesadores.

La mitigación de los ataques DoS desde el exterior de la red se lo hace a través del bloqueo de paquetes en el firewall perimetral. La configuración del IRS Snortsam permite generar una acción de respuesta cuando esta se comunica con un agente previamente instalado en un firewall o router de borde, por tal razón no es muy eficiente contar solo con esta herramienta para responder a ataques efectuados dentro de la red.

Otra de las falencias con las que trabaja Snortsam, es que a pesar que inicia respuestas activa frente a un determinado ataque, solo cuenta con la opción de generar una regla en el firewall, sin haber analizado el escenario o al Host que se pretende proteger, esto desemboca en que, a pesar de detectar y bloquear un ataque DoS, puede que el atacante haya conseguido ocultarse tras una dirección IP legítima, por ende la acción de respuesta se ejecutará sobre esta y mas no sobre el atacante.

Para los ataques internos, se hace uso de la funcionalidad de agente -servidor con la que cuenta Ossec, el cual analiza a cada host donde se ha instalado el agente, y en caso de detectar comportamientos anómalos este genera una alerta, la cual será analizada por Ossec server y mitigada en caso de ser necesario.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- Mediante la integración de herramientas open source se provee exitosamente de un sistema de respuesta activa frente ataques de denegación de servicio, los cuales fueron detectados y mitigados. A su vez el almacenamiento y presentación tanto de las detecciones como de las respuestas ejecutadas permiten conocer cuando una red está siendo atacada y cuál es la acción de respuesta que se ejecutó sobre la misma.

- La virtualización permite desplegar un escenario de red con topología jerárquica completo, haciendo un uso óptimo de los recursos de hardware (procesamiento- almacenamiento y memoria) provistos por el host que alberga el hipervisor y demás herramientas que permiten implementar y gestionar un escenario virtual

- Se realizó una revisión de las herramientas open source y comerciales que proveen la funcionalidad de IDS, IPS, IRS, y consolas que permitan presentar las intrusiones detectadas y respuestas ejecutadas, haciendo una comparación técnica funcional entre ellas, basándose en fuentes bibliográficas confiables. Esto permitió la selección de las herramientas de mejor desempeño y que presentaban la capacidad de adaptarse a los objetivos planteados.

- Se considera que existen 3 aportes esenciales que se han desarrollado en este proyecto.
    - El primero tiene que ver con la creación y despliegue de una topología de red jerárquica basada en xml. Si bien es cierto, que la documentación provee de ejemplos prácticos de la funcionalidad de VNX, no se ha enfocado en diseñar un prototipo de red jerárquico en el cual se pueda probar un IRS distribuido.

- El segundo, es la creación de scripts que permiten almacenar en una base de datos las respuestas activas ejecutadas tanto por Snortsam como Ossec, para después presentarlas a través de un interfaz web. Esto se debe a que a pesar de haber realizado una revisión del estado de la tecnología, no se encontró una herramienta o software que pueda hacerlo.
- El tercer aporte consiste en demostrar que se puede diseñar una solución de seguridad que provea un sistema de respuesta activa sin incurrir en costos de inversión. Es verdad que la curva de aprendizaje que llevo a conseguir una solución basada en la integración de varias herramientas open source fue grande, pero la ventaja de aquello, es que no se utilizó ningún recurso financiero. El diseño de seguridad es completo y eficiente.

- El ataque más complicado de detectar fue Ettercap, ya que este puede ocultar su dirección IP verdadera tras una dirección IP víctima ejecutando un envenenamiento ARP, para lo cual se debió configurar los preprocesadores de Snort de tal manera que puedan detectar el ataque, además del manejo de listas blancas en Snortsam, de tal manera que no se bloquee direcciones IP importantes como la del DNS, en caso de ser víctimas del ataque.

.

5.2 RECOMENDACIONES Y TRABAJOS FUTUROS

- La solución de seguridad respondió de manera correcta, pero su funcionamiento puede ser optimizado automatizando algunos de los módulos que se presentaron en la arquitectura, por ejemplo la selección de una respuesta a ser ejecutada puede estar basada en un análisis previo, siendo una respuesta reactiva pero más efectiva ya que pudo analizar ciertos parámetros antes de ser ejecutada.

- Otro módulo que puede ser optimizado es la interfaz gráfica desde la cual se presentan las herramientas de gestión de reportería. La optimización

consistiría en que desde esta se puedan configurar los IDSs, creando reglas, listas blancas, etc, sin necesidad de ingresar a la máquina virtual en sí.

- A pesar que VNX presenta un gráfico (a través de la ejecución del comando vnx *–show-map*) el cual indica cómo se encuentra la configuración de la red, sería interesante proponer una interfaz gráfica de la tecnología, la cual permita configurar y gestionar máquinas y redes virtuales, es decir que se pueda abstraer la dificultad de la programación en xml.

- Basándose en la efectividad que tuvo la plataforma al detectar y mitigar ataques DoS, podría ser llevada a un análisis comparativo con herramientas comerciales. Dicho análisis puede basarse en métricas, las cuales comparen varios parámetros de ambas soluciones, de los cuales se pueda obtener resultados valiosos para los administradores de seguridad, ya que podrán optar por usa solución open source, adquiriendo las complicaciones del caso, pero añadiendo una optimización de recursos económicos.

- Para alcanzar un óptimo desempeño en el diseño y despliegue de red basado en tecnología VNX, es necesario tener fundamentos de virtualización, ya que varios de los criterios de selección de imágenes (dispositivos de conexión o máquinas virtuales) dependen de la adaptabilidad que se tenga con la plataforma de virtualización.

- Es muy importante revisar la compatibilidad de versionamiento que deben cumplir las herramientas al momento de integrarse, ya que de esto dependerá la configuración y despliegue de cada uno de los módulos de la solución de manera adecuada.

REFERENCIAS BIBLIOGRÁFICAS

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

[2] "**Reference - VNX**". [En línea]. Disponible en: http://web.dit.upm.es/vnxwiki/index.php/Reference. [Consultado: 11-feb-2016].

[3] **Razique Mahroua Robert Locke, George Hacker Forrest Taylor, Scott McBrien Rudolf Kastl, y Wander Boessenkol**, *Red Hat Enterprise Virtualization - Student Worrkbook*. .

[4] "**Glosario de Seguridad**". [En línea]. Disponible en: http://www.symantec.com/es/mx/theme.jsp?themeid=glosario-de-seguridad. [Consultado: 11-feb-2016].

[5] "**Hierarchical Network Design Overview (1.1) > Cisco Networking Academy Connecting Networks Companion Guide: Hierarchical Network Design**". [En línea]. Disponible en: http://www.ciscopress.com/articles/article.asp?p=2202410&seqNum=4. [Consultado: 11-feb-2016].

[6] **"Structuring and Modularizing the Network with Cisco Enterprise Architecture > Network Hierarchy**". [En línea]. Disponible en: http://www.ciscopress.com/articles/article.asp?p=1073230. [Consultado: 11-feb-2016].

[7] **Michael Rash, Orebaugh Angela, y Clark Graham**, *Intrusion prevention and active response: deploying network and host IPS.* Syngress, 2005.

[8] **Fernandez David, Cordero Alejandro, Somavila Jorge, y Rodriguez Jorge**, "Distributed Virtual Scenarios over Multi-host Linux Environments: Virtual Networks over LinuX (VNX)".

[9] **Fernandez David,** "Creación de escenariosde red virtuales distribuidos", Universidad politécnica de madrid, 09-may-2011.

[10] **Mateos Lanchas Verónica,** "Contribución a la automatización de sistemas de respuesta frentea intrusiones mediante ontologías", Universidad Politécnica de Madrid, España, 2013.

[11] **Guamán Loachamín Danny Santiago**, "Propuesta de arquitectura e implementación de un módulo de ejecución de acciones de respuesta para un sistema autónomo de respuesta a intrusiones basado en ontologías", Universidad Politécnica de Madrid, España, 2013.

[12] **Daniel B. Cid,** "Log Analysis using OSSEC", 2007.

[13] **Emilio José Mira Alfaro,** "Implantación de un sistema de detección de intrusos en la universidad de valencia", Universidad de Valencia.

[14] **"Top five free enterprise network intrusion-detection tools**". [En línea]. Disponible en: http://searchsecurity.techtarget.com/tip/Top-five-free-enterprise-network-intrusion-detection-tools. [Consultado: 11-feb-2016].

[15] **Andrew S, Tanenbaum,** *" Redes de Computadoras"*. Pearson Educación,

[16] **Alireza Shameli-Sendi, Naser Ezzati-jivan, Masoume Jabbarifar, and Michel Dagena,** "Intrusion Response Systems : Survey and Taxonomy", *Department of Computer and Software Engineering École Poytechnique de Montréal*.

[17] **Natalia Stakhanova, Samik Basu, y Johnny S. Wong,** "A Taxonomy of Intrusion Response Systems", *Computer Science Technical Reports*.

[18] **Aleksandar Lazarevic,** Vipin Kumar, Jaideep Srivastava, "INTRUSION DETECTION: A SURVEY", *Computer Science Department, University of Minnesota*.

[19] **SAIDI BEN BOUBAKER Ourida**, "Implementation of an Intrusion Detection System",*Computer Science Departement, Hifgh Institute of Management, University of Tunis*, vol. 9, may 2012.

[20] **"IDS/IPS - Tech Newbie".** [En línea]. Disponible en: http://technewbie.weebly.com/idsips.html. [Consultado: 11-feb-2016].

[21] **"Cisco | Blog of Mustafa Saki Unix/Linux Geek | Page 3".** [En línea]. Disponible en: https://unixgeek.wordpress.com/category/cisco/page/3/. [Consultado: 11-feb-2016].

[22] **"Elementos de Snort". [En línea].** Disponible en: http://www.adminso.es/images/d/d0/Pfc_Carlos_cap3.pdf. [Consultado: 11-feb-2016].

[23] **“Introducción a conceptos de IDS y técnicas avanzadas con Snort**”. [En línea]. Disponible en: http://people.baicom.com/agramajo/notes/ids2005.pdf. [Consultado: 11-feb-2016].

[24] **“libvirt”.** [En línea]. Disponible en: https://help.ubuntu.com/lts/serverguide/libvirt.html. [Consultado: 12-feb-2016].

[25]“**KVM/Installation - Community Help Wiki”.** [En línea]. Disponible en: https://help.ubuntu.com/community/KVM/Installation. [Consultado: 12-feb-2016].

[26] **“Virtualization Archives - Natural Born Coder”.** [En línea]. Disponible en: http://www.naturalborncoder.com/category/virtualization/. [Consultado: 12-feb-2016].

[27] **“Virtual technologies”.** [En línea]. Disponible en: http://www.virtual.com.co/. [Consultado: 12-feb-2016].

[28] **“Tecnología de virtualización Intel® (Intel® VT)”.** [En línea]. Disponible en: http://www.intel.es/content/www/es/es/virtualization/virtualization-technology/intel-virtualization-technology.html. [Consultado: 12-feb-2016].

[29] **“AMD Virtualization”.** [En línea]. Disponible en: http://www.amd.com/en-us/solutions/servers/virtualization. [Consultado: 12-feb-2016].

[30] **“Qcow2 - KVM”.** [En línea]. Disponible en: http://www.linux-kvm.org/page/Qcow2. [Consultado: 12-feb-2016].

[31] **“KVM - QEMU”.** [En línea]. Disponible en: http://wiki.qemu.org/KVM. [Consultado: 12-feb-2016].

[32] **“Drivers - KVM**”. [En línea]. Disponible en: http://www.linux-kvm.org/page/WindowsGuestDrivers/Download_Drivers. [Consultado: 12-feb-2016].

[33]**“Introducción y configuración del Spanning Tree Protocol (STP) en los switches Catalyst - Cisco Systems”.** [En línea]. Disponible en: http://www.cisco.com/cisco/web/support/LA/7/73/73037_5.html. [Consultado: 12-feb-2016].

[34] **“VLANs - Cisco”.** [En línea]. Disponible en: http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/12-2SX/configuration/guide/book/vlans.html. [Consultado: 12-feb-2016].

[35] **"Network Infrastructure - Evolved Programmable Network (EPN) - Cisco".** [En línea]. Disponible en: http://www.cisco.com/c/en/us/solutions/service-provider/network-infrastructure/index.html. [Consultado: 12-feb-2016].

[36] **"Cisco Evolved Programmable Network Solution Overview - Cisco".** [En línea]. Disponible en: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/network-infrastructure/solution-overview-c22-731759.html. [Consultado: 12-feb-2016].

[37] **"SDN y NFV protagonizan la transformación de las redes | Telecomunicaciones para Gerentes".** [En línea]. Disponible en: http://www.telecomunicacionesparagerentes.com/sdn-y-nfv-protagonizan-la-transformacion-de-las-redes/. [Consultado: 12-feb-2016].

[38] **"Redes definidas por software (SDN) - Cisco Systems".** [En línea]. Disponible en: http://www.cisco.com/web/LA/soluciones/trends/sdn/index.html. [Consultado: 12-feb-2016].

[39] **"NFV – Network Functions Virtualization - Cisco".** [En línea]. Disponible en: http://www.cisco.com/c/en/us/solutions/service-provider/network-functions-virtualization-nfv/index.html. [Consultado: 12-feb-2016].

[40] **Wander Boessenkoll, Bruce Wolfe, Scote Mcbrien, George Hacker, y Chen Chang,** *Red Hat System Administration II*. Steven Bonneville.

[41] **D. Denning,** *IEEE Transactions on Software Engineering*, vol. 13. 1987.

[42] **"Clasificación de ataques DoS".** [En línea]. Disponible en: https://www.incibe.es/blogs/post/Seguridad/BlogSeguridad/Articulo_y_comentarios/Clasificacion_ataques_DoS. [Consultado: 14-feb-2016].

[43] **"HTG Explains: How the Great Firewall of China Works".** [En línea]. Disponible en: http://www.howtogeek.com/162092/htg-explains-how-the-great-firewall-of-china-works/. [Consultado: 14-feb-2016].

[44] **"DDoS attacks spread to vulnerable IPv6 Internet - CNET".** [En línea]. Disponible en: http://www.cnet.com/news/ddos-attacks-spread-to-vulnerable-ipv6-internet/. [Consultado: 14-feb-2016].

[45] **"IDEaaS para Data Centers: Ataques DoS y DDoS, prevención, detección y mitigación - El blog de Luis Javier Arizmendi Alonso".** [En línea].

Disponible en: http://luisarizmendi.blogspot.com/2014/03/ataques-dos-y-ddos-prevencion-deteccion.html. [Consultado: 14-feb-2016].

[46] **"Martin Roesch | LinkedIn".** [En línea]. Disponible en: https://www.linkedin.com/in/maroesch. [Consultado: 14-feb-2016].

[47] **"NIDS : SNORT / Barnyard2 / MySQL / BASE with Ubuntu 14.04 LTS - Computer Outlines Blog".** [En línea]. Disponible en: http://computer-outlines.over-blog.com/article-nids-snort-barnyard2-apache2-base-with-ubuntu-14-04-lts-123532107.html. [Consultado: 14-feb-2016].

[48] **"Aumentando el rendimiento de Snort con Barnyard2**". [En línea]. Disponible en: http://www.hackplayers.com/2011/03/aumentando-el-rendimiento-de-snort-con.html. [Consultado: 14-feb-2016].

[49] **"About SnortSam".** [En línea]. Disponible en: http://www.snortsam.net/. [Consultado: 18-feb-2016].

[50] **"Security Management White Papers (Management of Security Information, Managing IT Security, Management of Security, IT Security Management, Computer Security Management, Security Administration, Managing Security) Software Downloads, Definition and Webcasts - Bitpipe".** [En línea]. Disponible en: http://www.bitpipe.com/tlist/Security-Management.html. [Consultado: 20-feb-2016].

[51] **"Best practices for network security management | Network World".** [En línea]. Disponible en: http://www.networkworld.com/article/2173927/tech-primers/best-practices-for-network-security-management.html. [Consultado: 20-feb-2016].

[52] **"Technical Overview of DLP (data loss prevention) in Exchange".** [En línea]. Disponible en: https://technet.microsoft.com/en-us/library/jj150527(v=exchg.150).aspx. [Consultado: 21-feb-2016].

[53] **"DVLABS".** [En línea]. Disponible en: http://dvlabs.com/. [Consultado: 21-feb-2016].

[54] **"Zero Day Initiative".** [En línea]. Disponible en: http://www.zerodayinitiative.com/. [Consultado: 21-feb-2016].

[55] **"Technology Research | Gartner Inc."** [En línea]. Disponible en: http://www.gartner.com/technology/home.jsp. [Consultado: 22-feb-2016].

[56] **"Magic Quadrant Research Methodology | Gartner Inc."** [En línea]. Disponible en: http://www.gartner.com/technology/research/methodologies/research_mq.jsp. [Consultado: 22-feb-2016].

[57] **"G1 - Intel anuncia compra da McAfee por US$ 7,68 bilhões - notícias em Tecnologia e Games".** [En línea]. Disponible en: http://g1.globo.com/tecnologia/noticia/2010/08/intel-anuncia-compra-da-mcafee-por-us768-bilhoes-1.html. [Consultado: 22-feb-2016].

[58] **Jonas Tafto Rodfos,** "Comparison of Open Source Network Intrusion Detection Systems",*Network and System Administration Oslo University College*, may 2011.

[59] **"Snort vs Suricata - Aanval Wiki".** [En línea]. Disponible en: http://wiki.aanval.com/wiki/Snort_vs_Suricata. [Consultado: 25-feb-2016].

[60] **"Suricata-vs-snort - aldeid".** [En línea]. Disponible en: https://www.aldeid.com/wiki/Suricata-vs-snort. [Consultado: 25-feb-2016].

[61] Pritika Mehra, **"A brief study and comparison of Snort and Bro Open Source Network Intrusion Detection Systems",** *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 1, núm. 6, Agosto 2012.

[62] **"Samhain Labs | file integrity checkers".** [En línea]. Disponible en: http://www.la-samhna.de/library/scanners.html. [Consultado: 27-feb-2016].

[63] "Anonymous lanza campaña contra Donald Trump", 12-dic-2015.

[64] **Michael Dady**, *Oracle VM Server for Sparc Installatio and configuration*, 1a ed. 2013.

[65] **"Como hacer frente al proceso de ataque completo antes, durante y después",** Disponible en: https://technet.microsoft.com/en-us/library/jj150527(v=exchg.150).aspx. [Consultado: 21-feb-2016].

[66] **Luis Custodio,** "El desafío de invertir en la seguridad informática y añadir valor a la empresa.", Uruguay.

[67] **Cisco, "Sourcefire".** . [En línea]. Disponible en: http://www.cisco.com/c/en/us/products/security/index.html. [Consultado: 16-feb-2016].

[68] **"TrippingPoint Next-Generation IPS".** [En línea]. Disponible en: http://www.trendmicro.com/us/business/network-security/intrusion-prevention-system/. [Consultado: 17-feb-2016].

[69] **K. G. Anagnostakis, , S. Sidiroglou, y , P. Akritidis, K. Xinidis, E. Markatos, A. D. Keromytis,** "Detecting Targeted Attacks Using Shadow Honeypots"

[70] **"The Sagan Log Analysis Engine | Quadrant Information Security".** [En línea]. Disponible en: https://quadrantsec.com/sagan_log_analysis_engine/. [Consultado: 08-mar-2016].

[71] **Surya Bagavan Ambati**, "A brief study and comparison of, open source intrusion detection system tools".

[72] **"Host-Based Detection and Data Loss Prevention Using Open Source Tools",** *SANS Institute InfoSec Reading Room*.

[73] **"Corero | First Line of Defense".** [En línea]. Disponible en: https://www.corero.com/es/cobertura/los-ataques-ddos-utilizados-como-tcnica-de-distraccin-aumentarn-en-2016-/. [Consultado: 12-mar-2016].

[74] **"Principios y filosofía del software libre".** [En línea]. Disponible en: http://www.eveliux.com/mx/Principios-y-filosofia-del-software-libre.html. [Consultado: 13-mar-2016].

[75] **"A single DDoS attack can cost a company more than $400,000".** [En línea]. Disponible en: http://www.kaspersky.com/about/news/business/2015/A-single-DDoS-attack-can-cost-a-company-more-than-400000-dollar. [Consultado: 13-mar-2016].

[76] **Symantec, "INTERNET SECURITY THREAT REPORT",** vol. 20, abr. 2015.

[77] **"The 2015 Magic Quadrant and the 2015 Critical Capabilities for SIEM".** [En línea]. Disponible en: http://www.splunk.com/goto/SIEM_MQ. [Consultado: 13-mar-2016].

[78] **"Entendiendo la importancia de APIs en Cloud Hosting | NESSYS IT".** [En línea]. Disponible en: https://www.nessys.es/entendiendo-la-importancia-de-apis-en-cloud-hosting/. [Consultado: 24-mar-2016].

[79] **"Home - Snort.Org".** [En línea]. Disponible en: https://www.snort.org/. [Consultado: 24-mar-2016].

[80] **"Home — OSSEC".** [En línea]. Disponible en: http://ossec.github.io/. [Consultado: 24-mar-2016].

[81] **"The Bro Network Security Monitor".** [En línea]. Disponible en: https://www.bro.org/. [Consultado: 24-mar-2016].

[82] "**Suricata | Open Source IDS / IPS / NSM engine**". [En línea]. Disponible en: http://suricata-ids.org/. [Consultado: 24-mar-2016].

[83] "**Shadow IDS**". [En línea]. Disponible en: http://www.softpanorama.org/Security/IDS/shadow.shtml. [Consultado: 24-mar-2016].

[84] "**fwsnort - iptables Intrusion Detection with String Matching and Snort Rules**". [En línea]. Disponible en: http://cipherdyne.org/fwsnort/. [Consultado: 24-mar-2016].

[85] **"Fail2ban".** [En línea]. Disponible en: http://www.fail2ban.org/wiki/index.php/Main_Page. [Consultado: 24-mar-2016].

[86] **"the squertproject".** [En línea]. Disponible en: http://www.squertproject.org/. [Consultado: 24-mar-2016].

[87] "**Security Onion".** [En línea]. Disponible en: https://security-onion-solutions.github.io/security-onion/. [Consultado: 24-mar-2016].

[88] **"Symmetrix Tech - North Texas IT Support and Office 365 Migrations".** [En línea]. Disponible en: http://symmetrixtech.com/. [Consultado: 24-mar-2016].

[89] **"Sguil - Open Source Network Security Monitoring**". [En línea]. Disponible en: http://bammv.github.io/sguil/index.html. [Consultado: 24-mar-2016].

[90] "**Aanval ® - Snort, Suricata, and Syslog Intrusion Detection, Correlation and Threat Management".** [En línea]. Disponible en: https://www.aanval.com/. [Consultado: 24-mar-2016].

[91] **"Prelude | Products".** [En línea]. Disponible en: http://www.prelude-siem.com/index.php/en/. [Consultado: 24-mar-2016].

[92] **"Snort-Personalizando las reglas".** [En línea]. Disponible en: http://www.adminso.es/index.php/Snort-Personalizando_las_reglas. [Consultado: 24-mar-2016].

[93] **“GitHub - llaera/slowloris.pl: A new DOS Perl Programm”.** [En línea]. Disponible en: https://github.com/llaera/slowloris.pl. [Consultado: 24-mar-2016].

[94] “**Caracteristicas del Software Propietario ”.** [En línea]. Disponible en: https://sites.google.com/site/jachsistemascomputacionales/classroom-news/21-caracteristicas-del-software-propietario [Consultado: 24-mar-2016].

ANEXOS

ANEXO A: Diagrama de red con topología jerárquica, archivo XML

ANEXO B: Archivo de configuración de Snort

ANEXO C: Archivo de configuración de Ossec

ANEXO D: Archivo de configuración de reglas de Snort

ANEXO A

Diagrama de red con topología jerárquica.

```
<?xml version="1.0" encoding="UTF-8"?>

<!--*****************************************************************************
                ESCUELA POLITECNICA NACIONAL
          INGENIERÍA ELECTRÓNICA Y REDES DE LA INFORMACIÓN
          ESCENARIO DE RED CON TOPOLOGÍA JERÁRQUICA

        Autor:                  ROBERTO CÁRDENAS
        Director de proyecto:   DANNY GUAMÁN
        Fecha:                  2016/03/16

*****************************************************************************--
>
<!--_____________________________________________________________________-->
 <vnx xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="/usr/share/xml/vnx/vnx-2.00.xsd">

  <!--DEFINICIONES GLOBALES:
      Nombre del escenario.
      Etiquetas globales.
      Especificación del archivo con imágenes Cisco-->

 <global>
  <version>2.0</version>
  <scenario_name>Red_Topologia_Jerarquica</scenario_name>
  <automac/><!--generación de direcciones mac automáticas para cada virtual
machine-->
  <vm_mgmt type="none"/><!--la gestión de cada vm es dedicada no general-->
  <vm_defaults>
     <console id="0" display="no"/><!--No se despliega la interfaz gŕafica-->
     <console id="1" display="yes"/><!--Se despliega una consola-->
  </vm_defaults>
  <dynamips_ext>ciscoall-dn.xml</dynamips_ext><!--Archivo xml que contiene la
funcionalidad de las imágenes dynamips Cisco que se utilizan en el escenario-->
 </global>


<!--_____________________________________________________________________-->
 <!--REDES: Presentes en la arquitectura  -->
 <net name="Net0" mode="virtual_bridge" /><!--RED EXTERNA-->
 <net name="Net1" mode="virtual_bridge" /><!--RED INTERNA-->
 <net name="Net2" mode="virtual_bridge" /><!--VNX -  HOST-FISICO-->
```

```
 <net name="Net3" mode="virtual_bridge" /><!--VLAN 1-->
 <net name="Net4" mode="virtual_bridge" /><!--VLAN 2-->
 <net name="br1" mode="virtual_bridge" managed="no"/><!--RED DE CONEXIÓN A
INTERNET-->

<!--_____-->
<!--HOST: Programación de Host involucrados en la solución
      Dentro de las características escenciales deberán constar
      Nombre, Asignación de Recursos, IP Address; Gateway ,
      Sistema operativo (indicar arquitectura 64 bits), Permisos y comandos
          por defecto
          Para el caso de dispositivos de red, indicar las interfaces de red
          y gateways a utilizar-->

<!--_____HOSTs DE RED EXTERNA_____-->
  <vm name="kali_e" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_kali</filesystem>
  <mem>128M</mem>
  <console id="0" display="no"/><!--INTERFAZ GRÁFICA-->
  <console id="1" display="yes"/><!--CONSOLA-->
  <if id="1" net="Net0"><!--PERTENECIENTE A RED EXTERNA-->
   <ipv4>10.1.0.2/24</ipv4>
  </if>
  <route type="ipv4" gw="10.1.0.1">default</route><!--GATEWAY INTERFAZ DE
ROUTER-->
  </vm>
<!--_____-->


<!---_____ROUTER CISCO c3640  "GATEWAY"_____--
>
  <vm name="rgw" type="dynamips" subtype="3600" os="">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
  <mem>96M</mem>
  <if id="1" net="Net0" name="fa 1/0"><!--GW DE RED EXTERNAi conexión con
KALI-->
    <ipv4>10.1.0.1/24</ipv4>
  </if>
  <if id="2" net="Net1" name="fa 1/1"><!--GW RED LAN-->
   <ipv4>10.1.1.1/24</ipv4>
  </if>
  <if id="3" net="Net2" name="fa 1/2"><!--GW A EQUIPO LOCAL-->
   <ipv4>10.1.2.1/24</ipv4>
  </if>
  <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscorgw.conf</exec>
```

```
   <route type="ipv4" gw="10.1.1.2">10.1.0.0/16</route><!--SIGUIENTE SALTO A
FW-->
  </vm>
<!--_____-->

<!--_____RED LAN _____  _____-->
<!--FIREWALL DE LA RED (Server Linux 14.04) "3 INTERFACES DE RED"-->
  <vm name="firewall" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_firewall</filesystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI en este caso sonsola-->
   <console id="1" display="yes"/>
   <if id="1" net="Net1"><!--CONEXIÓN CON ROUTER CISCO-->
    <ipv4>10.1.1.2/24</ipv4>
   </if>
   <if id="2" net="Net1"><!--GW PARA RED DMZ-->
    <ipv4>10.1.1.3/24</ipv4>
   </if>
   <if id="3" net="Net1"><!--GW PARA RED LAN-->
    <ipv4>10.1.1.6/24</ipv4>
   </if>
   <if id="4" net="Net1"><!--CONEXIÓN CON SNORT1 (protección contra ataques
externos)-->
   <ipv4>10.1.1.4/24</ipv4>
   </if>
   <route type="ipv4" gw="10.1.1.1">default</route><!--SIGUIENTE SALTO A
ROUTER DE BORDE-->
   <route type="ipv4" gw="10.1.3.5">default</route>
   <route type="ipv4" gw="10.1.4.5">default</route>
   <route type="ipv4" gw="10.1.3.6">default</route>
   <route type="ipv4" gw="10.1.4.6">default</route>
   <route type="ipv4" gw="10.1.1.7">default</route>
   <forwarding type="ip" />

  </vm>
<!--_____-->


<!--_____ROUTER CISCO c3640  "CORE 1"_____-->
   <vm name="core1" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net1" name="fa 1/0"><!--GATEWAY PARA GRANJA DE
SERVIDORES-->
    <ipv4>10.1.1.7/24</ipv4>
   </if>
```

```
   <if id="2" net="Net4" name="fa 1/1">
    <ipv4>10.1.4.5/24</ipv4>
   </if>
   <if id="3" net="Net3" name="fa 1/2">
    <ipv4>10.1.3.5/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscocore1.conf</exec>
   <route type="ipv4" gw="10.1.1.6">10.1.0.0/16</route><!--SIGUIENTE SALTO A
FW-->
   <!--route type="ipv4" gw="10.1.1.3">10.1.0.0/16</route-->
   <!--route type="ipv4" gw="10.1.4.5">10.1.0.0/16</route-->
 </vm>
<!--_____-->

<!--_____ROUTER CISCO c3640  "CORE 2"_____-->
   <vm name="core2" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net1" name="fa 1/0"><!--GATEWAY PARA GRANJA DE
SERVIDORES-->
    <ipv4>10.1.1.8/24</ipv4>
   </if>
   <if id="2" net="Net4" name="fa 1/1">
    <ipv4>10.1.4.6/24</ipv4>
   </if>
   <if id="3" net="Net3" name="fa 1/2">
    <ipv4>10.1.3.6/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscocore2.conf</exec>
   <route type="ipv4" gw="10.1.1.6">10.1.0.0/16</route><!--SIGUIENTE SALTO A
FW-->
   <!--route type="ipv4" gw="10.1.1.3">10.1.0.0/16</route-->
   <!--route type="ipv4" gw="10.1.4.5">10.1.0.0/16</route-->
 </vm>
<!--_____-->

<!--_____ROUTER CISCO c3640  "DISTRIBUCIÓN 1 "_____--
>
   <vm name="dist1" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net4" name="fa 1/1"><!--CONEXIÓN CORE 1-->
    <ipv4>10.1.4.3/24</ipv4>
   </if>
```

```
   <if id="2" net="Net3" name="fa 1/2"><!--CONEXIÓN CORE 1-->
    <ipv4>10.1.3.3/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscodist1.conf</exec>
   <route type="ipv4" gw="10.1.4.5">default</route>
   <route type="ipv4" gw="10.1.3.6">default</route>
   <route type="ipv4" gw="10.1.1.7">default</route>
   <route type="ipv4" gw="10.1.1.8">default</route>
 </vm>
<!--_____________________________________________________________________-->


<!--_____________________ROUTER CISCO c3640  "DISTRIBUCIÓN 2 "_____________________--
>
   <vm name="dist2" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net4" name="fa 1/1"><!--CONEXIÓN CORE 1-->
    <ipv4>10.1.4.4/24</ipv4>
   </if>
   <if id="2" net="Net3" name="fa 1/2"><!--CONEXIÓN CORE 1-->
    <ipv4>10.1.3.4/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscodist2.conf</exec>
   <route type="ipv4" gw="10.1.3.5">default</route>
   <route type="ipv4" gw="10.1.4.6">default</route>
   <route type="ipv4" gw="10.1.1.7">default</route>
   <route type="ipv4" gw="10.1.1.8">default</route>
 </vm>
<!--_____________________________________________________________________-->


<!--_____________________ROUTER CISCO c3640  "ACCESO 1 "_____________________-->
   <vm name="acce1" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net3" name="fa 1/1"><!--CONEXIÓN DISTRIBUCIÓN 1-->
    <ipv4>10.1.3.1/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscoacce1.conf</exec>
   <route type="ipv4" gw="10.1.4.3">default</route><!--SIGUIENTE SALTO A FW-->
   <route type="ipv4" gw="10.1.3.4">default</route>
 </vm>
```

```
<!--_____________________________________________________________________________-->

<!--___________________ROUTER CISCO c3640  "ACCESO 2 "___________________-->
   <vm name="acce2" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net4" name="fa 1/1"><!--CONEXIÓN DISTRIBUCIÓN 1-->
    <ipv4>10.1.4.1/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscoacce2.conf</exec>
   <route type="ipv4" gw="10.1.4.3">default</route><!--SIGUIENTE SALTO A FW-->
   <route type="ipv4" gw="10.1.3.4">default</route>
 </vm>
<!--_____________________________________________________________________________-->

<!--___________________ROUTER CISCO c3640  "ACCESO 3 "___________________-->
   <vm name="acce3" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net3" name="fa 1/1"><!--CONEXIÓN DISTRIBUCIÓN 1-->
    <ipv4>10.1.3.2/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscoacce3.conf</exec>
   <route type="ipv4" gw="10.1.3.3">default</route><!--SIGUIENTE SALTO A FW-->
   <route type="ipv4" gw="10.1.4.4">default</route>
 </vm>
<!--_____________________________________________________________________________-->

<!--___________________ROUTER CISCO c3640  "ACCESO 4 "___________________-->
   <vm name="acce4" type="dynamips" subtype="3600" os="">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Cisco/c3640</filesystem>
   <mem>96M</mem>
   <if id="1" net="Net4" name="fa 1/1"><!--CONEXIÓN DISTRIBUCIÓN 1-->
    <ipv4>10.1.4.2/24</ipv4>
   </if>
   <exec seq="loadcfg"  type="verbatim"
ostype="load">confvlan/ciscoacce4.conf</exec>
   <route type="ipv4" gw="10.1.3.3">default</route><!--SIGUIENTE SALTO A FW-->
   <route type="ipv4" gw="10.1.4.4">default</route>
 </vm>
<!--_____________________________________________________________________________-->

<!--_____________________________S-N-O-R-T_____________________________________-->
```

```
<!--_____SNORT 1 (PROTECCION EXTERNA)_____-->
<vm name="snort1" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_snort</filesystem>
  <mem>512M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4>10.1.1.10</ipv4><!--IP RED DMZ-->
  </if>
  <route type="ipv4" gw="10.1.1.4">default</route><!--GW RED DMZ-->
  <forwarding type="ip" />
 </vm>
 <!--_____-->


<!--_____SNORT 2 (PROTECCION DMZ)_____-->
<vm name="snort2" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_snort</filesystem>
  <mem>512M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4>10.1.1.30</ipv4><!--IP RED DMZ-->
  </if>
  <if id="2" net="Net1">
   <ipv4>10.1.1.31</ipv4><!--IP RED DMZ-->
  </if>
  <route type="ipv4" gw="10.1.1.3">default</route><!--GW RED DMZ-->
  <forwarding type="ip" />
 </vm>
 <!--_____-->


<!--_____SNORT 3 (PROTECCION GRANJA DE
SERVIDORES)_____-->
<vm name="snort3" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_snort</filesystem>
  <mem>512M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4>10.1.1.20</ipv4><!--IP RED DMZ-->
  </if>
  <if id="2" net="Net1">
```

```
    <ipv4>10.1.1.21</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.1.7">default</route><!--GW RED DMZ-->
   <forwarding type="ip" />
  </vm>
 <!--_____________________________________________________________________________-->


<!--_____________________________M-E-T-A-S-P-L-O-I-T-A-B-L-E_____________________-
->
<!--_____________________________METASPLOITABLE 1_____________________________-
->
  <vm name="metas1" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI-->
   <console id="1" display="yes"/><!--Consola apagada-->
   <if id="1" net="Net3">
    <ipv4 mask="255.255.255.0">10.1.3.7</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.3.1">default</route><!--GW RED DMZ-->
  </vm>
 <!--_____________________________________________________________________________-->

<!--_____________________________METASPLOITABLE 2_____________________________-
->
  <vm name="metas2" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI-->
   <console id="1" display="yes"/><!--Consola apagada-->
   <if id="1" net="Net4">
    <ipv4 mask="255.255.255.0">10.1.4.7</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.4.1">default</route><!--GW RED DMZ-->
  </vm>
 <!--_____________________________________________________________________________-->


<!--_____________________________METASPLOITABLE 3_____________________________-
->
  <vm name="metas3" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
```

```
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI-->
   <console id="1" display="yes"/><!--Consola apagada-->
   <if id="1" net="Net3">
    <ipv4 mask="255.255.255.0">10.1.3.9</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.3.2">default</route><!--GW RED DMZ-->
 </vm>
 <!--_____-->


<!--_____METASPLOITABLE  4_____-
->
  <vm name="metas4" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI-->
   <console id="1" display="yes"/><!--Consola apagada-->
   <if id="1" net="Net4">
    <ipv4 mask="255.255.255.0">10.1.4.8</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.4.2">default</route><!--GW RED DMZ-->
 </vm>
 <!--_____-->


<!--_____METASPLOITABLE
DMZ_____-->
  <vm name="metasdmz" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
   <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
   <mem>128M</mem>
   <console id="0" display="no"/><!--INTERFAZ GUI-->
   <console id="1" display="yes"/><!--Consola apagada-->
   <if id="1" net="Net1">
    <ipv4 mask="255.255.255.0">10.1.1.9</ipv4><!--IP RED DMZ-->
   </if>
   <route type="ipv4" gw="10.1.1.31">default</route><!--GW RED DMZ-->
 </vm>
 <!--_____-->
```

```
<!--_____METASPLOITABLE  GRANJA DE
SERVIDORES_____-->
 <vm name="metasgs" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_metasploitable</files
ystem>
  <mem>128M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4 mask="255.255.255.0">10.1.1.14</ipv4><!--IP RED DMZ-->
  </if>
  <route type="ipv4" gw="10.1.1.21">default</route><!--GW RED DMZ-->
 </vm>
<!--_____-->


<!--_____O-S-S-E-C_____-->
<!--_____OSSEC GRANJA DE
SERVIDORES_____-->
 <vm name="ossecserver1" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_ossecs</filesystem>
  <mem>512M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4 mask="255.255.255.0">10.1.1.15</ipv4><!--IP RED DMZ-->
  </if>
  <route type="ipv4" gw="10.1.1.21">default</route><!--GW RED DMZ-->
 </vm>
<!--_____-->


<!--_____R-E-P-O-R-T-E-R-I-A_____-->
 <vm name="ossecserver" type="libvirt" subtype="kvm" os="linux" arch="x86_64">
  <filesystem
type="cow">/home/cardenas/Desktop/EPN/Imágenes/rootfs_xampp</filesystem>
  <mem>512M</mem>
  <console id="0" display="no"/><!--INTERFAZ GUI-->
  <console id="1" display="yes"/><!--Consola apagada-->
  <if id="1" net="Net1">
   <ipv4 mask="255.255.255.0">10.1.1.13</ipv4><!--IP RED DMZ-->
  </if>
  <route type="ipv4" gw="10.1.1.21">default</route><!--GW RED DMZ-->
 </vm>
<!--_____-->
```

```
<!--_____H-O-S-T_____-->
 <host>
  <hostif net="Net2"><!--RED DE HOST FÍSICO-->
    <ipv4>10.1.2.2/24</ipv4>
  </hostif>
  <route type="ipv4" gw="10.1.2.1">10.1.0.0/16</route><!--GW INTERFAZ DE
ROUTER-->
 </host>

</vnx>
```

ANEXO B

Archivo de configuración de Snort.

```
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#    http://www.snort.org                Snort Website
#    http://vrt-blog.snort.org/    Sourcefire VRT Blog
#
#    Mailing list Contact:      snort-sigs@lists.sourceforge.net
#    False Positive reports:    fp@sourcefire.com
#    Snort bugs:                bugs@snort.org
#
#    Compatible with Snort Versions:
#    VERSIONS : 2.9.7.x
#
#    Snort build options:
#    OPTIONS : --enable-gre --enable-mpls --enable-targetbased --enable-ppm --
enable-perfprofiling --enable-zlib --enable-active-response --enable-normalizer --
enable-reload --enable-react --enable-flexresp3
#
#    Additional information:
#    This configuration file enables active response, to run snort in
#    test mode -T you are required to supply an interface -i <interface>
#    or test mode will fail to fully validate the configuration and
#    exit with a FATAL error
#--------------------------------------------------

######################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
######################################################

######################################################
```

```
# Step #1: Set the network variables.  For more information, see README.variables
#####################################################

# Setup the network addresses you are protecting

#ipvar HOME_NET 192.168.122.0/24
#ipvar HOME_NET 192.168.122.181
ipvar DMZ_NET 10.1.5.0/24
ipvar HOME_NET 10.1.1.0/24
ipvar EXT_NET 10.1.0.0/16


# Set up the external network addresses. Leave as "any" in most situations
#ipvar EXTERNAL_NET any
ipvar EXTERNAL_NET !$HOME_NET

#SnortSam Config Line ? Maybe ? idk
#output alert_fwsam: 192.168.122.249:898/password
#output alert_fwsam: 10.1.2.1:898/password

# List of DNS servers on your network
#ipvar DNS_SERVERS $HOME_NET
#ipvar DNS_SERVERS 192.168.122.1

# List of SMTP servers on your network
#ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
#ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
#ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
#ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
#ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
#ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
#ipvar SIP_SERVERS $HOME_NET

# List of ports you run web servers on
portvar HTTP_PORTS
[80,81,311,383,591,593,901,1220,1414,1741,1830,2301,2381,2809,3037,3128,3702,
```

```
4343,4848,5250,6988,7000,7001,7144,7145,7510,7777,7779,8000,8008,8014,8028,
8080,8085,8088,8090,8118,8123,8180,8181,8243,8280,8300,8800,8888,8899,9000,
9060,9080,9090,9091,9443,9999,11371,34443,34444,41080,50002,55555]

# List of ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

# List of ports you might see oracle attacks on
portvar ORACLE_PORTS 1024:

# List of ports you want to look for SSH connections on:
portvar SSH_PORTS 22

# List of ports you run ftp servers on
portvar FTP_PORTS [21,2100,3535]

# List of ports you run SIP servers on
portvar SIP_PORTS [5060,5061,5600]

# List of file data ports for file inspection
portvar FILE_DATA_PORTS [$HTTP_PORTS,110,143]

# List of GTP ports for GTP preprocessor
portvar GTP_PORTS [2123,2152,3386]

# other variables, these should not be modified
ipvar AIM_SERVERS
[64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.18
8.3.0/24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.
179.0/24,205.188.248.0/24]

# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH so_/etc/snort/rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules

# If you are using reputation preprocessor set these
# Currently there is a bug with relative paths, they are relative to where snort is
# not relative to snort.conf like the above variables
# This is completely inconsistent with how other vars work, BUG 89986
# Set the absolute path appropriately
var WHITE_LIST_PATH /etc/snort/rules
var BLACK_LIST_PATH /etc/snort/rules

##################################################
# Step #2: Configure the decoder.  For more information, see README.decode
```

```
##################################################

# Stop generic decode events:
config disable_decode_alerts

# Stop Alerts on experimental TCP options
config disable_tcpopt_experimental_alerts

# Stop Alerts on obsolete TCP options
config disable_tcpopt_obsolete_alerts

# Stop Alerts on T/TCP alerts
config disable_tcpopt_ttcp_alerts

# Stop Alerts on all other TCPOption type events:
config disable_tcpopt_alerts

# Stop Alerts on invalid ip options
config disable_ipopt_alerts

# Alert if value in length field (IP, TCP, UDP) is greater th elength of the packet
# config enable_decode_oversized_alerts

# Same as above, but drop packet if in Inline mode (requires
enable_decode_oversized_alerts)
# config enable_decode_oversized_drops

# Configure IP / TCP checksum mode
config checksum_mode: all

# Configure maximum number of flowbit references.  For more information, see
README.flowbits
# config flowbits_size: 64

# Configure ports to ignore
# config ignore_ports: tcp 21 6667:6671 1356
# config ignore_ports: udp 1:17 53

# Configure active response for non inline operation. For more information, see
REAMDE.active
# config response: eth0 attempts 2

# Configure DAQ related options for inline operation. For more information, see
README.daq
#
# config daq: <type>
# config daq_dir: <dir>
# config daq_mode: <mode>
```

```
# config daq_var: <var>
#
# <type> ::= pcap | afpacket | dump | nfq | ipq | ipfw
# <mode> ::= read-file | passive | inline
# <var> ::= arbitrary <name>=<value passed to DAQ
# <dir> ::= path as to where to look for DAQ module so's

# Configure specific UID and GID to run snort as after dropping privs. For more
information see snort -h command line options
#
# config set_gid:
# config set_uid:

# Configure default snaplen. Snort defaults to MTU of in use interface. For more
information see README
#
# config snaplen:
#

# Configure default bpf_file to use for filtering what traffic reaches snort. For more
information see snort -h command line options (-F)
#
# config bpf_file:
#

# Configure default log directory for snort to log to.  For more information see snort -
h command line options (-l)
#
# config logdir:


##################################################
# Step #3: Configure the base detection engine.  For more information, see
README.decode
##################################################

# Configure PCRE match limitations
config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500

# Configure the detection engine  See the Snort Manual, Configuring Snort - Includes -
Config
config detection: search-method ac-split search-optimize max-pattern-len 20

# Configure the event queue.  For more information, see README.event_queue
config event_queue: max_queue 8 log 5 order_events content_length

##################################################
```

```
## Configure GTP if it is to be used.
## For more information, see README.GTP
##################################################

# config enable_gtp

##################################################
# Per packet and rule latency enforcement
# For more information see README.ppm
##################################################

# Per Packet latency configuration
#config ppm: max-pkt-time 250, \
#   fastpath-expensive-packets, \
#   pkt-log

# Per Rule latency configuration
#config ppm: max-rule-time 200, \
#   threshold 3, \
#   suspend-expensive-rules, \
#   suspend-timeout 20, \
#   rule-log alert

##################################################
# Configure Perf Profiling for debugging
# For more information see README.PerfProfiling
##################################################

#config profile_rules: print all, sort avg_ticks
#config profile_preprocs: print all, sort avg_ticks

##################################################
# Configure protocol aware flushing
# For more information see README.stream5
##################################################
config paf_max: 16000

##################################################
# Step #4: Configure dynamic loaded libraries.
# For more information, see Snort Manual, Configuring Snort - Dynamic Modules
##################################################

# path to dynamic preprocessor libraries
dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/

# path to base preprocessor engine
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
```

```
# path to dynamic rules libraries
dynamicdetection directory /usr/local/lib/snort_dynamicrules

##################################################
# Step #5: Configure preprocessors
# For more information, see the Snort Manual, Configuring Snort - Preprocessors
##################################################

# GTP Control Channle Preprocessor. For more information, see README.GTP
# preprocessor gtp: ports { 2123 3386 2152 }

# Inline packet normalization. For more information, see README.normalize
# Does nothing in IDS mode
preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6

# Target-based IP defragmentation.  For more inforation, see README.frag3
#Cambio realizado para detectar ataques DoS
#preprocessor frag3_global: max_frags 65536
preprocessor frag3_global: prealloc_frags 8192
#preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10
min_fragment_length 100 timeout 180
preprocessor frag3_engine: policy linux detect_anomalies overlap_limit 1
min_fragment_length 5 timeout 1 bind_to 10.1.5.4

# Target-Based stateful inspection/stream reassembly.  For more inforation, see
README.stream5
preprocessor stream5_global: track_tcp yes, \
  track_udp yes, \
  track_icmp no, \
  max_tcp 262144, \
  memcap 1073741824, \
  max_udp 131072, \
  prune_log_max 1073741824, \
  max_active_responses 2, \
  min_response_seconds 5
preprocessor stream5_tcp: policy windows, detect_anomalies, require_3whs 180, \
  overlap_limit 10, small_segments 3 bytes 150, timeout 180, max_queued_bytes
90485760, max_queued_segs 40485760, \
   ports client 21 22 23 25 42 53 79 109 110 111 113 119 135 136 137 139 143 \
      161 445 513 514 587 593 691 1433 1521 1741 2100 3306 6070 6665 6666
6667 6668 6669 \
      7000 8181 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779,
\
```

```
   ports both 80 81 311 383 443 465 563 591 593 636 901 989 992 993 994 995
1220 1414 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7907 7000
7001 7144 7145 7510 7802 7777 7779 \
      7801 7900 7901 7902 7903 7904 7905 7906 7908 7909 7910 7911 7912 7913
7914 7915 7916 \
      7917 7918 7919 7920 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123
8180 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090 9091 9443 9999
11371 34443 34444 41080 50002 55555
preprocessor stream5_udp: timeout 180

# performance statistics.  For more information, see the Snort Manual, Configuring
Snort - Preprocessors - Performance Monitor
# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt 10000

# HTTP normalization and anomaly detection.  For more information, see
README.http_inspect
preprocessor http_inspect: global iis_unicode_map unicode.map 1252
compress_depth 65535 decompress_depth 65535
preprocessor http_inspect_server: server default \
   http_methods { GET POST PUT SEARCH MKCOL COPY MOVE LOCK UNLOCK
NOTIFY POLL BCOPY BDELETE BMOVE LINK UNLINK OPTIONS HEAD DELETE
TRACE TRACK CONNECT SOURCE SUBSCRIBE UNSUBSCRIBE PROPFIND
PROPPATCH BPROPFIND BPROPPATCH RPC_CONNECT PROXY_SUCCESS BITS_POST
CCM_POST SMS_POST RPC_IN_DATA RPC_OUT_DATA RPC_ECHO_DATA } \
   chunk_length 500000 \
   server_flow_depth 0 \
   client_flow_depth 0 \
   post_depth 65495 \
   oversize_dir_length 500 \
   max_header_length 750 \
   max_headers 100 \
   max_spaces 200 \
   small_chunk_length { 10 5 } \
   ports { 80 81 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037
3128 3702 4343 4848 5250 6988 7000 7001 7144 7145 7510 7777 7779 8000 8008
8014 8028 8080 8085 8088 8090 8118 8123 8180 8181 8243 8280 8300 8800 8888
8899 9000 9060 9080 9090 9091 9443 9999 11371 34443 34444 41080 50002
55555 } \
   non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \
   enable_cookie \
   extended_response_inspection \
   inspect_gzip \
   normalize_utf \
   unlimited_decompress \
   normalize_javascript \
   apache_whitespace no \
   ascii no \
   bare_byte no \
```

```
   directory no \
   double_decode no \
   iis_backslash no \
   iis_delimiter no \
   iis_unicode no \
   multi_slash no \
   utf_8 no \
   u_encode yes \
   webroot no

# ONC-RPC normalization and anomaly detection.  For more information, see the
Snort Manual, Configuring Snort - Preprocessors - RPC Decode
preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776
32777 32778 32779 no_alert_multiple_requests no_alert_large_fragments
no_alert_incomplete

# Back Orifice detection.
preprocessor bo

# FTP / Telnet normalization and anomaly detection.  For more information, see
README.ftptelnet
preprocessor ftp_telnet: global inspection_type stateful encrypted_traffic no
check_encrypted
preprocessor ftp_telnet_protocol: telnet \
   ayt_attack_thresh 20 \
   normalize ports { 23 } \
   detect_anomalies
preprocessor ftp_telnet_protocol: ftp server default \
   def_max_param_len 100 \
   ports { 21 2100 3535 } \
   telnet_cmds yes \
   ignore_telnet_erase_cmds yes \
   ftp_cmds { ABOR ACCT ADAT ALLO APPE AUTH CCC CDUP } \
   ftp_cmds { CEL CLNT CMD CONF CWD DELE ENC EPRT } \
   ftp_cmds { EPSV ESTA ESTP FEAT HELP LANG LIST LPRT } \
   ftp_cmds { LPSV MACB MAIL MDTM MIC MKD MLSD MLST } \
   ftp_cmds { MODE NLST NOOP OPTS PASS PASV PBSZ PORT } \
   ftp_cmds { PROT PWD QUIT REIN REST RETR RMD RNFR } \
   ftp_cmds { RNTO SDUP SITE SIZE SMNT STAT STOR STOU } \
   ftp_cmds { STRU SYST TEST TYPE USER XCUP XCRC XCWD } \
   ftp_cmds { XMAS XMD5 XMKD XPWD XRCP XRMD XRSQ XSEM } \
   ftp_cmds { XSEN XSHA1 XSHA256 } \
   alt_max_param_len 0 { ABOR CCC CDUP ESTA FEAT LPSV NOOP PASV PWD QUIT
REIN STOU SYST XCUP XPWD } \
   alt_max_param_len 200 { ALLO APPE CMD HELP NLST RETR RNFR STOR STOU
XMKD } \
   alt_max_param_len 256 { CWD RNTO } \
   alt_max_param_len 400 { PORT } \
```

```
   alt_max_param_len 512 { SIZE } \
   chk_str_fmt { ACCT ADAT ALLO APPE AUTH CEL CLNT CMD } \
   chk_str_fmt { CONF CWD DELE ENC EPRT EPSV ESTP HELP } \
   chk_str_fmt { LANG LIST LPRT MACB MAIL MDTM MIC MKD } \
   chk_str_fmt { MLSD MLST MODE NLST OPTS PASS PBSZ PORT } \
   chk_str_fmt { PROT REST RETR RMD RNFR RNTO SDUP SITE } \
   chk_str_fmt { SIZE SMNT STAT STOR STRU TEST TYPE USER } \
   chk_str_fmt { XCRC XCWD XMAS XMD5 XMKD XRCP XRMD XRSQ } \
   chk_str_fmt { XSEM XSEN XSHA1 XSHA256 } \
   cmd_validity ALLO < int [ char R int ] > \
   cmd_validity EPSV < [ { char 12 | char A char L char L } ] > \
   cmd_validity MACB < string > \
   cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
   cmd_validity MODE < char ASBCZ > \
   cmd_validity PORT < host_port > \
   cmd_validity PROT < char CSEP > \
   cmd_validity STRU < char FRPO [ string ] > \
   cmd_validity TYPE < { char AE [ char NTC ] | char I | char L [ number ] } >
preprocessor ftp_telnet_protocol: ftp client default \
   max_resp_len 256 \
   bounce yes \
   ignore_telnet_erase_cmds yes \
   telnet_cmds yes


# SMTP normalization and anomaly detection.  For more information, see
README.SMTP
preprocessor smtp: ports { 25 465 587 691 } \
   inspection_type stateful \
   b64_decode_depth 0 \
   qp_decode_depth 0 \
   bitenc_decode_depth 0 \
   uu_decode_depth 0 \
   log_mailfrom \
   log_rcptto \
   log_filename \
   log_email_hdrs \
   normalize cmds \
   normalize_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM
ESND ESOM ETRN EVFY } \
   normalize_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT
RSET SAML SEND SOML } \
   normalize_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-
DRCP X-ERCP X-EXCH50 } \
   normalize_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN
XLICENSE XQUE XSTA XTRN XUSR } \
   max_command_line_len 512 \
   max_header_line_len 1000 \
```

```
   max_response_line_len 512 \
   alt_max_command_line_len 260 { MAIL } \
   alt_max_command_line_len 300 { RCPT } \
   alt_max_command_line_len 500 { HELP HELO ETRN EHLO } \
   alt_max_command_line_len 255 { EXPN VRFY ATRN SIZE BDAT DEBUG EMAL ESAM
ESND ESOM EVFY IDENT NOOP RSET } \
   alt_max_command_line_len 246 { SEND SAML SOML AUTH TURN ETRN DATA RSET
QUIT ONEX QUEU STARTTLS TICK TIME TURNME VERB X-EXPS X-LINK2STATE
XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
   valid_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND
ESOM ETRN EVFY } \
   valid_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET
SAML SEND SOML } \
   valid_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-
ERCP X-EXCH50 } \
   valid_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE
XQUE XSTA XTRN XUSR } \
   xlink2state { enabled }

# Portscan detection.  For more information, see README.sfportscan
# preprocessor sfportscan: proto  { all } memcap { 10000000 } sense_level { low }

# ARP spoof detection.  For more information, see the Snort Manual - Configuring
Snort - Preprocessors - ARP Spoof Preprocessor
#
preprocessor arpspoof
# preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00

# SSH anomaly detection.  For more information, see README.ssh
preprocessor ssh: server_ports { 22 } \
                autodetect \
                max_client_bytes 19600 \
                max_encrypted_packets 20 \
                max_server_version_len 100 \
                enable_respoverflow enable_ssh1crc32 \
                enable_srvoverflow enable_protomismatch

# SMB / DCE-RPC normalization and anomaly detection.  For more information, see
README.dcerpc2
preprocessor dcerpc2: memcap 102400, events [co ]
preprocessor dcerpc2_server: default, policy WinXP, \
   detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server 593], \
   autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
   smb_max_chain 3, smb_invalid_shares ["C$", "D$", "ADMIN$"]

# DNS anomaly detection.  For more information, see README.dns
preprocessor dns: ports { 53 } enable_rdata_overflow
```

```
# SSL anomaly detection and traffic bypass.  For more information, see README.ssl
preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 7801 7802 7900
7901 7902 7903 7904 7905 7906 7907 7908 7909 7910 7911 7912 7913 7914 7915
7916 7917 7918 7919 7920 }, trustservers, noinspect_encrypted

# SDF sensitive data preprocessor.  For more information see README.sensitive_data
preprocessor sensitive_data: alert_threshold 25

# SIP Session Initiation Protocol preprocessor.  For more information see
README.sip
preprocessor sip: max_sessions 40000, \
  ports { 5060 5061 5600 }, \
  methods { invite \
            cancel \
            ack \
            bye \
            register \
            options \
            refer \
            subscribe \
            update \
            join \
            info \
            message \
            notify \
            benotify \
            do \
            qauth \
            sprack \
            publish \
            service \
            unsubscribe \
            prack }, \
  max_uri_len 512, \
  max_call_id_len 80, \
  max_requestName_len 20, \
  max_from_len 256, \
  max_to_len 256, \
  max_via_len 1024, \
  max_contact_len 512, \
  max_content_len 2048

# IMAP preprocessor.  For more information see README.imap
preprocessor imap: \
  ports { 143 } \
  b64_decode_depth 0 \
  qp_decode_depth 0 \
  bitenc_decode_depth 0 \
```

```
  uu_decode_depth 0

# POP preprocessor. For more information see README.pop
preprocessor pop: \
  ports { 110 } \
  b64_decode_depth 0 \
  qp_decode_depth 0 \
  bitenc_decode_depth 0 \
  uu_decode_depth 0

# Modbus preprocessor. For more information see README.modbus
preprocessor modbus: ports { 502 }

# DNP3 preprocessor. For more information see README.dnp3
preprocessor dnp3: ports { 20000 } \
  memcap 262144 \
  check_crc

# Reputation preprocessor. For more information see README.reputation
preprocessor reputation: \
  memcap 500, \
  priority whitelist, \
  nested_ip inner, \
  whitelist $WHITE_LIST_PATH/white_list.rules, \
  blacklist $BLACK_LIST_PATH/black_list.rules
#configurado para dar sensibilidad en el esacneo de puertos
#preprocessor sfportscan:  $HOME_NET 10 2 /path/to/logs/portscan./og
#preprocessor sfportscan: proto  all memcap 10000000 sense_level low

##################################################
# Step #6: Configure output plugins
# For more information, see Snort Manual, Configuring Snort - Output Modules
##################################################

# unified2
# Recommended for most installs
# output unified2: filename merged.log, limit 128, nostamp, mpls_event_types,
vlan_event_types
output unified2: filename snort.u2, limit 128
#vlan_event_types, mpls_event_types

# Additional configuration for specific types of installs
output alert_unified2: filename snort.alert, limit 128, nostamp
output log_unified2: filename snort.log, limit 128, nostamp

# syslog
output alert_syslog: LOG_AUTH LOG_ALERT
```

```
# pcap
output log_tcpdump: tcpdump.log

##################################################
# snortsam
##################################################
# In order to cause Snort to send a blocking request to the SnortSam agent,
# that agent has to be listed, including the port it listens on,
# and the encryption key it is using. The statement for that is:
#
# output alert_fwsam: {SnortSam Station}:{port}/{password}
#
#  {SnortSam Station}: IP address or host name of the host where SnortSam is
running.
#  {port}:           The port the remote SnortSam agent listens on.
#  {password}:       The password, or key, used for encryption of the
#                    communication to the remote agent.
#
output alert_fwsam: 10.1.6.1:898/Passw0rd
# At the very least, the IP address or host name of the host running SnortSam
# needs to be specified. If the port is omitted, it defaults to TCP port 898.
# If the password is omitted, it defaults to a preset password.
# (In which case it needs to be omitted on the SnortSam agent as well)
#
# More than one host can be specified, but has to be done on the same line.
# Just separate them with one or more spaces.
#
# Examples:
#
# output alert_fwsam: firewall/idspassword
# output alert_fwsam: fw1.domain.tld:898/mykey
# output alert_fwsam: 192.168.0.1/borderfw  192.168.1.254/wanfw

# metadata reference data.  do not modify these lines
include classification.config
include reference.config

##################################################
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
##################################################

# site specific rules
include $RULE_PATH/ReglasSnortAzul.rules
#include $RULE_PATH/ReglasSnortVerde.rules
#include $RULE_PATH/ReglasSnortRojo.rules
```

```
##################################################
# Step #8: Customize your preprocessor and decoder alerts
# For more information, see README.decoder_preproc_rules
##################################################

# decoder and preprocessor event rules
# include $PREPROC_RULE_PATH/preprocessor.rules
# include $PREPROC_RULE_PATH/decoder.rules
# include $PREPROC_RULE_PATH/sensitive-data.rules

##################################################
# Step #9: Customize your Shared Object Snort Rules
# For more information, see http://vrt-blog.snort.org/2009/01/using-vrt-certified-
shared-object-rules.html
##################################################

# dynamic library rules
# include $SO_RULE_PATH/bad-traffic.rules
# include $SO_RULE_PATH/chat.rules
# include $SO_RULE_PATH/dos.rules
# include $SO_RULE_PATH/exploit.rules
# include $SO_RULE_PATH/icmp.rules
# include $SO_RULE_PATH/imap.rules
# include $SO_RULE_PATH/misc.rules
# include $SO_RULE_PATH/multimedia.rules
# include $SO_RULE_PATH/netbios.rules
# include $SO_RULE_PATH/nntp.rules
# include $SO_RULE_PATH/p2p.rules
# include $SO_RULE_PATH/smtp.rules
# include $SO_RULE_PATH/snmp.rules
# include $SO_RULE_PATH/specific-threats.rules
# include $SO_RULE_PATH/web-activex.rules
# include $SO_RULE_PATH/web-client.rules
# include $SO_RULE_PATH/web-iis.rules
# include $SO_RULE_PATH/web-misc.rules

# Event thresholding or suppression commands. See threshold.conf
include threshold.conf
```

ANEXO C

Archivo de configuración de Ossec.

```
<ossec_config>
 <global>
  <email_notification>no</email_notification>
 </global>

 <rules>
  <include>rules_config.xml</include>
  <include>pam_rules.xml</include>
  <include>sshd_rules.xml</include>
  <include>telnetd_rules.xml</include>
  <include>syslog_rules.xml</include>
  <include>arpwatch_rules.xml</include>
  <include>symantec-av_rules.xml</include>
  <include>symantec-ws_rules.xml</include>
  <include>pix_rules.xml</include>
  <include>named_rules.xml</include>
  <include>smbd_rules.xml</include>
  <include>vsftpd_rules.xml</include>
  <include>pure-ftpd_rules.xml</include>
  <include>proftpd_rules.xml</include>
  <include>ms_ftpd_rules.xml</include>
  <include>ftpd_rules.xml</include>
  <include>hordeimp_rules.xml</include>
  <include>roundcube_rules.xml</include>
  <include>wordpress_rules.xml</include>
  <include>cimserver_rules.xml</include>
  <include>vpopmail_rules.xml</include>
  <include>vmpop3d_rules.xml</include>
  <include>courier_rules.xml</include>
  <include>web_rules.xml</include>
  <include>web_appsec_rules.xml</include>
  <include>apache_rules.xml</include>
  <include>nginx_rules.xml</include>
  <include>php_rules.xml</include>
  <include>mysql_rules.xml</include>
  <include>postgresql_rules.xml</include>
  <include>ids_rules.xml</include>
  <include>squid_rules.xml</include>
  <include>firewall_rules.xml</include>
  <include>cisco-ios_rules.xml</include>
  <include>netscreenfw_rules.xml</include>
  <include>sonicwall_rules.xml</include>
  <include>postfix_rules.xml</include>
```

```
  <include>sendmail_rules.xml</include>
  <include>imapd_rules.xml</include>
  <include>mailscanner_rules.xml</include>
  <include>dovecot_rules.xml</include>
  <include>ms-exchange_rules.xml</include>
  <include>racoon_rules.xml</include>
  <include>vpn_concentrator_rules.xml</include>
  <include>spamd_rules.xml</include>
  <include>msauth_rules.xml</include>
  <include>mcafee_av_rules.xml</include>
  <include>trend-osce_rules.xml</include>
  <include>ms-se_rules.xml</include>
  <!-- <include>policy_rules.xml</include> -->
  <include>zeus_rules.xml</include>
  <include>solaris_bsm_rules.xml</include>
  <include>vmware_rules.xml</include>
  <include>ms_dhcp_rules.xml</include>
  <include>asterisk_rules.xml</include>
  <include>ossec_rules.xml</include>
  <include>attack_rules.xml</include>
  <include>openbsd_rules.xml</include>
  <include>clam_av_rules.xml</include>
  <include>dropbear_rules.xml</include>
  <include>local_rules.xml</include>
</rules>

<syscheck>
  <!-- Frequency that syscheck is executed - default to every 22 hours -->
  <frequency>79200</frequency>

  <!-- Directories to check  (perform all possible verifications) -->
  <directories check_all="yes">/etc,/usr/bin,/usr/sbin</directories>
  <directories check_all="yes">/bin,/sbin</directories>

  <!-- Files/directories to ignore -->
  <ignore>/etc/mtab</ignore>
  <ignore>/etc/mnttab</ignore>
  <ignore>/etc/hosts.deny</ignore>
  <ignore>/etc/mail/statistics</ignore>
  <ignore>/etc/random-seed</ignore>
  <ignore>/etc/adjtime</ignore>
  <ignore>/etc/httpd/logs</ignore>
  <ignore>/etc/utmpx</ignore>
  <ignore>/etc/wtmpx</ignore>
  <ignore>/etc/cups/certs</ignore>
  <ignore>/etc/dumpdates</ignore>
  <ignore>/etc/svc/volatile</ignore>
```

```
  <!-- Windows files to ignore -->
  <ignore>C:\WINDOWS/System32/LogFiles</ignore>
  <ignore>C:\WINDOWS/Debug</ignore>
  <ignore>C:\WINDOWS/WindowsUpdate.log</ignore>
  <ignore>C:\WINDOWS/iis6.log</ignore>
  <ignore>C:\WINDOWS/system32/wbem/Logs</ignore>
  <ignore>C:\WINDOWS/system32/wbem/Repository</ignore>
  <ignore>C:\WINDOWS/Prefetch</ignore>
  <ignore>C:\WINDOWS/PCHEALTH/HELPCTR/DataColl</ignore>
  <ignore>C:\WINDOWS/SoftwareDistribution</ignore>
  <ignore>C:\WINDOWS/Temp</ignore>
  <ignore>C:\WINDOWS/system32/config</ignore>
  <ignore>C:\WINDOWS/system32/spool</ignore>
  <ignore>C:\WINDOWS/system32/CatRoot</ignore>
</syscheck>

<rootcheck>
  <rootkit_files>/var/ossec/etc/shared/rootkit_files.txt</rootkit_files>
  <rootkit_trojans>/var/ossec/etc/shared/rootkit_trojans.txt</rootkit_trojans>
  <system_audit>/var/ossec/etc/shared/system_audit_rcl.txt</system_audit>
  <system_audit>/var/ossec/etc/shared/cis_debian_linux_rcl.txt</system_audit>
  <system_audit>/var/ossec/etc/shared/cis_rhel_linux_rcl.txt</system_audit>
  <system_audit>/var/ossec/etc/shared/cis_rhel5_linux_rcl.txt</system_audit>
</rootcheck>

<global>
  <white_list>127.0.0.1</white_list>
  <white_list>^localhost.localdomain$</white_list>
  <white_list>192.168.122.1</white_list>
</global>

<remote>
  <connection>syslog</connection>
</remote>

<remote>
  <connection>secure</connection>
</remote>

<alerts>
  <log_alert_level>1</log_alert_level>
</alerts>

<!--COMANDOS: utilizados para invocar active response-->
<command>
  <name>host-deny</name>
  <executable>host-deny.sh</executable>
  <expect>srcip</expect>
```

```
 <timeout_allowed>yes</timeout_allowed>
</command>

<command>
 <name>firewall-drop</name>
 <executable>firewall-drop.sh</executable>
 <expect>srcip</expect>
 <timeout_allowed>yes</timeout_allowed>
</command>

<command>
 <name>disable-account</name>
 <executable>disable-account.sh</executable>
 <expect>user</expect>
 <timeout_allowed>yes</timeout_allowed>
</command>

<command>
 <name>restart-ossec</name>
 <executable>restart-ossec.sh</executable>
 <expect></expect>
</command>


<command>
 <name>route-null</name>
 <executable>route-null.sh</executable>
 <expect>srcip</expect>
 <timeout_allowed>yes</timeout_allowed>
</command>

<!--Comando creado 09/01/2016-->
<command>
 <name>apache-deny</name>
 <executable>apache-deny.sh</executable>
 <expect>srcip</expect>
 <timeout_allowed>yes</timeout_allowed>
</command>

<!-- Active Response Config -->
<active-response>
 <!-- This response is going to execute the host-deny
   - command for every event that fires a rule with
   - level (severity) >= 6.
   - The IP is going to be blocked for  600 seconds.
  -->
 <command>host-deny</command>
 <location>local</location>
```

```
  <level>6</level>
  <timeout>600</timeout>
</active-response>

<active-response>
  <!-- Firewall Drop response. Block the IP for
    - 600 seconds on the firewall (iptables,
    - ipfilter, etc).
   -->
  <disabled>no</disabled>
  <command>firewall-drop</command>
  <location>local</location>
  <!--rules_id>5551, 5503, 1002, 510, 5720, 5716, 2502</rules_id-->
  <rules_id>5720, 5716, 2502</rules_id>
  <rules_group>authentication_failed,authentication_failures</rules_group>
  <level>6</level>
  <repeated_offenders>30,60,120</repeated_offenders>
  <timeout>60</timeout>
</active-response>

<!--ACTIVE RESPONSE CREADO PARA DENEGACIÓN DE APACHE-->
<active-response>
  <command>apache-deny</command>
  <location>all</location>
  <level>6</level>
  <timeout>600</timeout>
</active-response>


<!-- Files to monitor (localfiles) -->

<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/auth.log</location>
</localfile>

<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/syslog</location>
</localfile>

<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/dpkg.log</location>
</localfile>

<localfile>
  <log_format>apache</log_format>
```

```
  <location>/var/log/apache2/error.log</location>
 </localfile>

 <localfile>
  <log_format>apache</log_format>
  <location>/var/log/apache2/access.log</location>
 </localfile>

 <localfile>
  <log_format>command</log_format>
  <command>df -h</command>
 </localfile>

 <localfile>
  <log_format>full_command</log_format>
  <command>netstat -tan |grep LISTEN |grep -v 127.0.0.1 | sort</command>
 </localfile>

 <localfile>
  <log_format>full_command</log_format>
  <command>last -n 5</command>
 </localfile>
</ossec_config>
```

ANEXO D

Archivo de reglas de Snort.

```
#REGLAS PARA SLOWLORIS-HPING3-FUDP
alert tcp any any -> any any (msg:"DDOS Slowloris flooding the network)";
        content:"NOTIFY * HTTP/1.1"; sid:1234572; rev:1;fwsam: src, 1 minutes;)
alert udp any any -> any any (msg:"DDOS Slowloris flooding the network UDP)";
        content:"NOTIFY * HTTP/1.1"; sid:1234573; rev:1; fwsam: src, 1 minutes;)
alert tcp any any -> any any (msg: "DoS by slowloris Tool"; content:"GET /$rand
        HTTP/1.1"; sid: 1000446;rev:1; fwsam: src, 10 minutes;)
alert tcp any any -> any any (msg:"MALWARE-TOOLS Slowloris http DoS tool";
        flow:to_server,established; content:"User-Agent|3A| Mozilla/4.0
        |28|compatible|3B| MSIE 7.0|3B| Windows NT 5.1|3B| Trident/4.0|3B| .NET
        CLR 1.1.4322|3B| .NET CLR 2.0.503l3|3B| .NET CLR 3.0.4506.2152|3B| .NET
        CLR 3.5.30729|3B| MSOffice 12|29 0D 0A|"; content:"Content-Length|3A| 42";
        detection_filter:track by_src, count 10, seconds 300; metadata:service http;
        reference:cve,2007-0086; classtype:attempted-dos; sid:15578; rev:1; fwsam:
        src, 1 minutes;)
alert ip any any -> any any (msg:"Hping3  DDOS
        Detected";flow:to_server;detection_filter: track by_src, count 20, seconds
        5;fragbits:M+;sid:123123149; rev:1;fwsam: src, 1 minutes;)
#REGLA PARA ATAQUES HPING 33 (HACIANDO USO DE PUERTO 22 SSH)
alert tcp any any -> any 22 (flags: S; msg:"Ataque Hping3"; flow: stateless; threshold:
        type both, track by_src, count 70, seconds 10; sid:1000444;rev:1;fwsam: src, 1
        minutes;)
alert tcp any 22 -> any any (flags: AS; msg:"Ataque Hping3"; flow: stateless;
        threshold: type both, track by_src, count 70, seconds 10;
        sid:1000445;rev:1;fwsam: dst, 1 minutes;)
#alert udp any any -> any any (flags: S; msg:"Possible UDP DoS"; flow: stateless;
        threshold: type both, track by_src, count 70, seconds 10;
        sid:1000445;rev:1;fwsam: src, 1 minutes;)
```

```
#alert ip any any -> any any (msg:"Hping3  DDOS
        Detected";flow:to_server;detection_filter: track by_src, count 20, seconds
        5;fragbits:M+;sid:123123149; rev:1;fwsam: src, 1 minutes;)
alert tcp any any -> any any (msg:"Squid sync flood"; flow:established,to_server;
        detection_filter: track by_src, count 10, seconds 60; sid:1000001; rev:1;fwsam:
        src, 1 minutes;)
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"DOS Land attack"; id:3868;
        seq: 3868; flags:S; reference:cve,CVE-1999-0016; classtype:attempted-dos;
        sid:269; rev:1;fwsam: src, 1 minutes; )
#alert icmp any any -> any any (msg:"Alert Getting ICMP Flood
        Message";sid:1000004;rev:1;fwsam: src, 1 minutes;)
################################SLOWLORIS####################
        ########################################
alert tcp any 80 -> any any (msg:"Alert Getting TCP Flood Messag";sid:1000005;flags:
        A;rev:1;fwsam: dst, 1 minutes;)
##############################################################
        ########################################
#alert udp any any -> any any (msg:"Alert Getting UDP Flood
        Messag";sid:1000006;rev:1;fwsam: src, 1 minutes;fwsam: src, 1 minutes; )
alert tcp any any -> any any (msg:"Alert HTTP GET
        DDos";pcre:"/GET.*\htm/i";classtype:web-application-
        activity;sid:1000007;rev:1;fwsam: src, 1 minutes;fwsam: src, 1 minutes; )
alert tcp any any -> any any (msg:"Intento ataque a servicio
        HTTP";reference:arachnids,IDS411; classtype:attempted-admin;priority:10;
        sid:1000983; rev:1)
#####################################REGLA PARA ATAQUE
        FUDP#####################################
alert udp any any -> any any (msg:"DOS flood denial of service
        attempt";flow:to_server;detection_filter:track by_dst,count 50, seconds 1;
        metadata:service syslog; classtype:attempted-dos;sid:25102;rev:1;fwsam: src,
        1 minutes; )
```

```
#alert udp any any -> any any (msg:"DOS flood denial of service
        attempt";flow:to_server;detection_filter:track by_dst,count 50, seconds 1;
        itype:3; metadata:service syslog; classtype:attempted-
        dos;sid:25102;rev:1;fwsam: src, 1 minutes; )

######################REGLA PARA
        SLOWLORIS##################################################
        ############
alert tcp any any -> any 80 (msg:"DOS flood denial of service attempt";
        flow:to_server;detection_filter:track by_dst,count 50, seconds 1;
        metadata:service syslog; classtype:attempted-dos; sid:25101;rev:1;fwsam: src,
        1 minutes;)
#alert tcp any 80 -> any any (msg:"DOS flood denial of service attempt"; flags: A;
        flow:to_server;detection_filter:track by_dst,count 50, seconds 1;
        metadata:service syslog; classtype:attempted-dos; sid:25144;rev:1;fwsam: dst,
        1 minutes;)
#alert tcp any any -> any 80 (msg:"DOS flood denial of service
        attempt";flow:to_server;detection_filter:track by_dst,count 2, seconds 1;
        metadata:service syslog; classtype:attempted-dos; sid:25103;rev:1;fwsam: src,
        1 minutes;)
########################################escaneo de
        puertos########################################
#alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN nmap
        TCP";flags:A;ack:0; reference:arachnids,28; classtype:attemptedrecon;
        sid:628; rev:1;)
```