



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

“Implementación de un servidor IDS para monitoreo de tráfico de red”

INFORME DE ACTIVIDADES PROFESIONALES

Que para obtener el título de
Ingeniero en Computación

P R E S E N T A

David Tejada Rentería

ASESOR DE INFORME

Ing. Cruz Sergio Aguilar Díaz



Ciudad Universitaria, Cd. Mx., 2016

Agradecimientos

Irasema Lilian García Pantoja

A mi novia, quien me ha acompañado a lo largo de este viaje y que ha sido pieza fundamental en mi desarrollo personal y profesional, siempre dándome ánimos y apoyo para no claudicar y seguir adelante cuando las cosas se tornaban complicadas.

Lorena Rentería Lizardi y Erick Tejada Rentería

A mi madre y mi hermano quienes siempre creyeron en mí y estuvieron a mi lado en esos momentos que quería tirar la toalla, tolerando mi frustración y angustia a lo largo de la carrera.

Ing. Cruz Sergio Aguilar Díaz

A mi asesor académico cuya ayuda y consejos me permitieron terminar este ciclo de manera satisfactoria, siendo siempre alguien en quien pude depositar mi confianza y quien me brindó la oportunidad de desarrollarme más como persona cuando me acepto como personal de la sala 3 de UNICA.

UNICA

A los Ingenieros, compañeros y personas que conocí durante mi estancia en la Sala 3 de UNICA. Agradezco las lecciones aprendidas, los momentos de alegría, el conocimiento compartido, el compañerismo demostrado y la ayuda que me brindaron.

A la UNAM y la Facultad de Ingeniería

Finalmente agradezco a la UNAM y la Facultad de Ingeniería por brindarme la oportunidad de ser uno de los privilegiados en estudiar en sus aulas, permitiéndome convivir con personas de todo tipo, además de la gran experiencia que significó para mí, formar parte de estas maravillosas instituciones.

Índice

ÍNDICE DE FIGURAS	6
INTRODUCCIÓN	8
CAPÍTULO 1 DESCRIPCIÓN DE ACTIVIDADES PROFESIONALES	14
1.1 EXPERIENCIA PROFESIONAL	16
1.2 ACTIVIDADES COMO <i>INCIDENT HANDLER</i> DEL <i>BLUE TEAM</i>	17
1.2.1 Monitoreo de red	18
1.2.2 Gestión y respuesta a incidentes de seguridad (<i>Incident Handling</i>)	21
1.2.3 Cibervigilancia	22
CAPÍTULO 2 DESCRIPCIÓN DEL PROYECTO	24
2.1 ANTECEDENTES	26
2.2 PLANTEAMIENTO DEL PROBLEMA	26
2.3 OBJETIVO DEL PROYECTO	26
2.4 REQUISITOS PARA LA IMPLEMENTACIÓN	26
2.5 ETAPAS DEL PROYECTO	27
CAPÍTULO 3 IMPLEMENTACIÓN DE UN SERVIDOR IDS CON SNORT	30
3.1 FASE 2: INSTALACIÓN Y CONFIGURACIÓN DE SNORT	32
3.1.1 Requerimientos	33
3.1.2 Instalación	36
3.1.3 Configuración como IDS	37
3.1.4 Configuración del preprocesador HTTP Inspect	46
3.2 FASE 3: INSTALACIÓN Y CONFIGURACIÓN DE BARNYARD2	47
3.2.1 Requerimientos	47
3.2.2 Instalación	49
3.2.3 Configuración	49
3.3 FASE 4: INSTALACIÓN Y CONFIGURACIÓN DE PULLEDPORK	52
3.3.1 Requerimientos	52
3.3.2 Instalación	52
3.3.3 Configuración	53
3.3.3.1 Configuración de PuledPork para inicio automático	56
3.4 FASE 5: INSTALACIÓN Y CONFIGURACIÓN DE BASE	57
3.4.1 Requerimientos	57
3.4.2 Instalación	58
3.4.3 Configuración	59
CAPÍTULO 4 PRUEBAS Y RESULTADOS	62
4.1 PRUEBAS	64
4.2 RESULTADOS	68
CONCLUSIONES	70
GLOSARIO	76
REFERENCIAS	80
BIBLIOGRAFÍA:	82
REFERENCIAS ELECTRÓNICAS	82
GLOSARIO	83
ANEXOS	86
ANEXO A - OPCIONES DE CONFIGURACIÓN PARA HTTP_INSPECT GLOBAL	88
ANEXO B - CONFIGURACIÓN DE INTERFACES	93
ANEXO C - INSTALACIÓN MANUAL DE LAS REGLAS DE SNORT	94
ANEXO D - RECONSTRUYENDO LA BASE DE DATOS DE SNORT	95
ANEXO E - SCRIPTS DE INICIO AUTOMÁTICO PARA SNORT Y BARNYARD2	96

Índice de Figuras

Número	Título	Página
Figura A	Organigrama GCS-CERT.	8
Figura 1.1	Servidor IDS monitoreando toda la red.	15
Figura 1.2	Servidores IDS monitoreando distintos segmentos de una red.	16
Figura 1.3	Ciclo de vida de respuestas a un incidente.	17
Figura 2.1	Esquema de las etapas del proyecto.	21
Figura 3.1	Arquitectura de Snort.	24
Figura 3.2	Carpeta snort_src.	25
Figura 3.3	Requerimientos de DAQ.	25
Figura 3.4	Descomprimiendo el paquete DAQ.	27
Figura 3.5	Biblioteca de compresión.	28
Figura 3.6	Compilación de Snort.	28
Figura 3.7	Actualización de bibliotecas compartidas y creación de enlace simbólico.	29
Figura 3.8	Versión de Snort junto con los archivos de configuración que utiliza.	29
Figura 3.9	Creación del grupo y el usuario snort.	29
Figura 3.10	Creación de la estructura de archivos.	30
Figura 3.11	Se modifican tanto los permisos como el propietario de los directorios.	30
Figura 3.12	Estructura de directorios de Snort.	31
Figura 3.13	Resultado en el archivo snort.conf tras la ejecución del comando.	32
Figura 3.14	Configuración de las redes a monitorear.	33
Figura 3.15	Rutas a los archivos de las reglas.	33
Figura 3.16	Archivo local.rules habilitado para pruebas.	34
Figura 3.17	Regla que se agregó en el archivo local.rules.	34
Figura 3.18	Inicio de la prueba.	35
Figura 3.19	Verifica si Snort carga la regla correctamente.	35
Figura 3.20	Mensaje de prueba exitosa del archivo de configuración.	36
Figura 3.21	Snort ejecutándose.	36
Figura 3.22	Ping hacia un equipo con dirección IP 10.30.1.25.	37
Figura 3.23	Alertas de Snort.	37
Figura 3.24	Bitácoras de Snort.	37
Figura 3.25	Configuración del preprocesador Http inspect.	38
Figura 3.26	Creación de la contraseña del usuario root en MySQL.	39
Figura 3.27	Archivo <i>snort.conf</i> modificado.	40
Figura 3.28	Abriendo el archivo barnyard2.conf.	42
Figura 3.29	Modificando el archivo barnyard2.conf.	42
Figura 3.30	Snort en ejecución.	43
Figura 3.31	Comprobación de los eventos en la base de datos.	43
Figura 3.32	Comprobación del archivo pulledpork.pl.	46
Figura 3.33	Salida correcta al ejecutar pulledpork.pl.	46
Figura 3.34	Línea agregada al archivo snort.conf.	47
Figura 3.35	Selección del editor de su preferencia.	48
Figura 3.36	Añadiendo el archivo pulledpork.pl a crontab.	48
Figura 3.37	Instalación de los requerimientos de BASE.	49

Figura 3.38	Instalando <i>Image Graph</i> .	49
Figura 3.39	Preparando la instalación de BASE.	50
Figura 3.40	Instalando BASE.	50
Figura 3.41	Cambiando permisos del directorio de BASE.	51
Figura 3.42	Inicio de la configuración de BASE.	52
Figura 3.43	Creando las tablas necesarias para BASE.	52
Figura 3.44	Mensaje de tablas creadas correctamente.	53
Figura 3.45	BASE configurado y funcional.	53
Figura 4.1	Alertas de actividad generadas por el equipo con dirección IP 10.30.1.58.	55
Figura 4.2	Revisión de petición DNS a sitio para adultos.	56
Figura 4.3	Revisión de petición DNS a dominio .tk.	57
Figura 4.4	Revisión de petición DNS a dominio .su.	57
Figura 4.5	Visita a sitio 3dnews.su.	58



Introducción

En la actualidad, la seguridad informática es un tema de suma relevancia, debido a la necesidad de garantizar que la información y los recursos informáticos de una organización se encuentren disponibles en todo momento para cumplir con sus propósitos.

Es por ello que existen empresas, como en la que me encuentro laborando, que se brindan diferentes servicios de Ciberseguridad a instituciones públicas y privadas.

Breve reseña de la EMPRESA DE CIBERSEGURIDAD

Se trata de una empresa con experiencia en sistemas de gestión de seguridad de la información; en el diseño, implementación y manejo de centros de Ciberseguridad y respuesta a incidentes (SOC/CERT's), cuya finalidad es responder de un modo rápido y eficaz a posibles riesgos o ataques que puedan afectar la infraestructura crítica de distintos sectores.

La empresa ofrece una visión de seguridad alineada con los objetivos de instituciones públicas y privadas. Cuenta con procesos definidos y auditados periódicamente, además de metodologías propias para la prestación de servicios certificados en sistemas de calidad.

Servicios

En un entorno donde los sistemas de información cuentan cada vez con más complejidad y en el que el número de amenazas crece día a día, incluyendo ataques externos e internos cada vez más sofisticados. En la EMPRESA DE CIBERSEGURIDAD se cuenta con los siguientes servicios a la vanguardia para la defensa cibernética, con el esfuerzo humano y material que esto supone:

- *Blue Team*
 - Monitoreo de incidentes de seguridad
 - Gestión y respuesta a incidentes de seguridad
 - Análisis de seguridad a infraestructura de TI
 - Cibervigilancia

- *Red Team*
 - *Hacking ético*
 - *Hacking a SOC's*
 - Análisis de vulnerabilidades
 - *Security Awareness Program*
 - Análisis forense
 - Revisión de código
 - Laboratorio de malware

Organigrama

En la Figura A se presenta la estructura jerárquica y las relaciones que hay entre las diferentes áreas presentes dentro de la empresa.

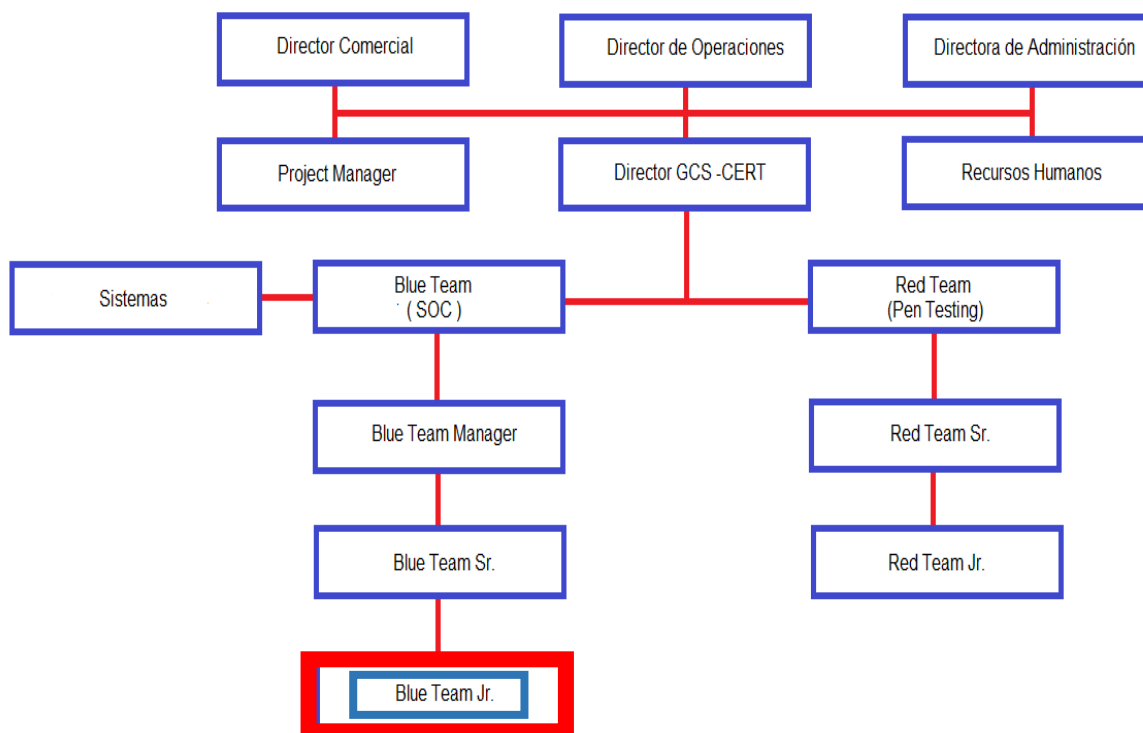


Figura. A Organigrama GCS - CERT.

Director Comercial.- Se trata del responsable del cumplimiento de los objetivos comerciales de la empresa.

Director de Operaciones.- Se encarga de contribuir con el desarrollo de los proyectos y operaciones en la empresa.

Directora de Administración.- Crea y propone normas, políticas y procedimientos para el mejor funcionamiento de las actividades relacionadas con la administración y contabilidad de la organización.

Project Manager.- Tiene la responsabilidad de planear y coordinar los procedimientos y ejecución de los proyectos de la empresa, además de fungir como enlace con los clientes.

Recursos Humanos.- Se ocupa de la administración de personal, artículos de oficina y control de asistencia. Elaboran informes referentes al cumplimiento de metas, de la normativa legal laboral y llevan a cabo actividades para el control y aplicación de las mismas.

Director GCS-CERT.- Dirige y coordina a los equipos rojo y azul, gestionando y asignando tareas acordes a los proyectos de cada área.

Sistemas.- Planean y coordinan las estrategias necesarias para establecer los sistemas de información que apoyen y beneficien a la empresa. También administran la configuración y seguridad de la red local asegurando el uso óptimo y racional de las Tecnologías de la Información.

Blue Team (SOC).- El equipo azul se encarga de defender los sistemas de información de una empresa y que ante la detección de vulnerabilidades recomienda técnicas de mitigación a los clientes.

Red Team (Pen Testing).- El equipo rojo simula ser un atacante, y que autorizado por una organización, detecta vulnerabilidades y realiza ataques en contra de sus sistemas de información.

Blue Team Manager.- Coordina a los integrantes del equipo azul asignando a cada miembro actividades relacionadas con los proyectos del área.

Blue/Red Team Sr.- Responsable de alguno de los proyectos del área correspondiente.

Blue/Red Team Jr.- Se dedica a distintas actividades en el área correspondiente como son; monitoreo de red, gestión y respuesta a incidentes de seguridad y cibervigilancia. Siguiendo las indicaciones del Blue Team Sr., del Blue Team Manager o del Director GCS-CERT.

Al terminar mis estudios en la Facultad de Ingeniería, ingrese a trabajar a la EMPRESA DE CIBERSEGURIDAD como miembro del equipo azul (*Incident Handler Jr. del Blue Team*) dentro del SOC de la empresa en cuestión.

Actualmente llevo alrededor de año y medio desempeñando las funciones propias del puesto, donde he tenido la oportunidad de aprender a utilizar diversas herramientas que son necesarias para poder cumplir puntualmente con mis obligaciones laborales, viajar al interior de la república para trabajar directamente en las instalaciones de algún cliente y participar en diferentes proyectos junto con mis compañeros del SOC.

El presente informe para titulación por experiencia profesional, tiene como fin, dar un panorama general acerca del procedimiento para habilitar un servidor IDS (*Intrusion Detection System*) utilizando herramientas de código libre, instalando y configurando los paquetes de software necesarios para ello.

La necesidad de implementar este tipo de herramienta, surgió debido a los servicios proporcionados por la empresa, ya que al encontrarse en el ámbito de seguridad de la información, se requiere de soluciones que permitan tener un enfoque integral capaz de adaptarse a las necesidades de cualquier cliente.

Desde la actuación preventiva a la reactiva y de contención, así como la recuperación de los activos en el mínimo tiempo posible. Este proyecto por lo tanto, generó un aporte en beneficio de la empresa.

Además de lo anterior, se explicará también, el proceso de análisis de tráfico de red utilizando las herramientas instaladas previamente.

Por cuestiones de políticas de seguridad y confidencialidad de la empresa, este informe no muestra en su totalidad el diseño y desarrollo del proyecto. **Es por esta razón que se modificó o censuró parte de la información aquí utilizada, ya que se le considera de carácter sensible y/o confidencial.**



Capítulo 1

Descripción de actividades profesionales

A continuación se encuentra mi experiencia profesional en trabajos previos, donde aprendí lecciones valiosas, gracias a las cuales logre darme cuenta de la importancia de ser profesional y responsable al momento de desempeñar las labores asignadas dentro de una empresa. Además de mantener siempre la cordialidad con los compañeros de trabajo y una buena actitud ante las distintas situaciones que se pudieran suscitar.

1.1 Experiencia profesional

Becario en la Unidad de Servicios de Cómputo Académico (UNICA)

Durante mi estancia en la Facultad de Ingeniería, hice mi servicio social en la sala 3 de UNICA, comenzando aproximadamente, en agosto del año 2012 y concluyendo en enero del año 2013. Algunos meses después de terminar el servicio social, aún como estudiante de la Facultad y como personal de apoyo en la sala, se me ofreció la posibilidad de continuar como becario (posición que conservé hasta mi salida en enero de 2015), siendo esta una buena oportunidad de aprendizaje y desarrollo.

Actividades:

- Administración y soporte al equipo de cómputo del laboratorio.
- Instalación y configuración de sistemas operativos Windows y diversas distribuciones de Linux.
- Instalación de distintos programas utilizados por los alumnos de la Facultad.
- Administración de cuentas de usuario
- Atención al usuario.

Logros: Desarrollé proyectos tales como reconocimiento y mapeo de la red de la sala de cómputo, investigación e implementación de diversos tipos de servidores para uso del personal de la sala.

Profesor de informática en el Colegio Palmerston S.C.

Mis actividades como profesor comenzaron desde finales del año 2004 justo al terminar mi carrera técnica como programador y mucho antes de entrar a la facultad de ingeniería. Mi estancia en el colegio concluyó en enero del año 2015 cuando me retiré para ingresar a mi trabajo actual. Durante ese tiempo, además de experiencia, obtuve un conjunto de habilidades muy útiles, entre las que puedo destacar la de poder hablar y desenvolverme al momento de exponer temas ante un grupo de personas, también tuve la oportunidad de aplicar de forma práctica los conocimientos que iba adquiriendo durante la carrera, ya que durante este periodo de tiempo se dio mi ingreso a la facultad.

Actividades:

- Dar clases de informática a niños de entre 3 y 12 años.
- Soporte preventivo y correctivo al equipo de cómputo del colegio.

Logros: Desarrollé, administré y mantuve el sitio web del colegio. Sugerí la renovación y estandarización del equipo de cómputo para una mejor administración del mismo. También configuré las redes del salón de cómputo y de la dirección del plantel.

1.2 Actividades como *Incident Handler* del *Blue Team*

El presente informe describe algunas de las actividades profesionales que realicé en el SOC (Centro de Operaciones de Seguridad) de la EMPRESA DE CIBERSEGURIDAD desde mi ingreso a la misma el mes de enero del año 2015. Dichas actividades incluyen la implementación de servidores IDS basados en SNORT con interfaces web que permiten visualizar las ocurrencias de incidentes detectados y almacenados en la base de datos SNORT.

Lo anterior permite llevar a cabo un monitoreo constante en la red de los clientes, con lo que se pueden prevenir y/o corregir ataques a su infraestructura crítica.

Dentro de las funciones generales que se realizan en el SOC se encuentran:

- Soporte técnico al equipo de cómputo del área.
- Monitoreo de red.
- Gestión y respuesta a incidentes de seguridad.
- Administración de los Servidores del área.
- Análisis de seguridad a infraestructura de TI.
- Cibervigilancia.
- Administración de la infraestructura de red del área.

1.2.1 Monitoreo de red

Debido a que en la actualidad se depende cada vez más de las redes de datos y a que la cantidad de aplicaciones y servicios presente en las mismas crece día con día, se hace evidente la necesidad de contar con dispositivos que permitan brindar seguridad a todos estos activos en una organización.

Es por este motivo que se han implementado una serie de servidores IDS (Sistema de Detección de Intrusos por sus siglas en inglés) con la finalidad de tener conocimiento constante sobre la actividad presente en la red del cliente, con esto, se busca detectar e identificar cualquier actividad que pudiera considerarse inusual, sospechosa o nociva.

Se debe tener siempre presente que la actividad antes mencionada se puede dar de dos formas: la externa, es decir, aquella que proviene de internet hacia la red que se está vigilando (ataques tipo *defacement*, intentos de subir archivos maliciosos a un servidor para obtener acceso remoto, entre otros.), y la interna, que es la generada desde el interior de una red hacia el exterior (por ejemplo, actividad por código malicioso o violaciones de políticas institucionales de navegación).

Es necesario que los dispositivos se conecten a un puerto configurado como espejo en un *switch*, es decir, un puerto al que se envían copias de los paquetes que pasan por dicho dispositivo sin interrumpir su flujo normal.

El punto físico de conexión de estas herramientas puede variar dependiendo de lo que se desee vigilar:

- En la Figura 1.1 se muestra un sensor (servidor IDS) monitoreando desde la zona perimetral de la red, en este punto, se puede detectar todo lo que proviene desde la red local (alámbrica e inalámbrica) y de la DMZ (zona desmilitarizada). No obstante, además de que esto significa una mayor carga de trabajo para el servidor, el tráfico es detectado una vez que fue filtrado por el Firewall, por lo que generalmente es necesario revisar la bitácoras del mismo para identificar los equipos que pudieran estar generando actividad maliciosa.
- En la Figura 1.2 se muestran varios servidores IDS en una misma red, donde cada uno se encarga de vigilar el tráfico de distintos segmentos de la misma. Esto permite contar con un mejor control sobre la actividad monitoreada dando oportunidad de detectar incidentes con mayor facilidad, además de dividir la carga de trabajo entre varios equipos mejorando así su desempeño.

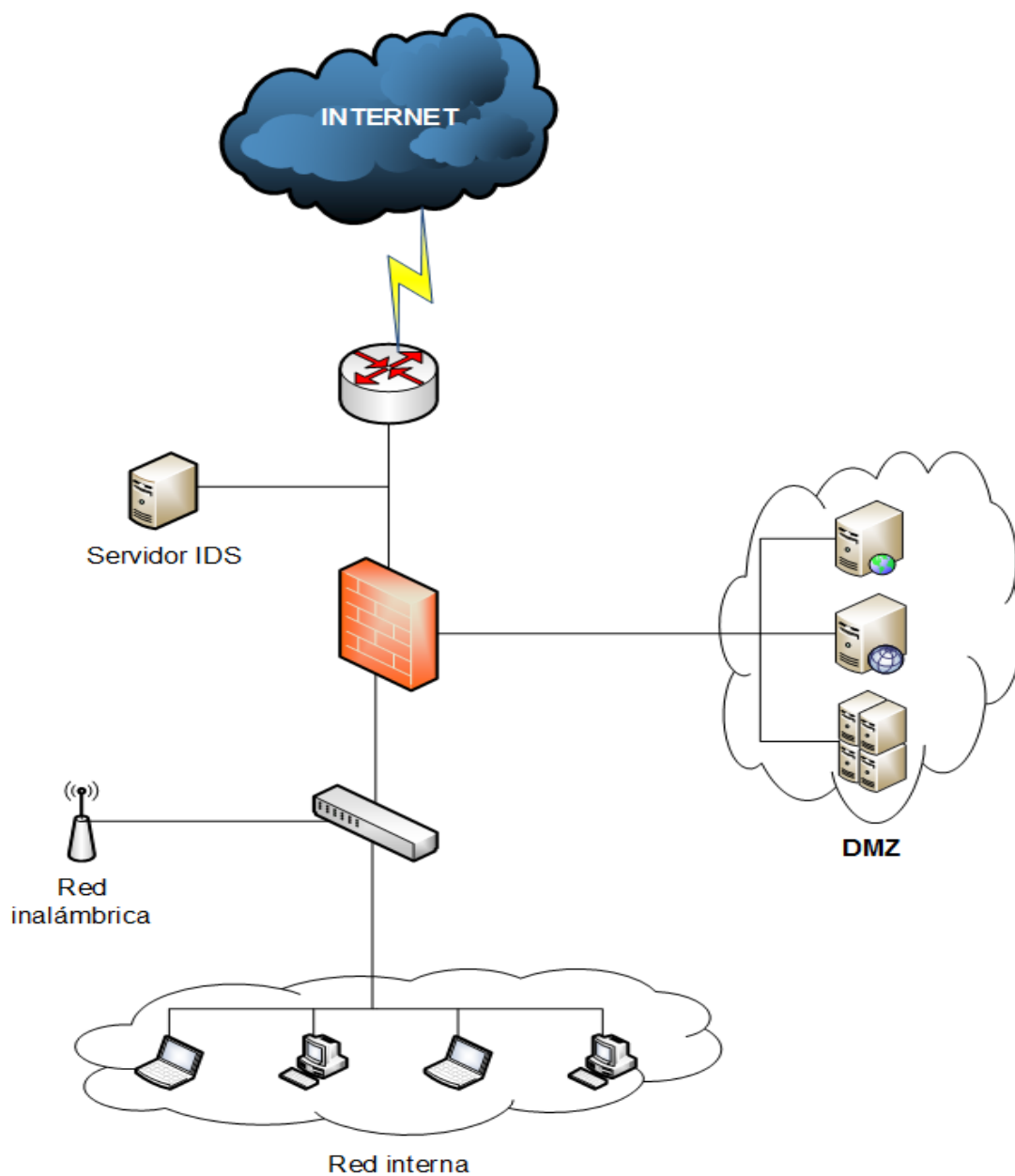


Figura. 1.1 Servidor IDS monitoreando toda la red.

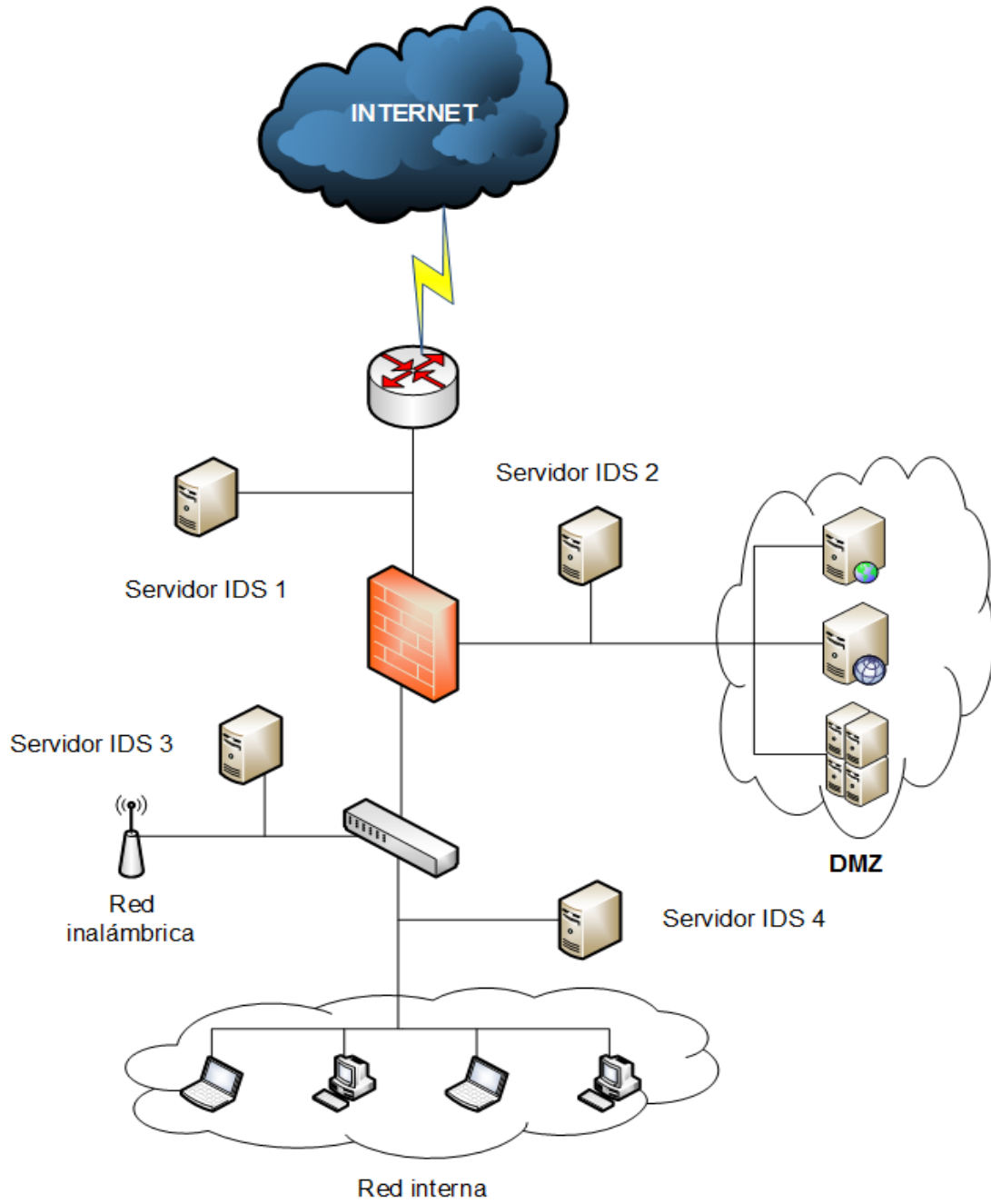


Figura. 1.2 Servidores IDS monitoreando distintos segmentos de una red.

1.2.2 Gestión y respuesta a incidentes de seguridad (*Incident Handling*)

El proceso de respuesta a incidentes cuenta con varias fases tal como se muestra en la Figura 1.3. Mi función es detectar y analizar los incidentes que se presentan en las redes monitoreadas, apoyándome en la actividad visible por medio de los IDS, posteriormente doy una serie de recomendaciones con lo que se busca mitigar la amenaza.

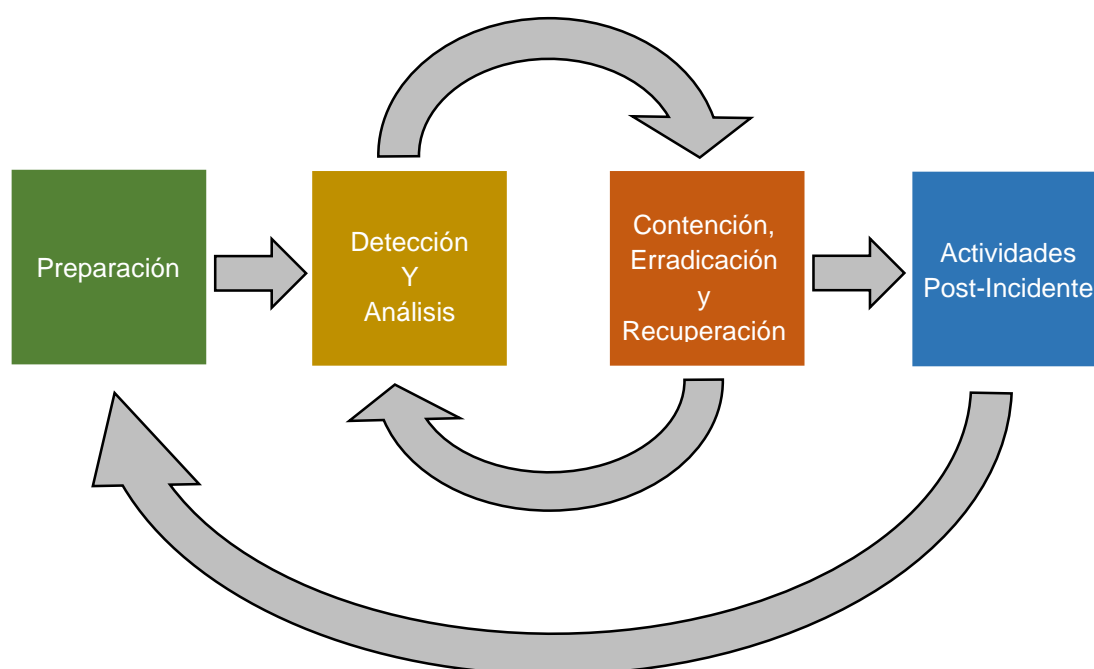


Figura. 1.3 Ciclo de vida de respuesta a un incidente.

La detección se hace mediante una serie de Indicadores de Compromiso (IoC) que me permiten tener una idea general de qué tipo de actividad es la que estoy monitoreando. En muchas ocasiones estos indicadores contienen información útil que permite realizar un análisis más profundo, sin embargo, también se da el caso de que únicamente muestren la actividad detectada sin ningún tipo de información extra.

Cuando se da la ocurrencia de algún evento, mi primer paso es verificar que efectivamente se trate de un incidente de seguridad y no de algún falso positivo. Esto se consigue analizando el contenido (*payload*) que pudiera haber dentro del tráfico marcado por los indicadores de compromiso.

El análisis puede brindar datos tales como:

- El **vector de ataque** utilizado en caso de tratarse de un incidente de seguridad, lo que permite tener certeza acerca de las medidas preventivas que se deben tomar.

- Las **direcciones IP** involucradas en el evento. Esta información es útil ya que ayuda a determinar cuál es el origen y el destino de la actividad analizada, lo que permite determinar el flujo de la actividad el cual puede presentarse como: **extrusión, intrusión o tráfico local**.

Dependiendo del tipo de flujo se pueden tomar medidas diferentes ya sea para contención o erradicación. Además, en caso de tratarse de una dirección IP pública existen diversas herramientas, como pueden ser las **listas negras**, que permiten determinar si dicha dirección es maliciosa y en caso de serlo, que tipo de actividad presenta.

- El **host** a donde se dirige el tráfico y que generalmente se trata de algún **dominio** de internet, en algunas ocasiones se puede dar una situación donde una dirección IP pública no se encuentra presente en ninguna lista negra, sin embargo, puede estar asociada con dominios que si lo estén debido a algún tipo de actividad maliciosa como puede ser la propagación de malware, o que se trate por ejemplo, de un sitio con material pornográfico. Es por este motivo, que además de las direcciones IP también es importante analizar el *host* para lograr un análisis más concluyente.
- El **agente de usuario** es una cabecera presente dentro de una petición **HTTP** que sirve para identificar la aplicación que se encuentra realizando dicha actividad, existen diversas cepas de malware que realizan peticiones a servidores C&C (Command and Control) y que utilizan esta cabecera para identificarse en dicho servidor. Esto permite detectar por ejemplo si un equipo en particular se encuentra infectado por algún tipo de malware entre otras cosas.

Una vez identificado el tipo de incidente que se presentó, se da al cliente una descripción de lo observado junto con la evidencia pertinente que puede componerse por ejemplo; de capturas de pantalla, entre otras cosas, y una serie de recomendaciones basadas en los resultados arrojados por el análisis. Las cuales tienen como finalidad brindar una guía acerca de qué medidas tomar para mitigar y/o solucionar la actividad que se presentó.

Dichas recomendaciones pueden incluir soluciones tan sencillas como realizar análisis antivirus en los equipos que se reportan o implementar políticas de seguridad institucionales, hasta bloquear ciertas direcciones IP o Dominios dañinos en dispositivos de seguridad perimetral como pueden ser *Firewalls Proxys*.

1.2.3 Cibervigilancia

A través del uso de diversas técnicas y herramientas, realizo un monitoreo en la *deep web*, en internet y en distintas redes sociales, con la finalidad de detectar amenazas o información que pueda ser perjudicial para los objetivos vigilados.



Capítulo 2

Descripción del proyecto

2.1 Antecedentes

El presente proyecto se planteó por la necesidad de implementar una herramienta capaz de permitir a los integrantes del SOC, llevar a cabo de manera más sencilla sus actividades de monitoreo de tráfico de red, permitiendo hacer más eficiente el proceso de detección y notificación de incidentes de seguridad.

2.2 Planteamiento del problema

Debido a la naturaleza de los servicios ofrecidos por la empresa, se tenía la necesidad de contar con herramientas adecuadas para el monitoreo de redes. Sin embargo, al ser una empresa en desarrollo, debía ser una solución viable sin tener que invertir una gran suma de dinero en ella, y que a su vez, permitiera a los miembros del SOC realizar sus actividades de gestión y respuesta a incidentes de manera práctica y eficiente.

Fue por este motivo que se planteó el uso de servidores con software de código libre, lo que supondría un ahorro al invertir únicamente en el hardware necesario, prescindiendo de costos elevados por uso de licencias y así tener una herramienta; eficiente, funcional y moldeable.

2.3 Objetivo del proyecto

Describir de forma detallada la instalación y configuración del software de código libre necesario para implementar un servidor IDS, con lo que se busca contar con los medios necesarios para que el personal del SOC pueda detectar, analizar y responder de una manera más eficaz a incidentes de seguridad que busquen dañar la información, infraestructura crítica y/o recursos de una organización.

2.4 Requisitos para la implementación

Para implementar un servidor que utilice SNORT como motor IDS de manera adecuada, es necesario contar con conocimientos en distintas áreas, tales como:

- Manejo de Sistemas Operativos basados en Linux, utilizando la interfaz de línea de comandos.
- Bases de datos.
- Seguridad Informática.
- Redes de datos.

Dichos conocimientos ayudaron a cumplir exitosamente con el objetivo del proyecto asignado, cumpliendo con las metas que se enlistan a continuación:

- Instalar y configurar los servicios necesarios para habilitar un servidor como sistema para detección de intrusiones (IDS).
- Desplegar el servidor de manera exitosa en un punto designado, para poder llevar a cabo las labores de monitoreo por parte del personal del SOC.

2.5 Etapas del proyecto

La Figura 2.1 muestra el esquema de las etapas presentes en el proyecto. Se puede observar, además del inicio y fin del mismo, un ciclo en el cual se realizaron diversas pruebas por partes para verificar que todos los elementos funcionaran de manera correcta.

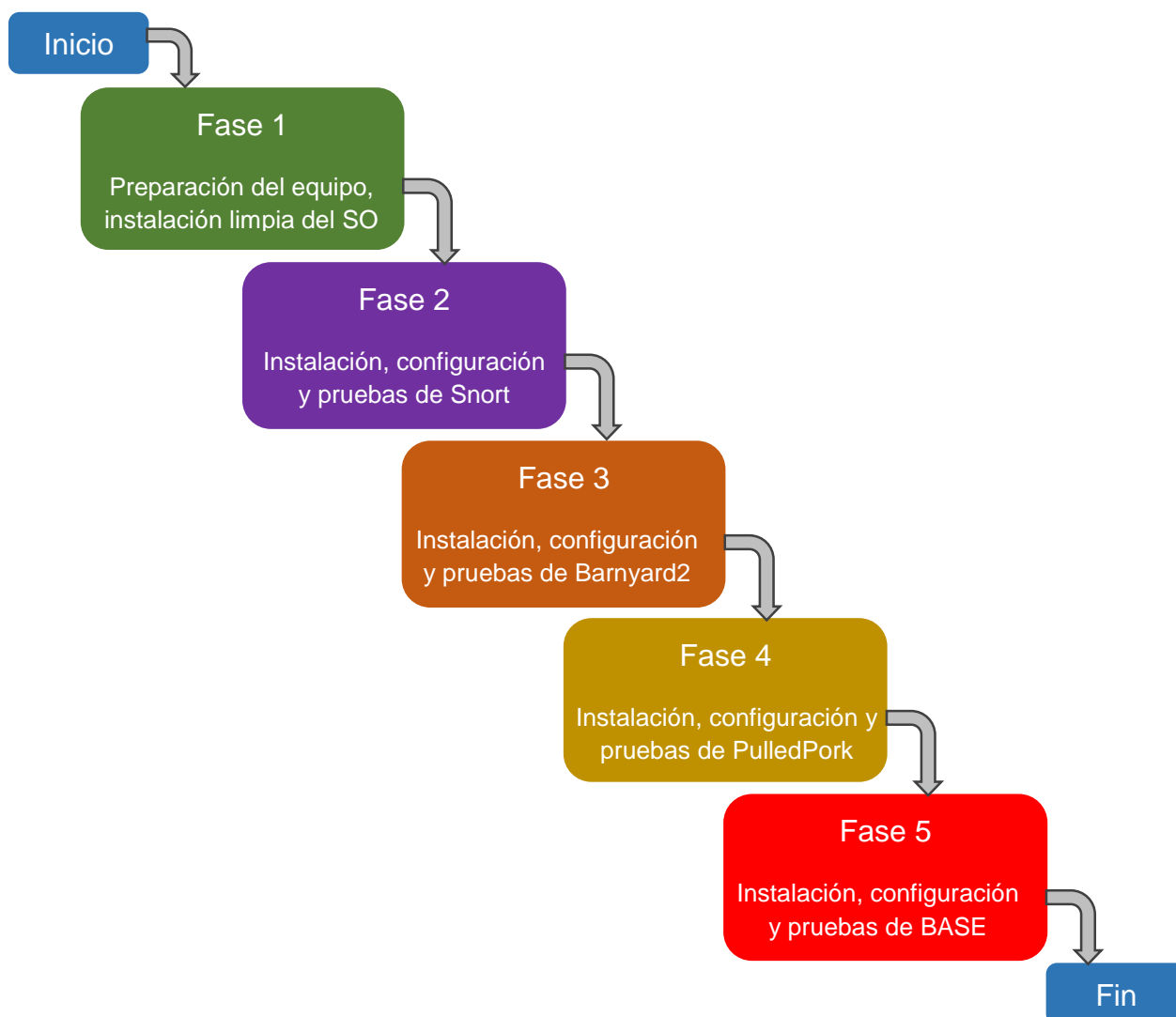


Figura. 2.1 Esquema de las etapas del proyecto.

- **Fase 1.- Preparación del equipo.**

Se trata de la etapa inicial del proyecto, es en esta fase donde se elige el hardware que se utilizará para montar el servidor y se alista el equipo realizando una instalación limpia del Sistema Operativo base que se desea.

- **Fase 2.- Instalación y configuración de SNORT.**

Se llevan a cabo todos los pasos necesarios para instalar **SNORT**, una vez hecho esto, se configura de acuerdo a las necesidades de la red que se desea monitorear y se realizan pruebas para corroborar que haya quedado correctamente instalado y que funcione de manera adecuada.

- **Fase 3.- Instalación y configuración de Barnyard2**

Una vez que SNORT se encuentra instalado y que se corroboró que funciona de manera correcta, la fase 3 consiste en instalar **Barnyard2**. Este paquete será el encargado de tomar los incidentes detectados por SNORT y almacenarlos en una base de datos.

- **Fase 4.- Instalación y configuración de PulledPork**

En esta fase se instala el paquete **PulledPork**, dicho paquete será el encargado de descargar a diario las reglas utilizadas por SNORT. Este software es muy relevante, ya que es el encargado de mantener a SNORT actualizado, ya que las reglas que utiliza, se encuentran cambiando constantemente.

- **Fase 5.- Instalación y configuración de BASE**

Se trata de la última fase del proyecto y es donde se trabaja con la interfaz web que brindará acceso a las alertas generadas por SNORT y almacenadas por Barnyard2 de una forma más amigable.

Debido a las dependencias que existen entre todas estas herramientas, es importante realizar pruebas después de cada fase para verificar que todo se encuentra funcionando de manera correcta. De lo contrario, será necesario verificar las configuraciones realizadas en cada una de ellas.



Capítulo 3

Implementación de un servidor IDS con SNORT

La Fase 1 del proyecto contempla la instalación del Sistema Operativo base, sin embargo, debido a que esta etapa es relativamente sencilla, se realiza prácticamente de manera automática y no requiere mayor configuración. No se abordará a detalle, por lo que se inicia con la Fase 2 del proyecto.

3.1 Fase 2: Instalación y configuración de SNORT

Se eligió utilizar este IDS porque representa una solución de código libre; ligera, estable, rápida, y robusta que puede ajustarse a las necesidades de diferentes organizaciones. Además de ser el motor utilizado por diferentes productos comerciales entre los que destacan los de Sourcefire, por ende, cuenta con un amplio soporte y se encuentra en constante actualización.

Snort es un sistema de detección de intrusiones basado en red (NIDS) que implementa un motor de detección con la capacidad de realizar un análisis de tráfico en tiempo real; puede llevar a cabo análisis de protocolos, barrido de puertos, permite registrar, alertar y responder ante cualquier anomalía previamente definida, como pueden ser: patrones que corresponden a ataques, análisis de vulnerabilidades en el Sistema Operativo de un equipo o intentos de explotar alguna vulnerabilidad previamente identificada.

La arquitectura central de Snort se basa en 4 componentes:

- Decodificador de paquetes o Sniffer
- Preprocesador
- Motor de detección
- Sistema de alertas e informes

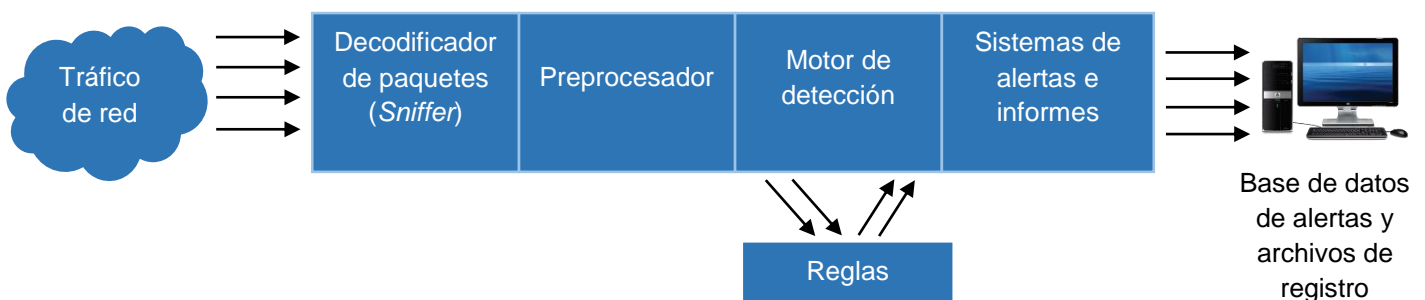


Figura. 3.1 Arquitectura de Snort.

Snort permite la captura y el procesado del tráfico de una red mediante los dos primeros componentes (Decodificador de paquetes y preprocesador), a continuación, realiza una comparación de los resultados contra el motor de detección (que depende del conjunto de reglas utilizado), para finalmente, generar las alertas y los informes necesarios a través del último componente.

Como se mencionó antes, el motor de detección depende de un conjunto de reglas (que están agrupados por categorías) para lograr el reconocimiento de usos indebidos mediante firmas de ataque.

Una regla de Snort se puede dividir en 2 secciones: **cabecera de la regla** y **opciones**:

- La **cabecera** contiene la acción de la regla (generación de archivo de registro o de una alerta), el protocolo (por ejemplo TCP, UDP, ICMP), además de puertos y direcciones IP de origen y destino.
- La sección de **opciones** contiene los mensajes e información necesaria que debe tener el paquete para que se cumpla la regla. Esta sección debe ir entre paréntesis.

Ejemplo:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "Escaneo ping con nmap"; flags: A;ack:0; reference: <referencia>; classtype: attempted-recon; sid: 628; rev: 1;)
```

En este documento se indica cómo instalar **Snort** en un equipo que cuenta con un Sistema Operativo **Ubuntu 14.04 LTS** y cuyas características de hardware son:

- Procesador Intel Core i5 de 4 núcleos a 3.40 GHz
- 8 GB de memoria RAM
- 520 GB de espacio en disco duro

Es importante resaltar que aunque Snort puede funcionar bien por sí solo, su rendimiento mejora bastante al instalarlo junto con **Barnyard2** y **PulledPork**. Más adelante se hablara a detalle de estas herramientas.

3.1.1 Requerimientos

Antes de iniciar, es necesario instalar los siguientes elementos, ya que son una serie de dependencias requeridas por Snort, y de lo contrario, es probable que ocurra algún error al momento de realizar la instalación:

- build-essential
- Pcap
- PCRE
- libdnet
- DAQ (Data AcQuisition)
- Zlibg

build-essentials es un conjunto de herramientas necesarias para poder compilar software, para instalarlas se ejecuta el siguiente comando.

```
usuario@equipo:~$ sudo apt-get install -y build-essential
```

A continuación, instalar las dependencias necesarias mediante el comando **apt-get**, dichas dependencias se hallan en los repositorios de Ubuntu. (En caso de que estas dependencias ya se encuentren instaladas y en su versión más reciente omitir este paso):

```
usuario@equipo:~$ sudo apt-get install -y libpcap-dev libpcrc3-dev  
libdumbnet-dev
```

Durante el desarrollo de este proyecto, se descargará código fuente de diversas herramientas, por lo que se creará en el **directorio raíz**, una carpeta con el nombre **“snort_src”** para mantener todos los códigos en la misma ubicación.

```
snort@snort:~$ sudo mkdir /snort_src  
snort@snort:~$ cd /snort_src/  
snort@snort:/snort_src$
```

Figura. 3.2 Carpeta snort_src.

Se descarga la versión más reciente de la biblioteca DAQ (desde el sitio oficial de Snort ^[1]) y se guardará en la carpeta **“snort_src”** (Que debió crearse en el paso anterior). Dicha biblioteca tiene algunos requerimientos que deben instalarse para que funcione de manera correcta.

Para instalar **bison** y **flex** (dependencias de DAQ) se utilizará el siguiente comando:

```
usuario@equipo:~$ sudo apt-get install -y bison flex
```

```
snort@snort:~$ sudo apt-get install -y bison flex  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.  
 libntdb1 python-ntdb  
Use 'apt-get autoremove' to remove them.  
Se instalarán los siguientes paquetes extras:  
 libbison-dev libfl-dev libsigsegv2 m4  
Paquetes sugeridos:  
 bison-doc  
Se instalarán los siguientes paquetes NUEVOS:  
 bison flex libbison-dev libfl-dev libsigsegv2 m4  
0 actualizados, 6 se instalarán, 0 para eliminar y 6 no actualizados.  
Necesito descargar 1 033 kB de archivos.  
Se utilizarán 2 809 kB de espacio de disco adicional después de esta operación.  
Des:1 http://mx.archive.ubuntu.com/ubuntu/ trusty/main libsigsegv2 amd64 2.10-2 [15.0 kB]  
Des:2 http://mx.archive.ubuntu.com/ubuntu/ trusty/main m4 amd64 1.4.17-2ubuntu1 [195 kB]  
Des:3 http://mx.archive.ubuntu.com/ubuntu/ trusty/main libfl-dev amd64 2.5.35-10.1ubuntu2 [17.2 kB]  
Des:4 http://mx.archive.ubuntu.com/ubuntu/ trusty/main flex amd64 2.5.35-10.1ubuntu2 [211 kB]  
Des:5 http://mx.archive.ubuntu.com/ubuntu/ trusty/main libbison-dev amd64 2:3.0.2.dfsg-2 [338 kB]  
Des:6 http://mx.archive.ubuntu.com/ubuntu/ trusty/main bison amd64 2:3.0.2.dfsg-2 [257 kB]  
Descargados 1 033 kB en 1seg. (750 kB/s)  
Seleccionando el paquete libsigsegv2:amd64 previamente no seleccionado.  
(Leyendo la base de datos ... 199894 ficheros o directorios instalados actualmente.)  
Preparing to unpack .../libsigsegv2_2.10-2_amd64.deb ...  
Unpacking libsigsegv2:amd64 (2.10-2) ...  
Seleccionando el paquete m4 previamente no seleccionado.  
Preparing to unpack .../m4_1.4.17-2ubuntu1_amd64.deb ...  
Unpacking m4 (1.4.17-2ubuntu1) ...  
Seleccionando el paquete libfl-dev:amd64 previamente no seleccionado.  
Preparing to unpack .../libfl-dev_2.5.35-10.1ubuntu2_amd64.deb ...  
Unpacking libfl-dev:amd64 (2.5.35-10.1ubuntu2) ...  
Seleccionando el paquete flex previamente no seleccionado.  
Preparing to unpack .../flex_2.5.35-10.1ubuntu2_amd64.deb ...  
Unpacking flex (2.5.35-10.1ubuntu2) ...
```

Figura. 3.3 Requerimientos de DAQ.

[1] <https://www.snort.org/>

Hecho lo anterior, se compilará e instalará la biblioteca DAQ, por lo que primero debe desempaquetarse el archivo previamente descargado.

```
usuario@equipo:~$ sudo tar -xvzf /snort_src/daq-2.0.6.tar.gz
```

```
snort@snort:/snort_src$ sudo tar -xvzf /snort_src/daq-2.0.6.tar.gz
daq-2.0.6/
daq-2.0.6/ChangeLog
daq-2.0.6/missing
daq-2.0.6/daq.dsp
daq-2.0.6/configure
daq-2.0.6/sfbpf/
daq-2.0.6/sfbpf/sf_bpf_printer.c
daq-2.0.6/sfbpf/IP6_misc.h
daq-2.0.6/sfbpf/sf_gencode.c
daq-2.0.6/sfbpf/lc.h
daq-2.0.6/sfbpf/ppp.h
daq-2.0.6/sfbpf/grammar.y
daq-2.0.6/sfbpf/sf_nametoaddr.c
daq-2.0.6/sfbpf/sf_bpf_filter.c
daq-2.0.6/sfbpf/sfbpf_dlt.h
daq-2.0.6/sfbpf/ethertype.h
daq-2.0.6/sfbpf/arcnet.h
daq-2.0.6/sfbpf/ieee80211.h
daq-2.0.6/sfbpf/sfbpf-int.h
daq-2.0.6/sfbpf/namedb.h
daq-2.0.6/sfbpf/Makefile.am
daq-2.0.6/sfbpf/runlex.sh
daq-2.0.6/sfbpf/atmuni31.h
daq-2.0.6/sfbpf/sf_redefines.h
daq-2.0.6/sfbpf/win32-stdinc.h
daq-2.0.6/sfbpf/sunatmpos.h
daq-2.0.6/sfbpf/sf_optimize.c
daq-2.0.6/sfbpf/sfbpf-int.c
daq-2.0.6/sfbpf/sfbpf.h
daq-2.0.6/sfbpf/gencode.h
daq-2.0.6/sfbpf/scanner.l
daq-2.0.6/sfbpf/bittypes.h
daq-2.0.6/sfbpf/sll.h
daq-2.0.6/sfbpf/nlpid.h
daq-2.0.6/sfbpf/Makefile.in
```

Figura. 3.4 Descomprimiendo el paquete DAQ.

Una vez que se desempaquetaron los archivos, se debe ingresar al directorio **“/snort_src/daq-2.0.6”** donde se encuentra el archivo necesario para iniciar la compilación del mismo. Para compilar e instalar, utilizar los siguientes comandos:

```
usuario@equipo:~$ cd daq-2.0.6
usuario@equipo:~$ sudo ./configure
```

Al terminar el proceso de compilación, se deberán generar los archivos necesarios para la instalación y después realizar la misma, para esto, se utilizan los siguientes comandos.

```
usuario@equipo:~$ sudo make
usuario@equipo:~$ sudo make install
```

Además de lo anterior, también se instalará la biblioteca **“zlibg”** usando el comando que se muestra a continuación. Dicha biblioteca es relevante ya que cumple con funciones para compresión de archivos.

```
usuario@equipo:~$ sudo apt-get install -y zlib1g-dev
```

```
snort@snort:/$ sudo apt-get install -y zlibg-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
 libntdb1 python-ntdb
Use 'apt-get autoremove' to remove them.
Se instalarán los siguientes paquetes NUEVOS:
  zlibg-dev
0 actualizados, 1 se instalarán, 0 para eliminar y 6 no actualizados.
Necesito descargar 183 kB de archivos.
Se utilizarán 454 kB de espacio de disco adicional después de esta operación.
Des:1 http://mx.archive.ubuntu.com/ubuntu/ trusty/main zlibg-dev amd64 1:1.2.8.dfsg-1ubuntu1 [183 kB]
Descargados 183 kB en 1seg. (143 kB/s)
Seleccionando el paquete zlibg-dev:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 200038 ficheros o directorios instalados actualmente.)
Preparing to unpack .../zlibg-dev_1%3a1.2.8.dfsg-1ubuntu1_amd64.deb ...
Unpacking zlibg-dev:amd64 (1:1.2.8.dfsg-1ubuntu1) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Configurando zlibg-dev:amd64 (1:1.2.8.dfsg-1ubuntu1) ...
snort@snort:/$
```

Figura. 3.5 Biblioteca de compresión.

3.1.2 Instalación

Para instalar Snort, es necesario descargar el paquete desde su sitio oficial ^[2] y posteriormente, utilizar los comandos que se muestran a continuación para compilarlo e instalarlo. El proceso es similar al realizado anteriormente para la instalación de DAQ.

Lo primero es situarse en el directorio “/snort src”, una vez en dicho directorio, es necesario desempaquetar la herramienta.

```
usuario@equipo:~$ sudo tar -xvzf snort-2.9.8.2.tar.gz
```

El proceso de compilado, generación de archivos para instalación y la instalación es igual al realizado para DAQ (se utilizan los mismos comandos), la única diferencia radica en el hecho de que el directorio donde se ejecutarán es “/snort-**src/snort-2.9.8.2**”, es por esta razón que se omitirán dichos comandos.

```
snort@snort:/snort_src/snort-2.9.8.2$ sudo ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... no
checking for mawk... mawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for style of include used by make... GNU
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking dependency style of gcc... gcc3
checking for gcc option to accept ISO C99... -std=gnu99
checking for gcc -std=gnu99 option to accept ISO Standard C... (cached) -std=gnu99
checking for gcc... (cached) gcc
checking whether we are using the GNU C compiler... (cached) yes
checking whether gcc accepts -g... (cached) yes
checking for gcc option to accept ISO C89... (cached) none needed
```

Figura. 3.6 Compilación de Snort.

[2] <https://www.snort.org/>

Lo siguiente es actualizar las bibliotecas compartidas, **ya que sin este paso, Snort mandará un error al tratar de ejecutarlo**. También se creará en el directorio “*/usr/sbin/snort*”, un enlace simbólico al archivo binario de snort que se encuentra ubicado en el siguiente directorio “*/usr/local/bin/snort*”.

```
usuario@equipo:~$ sudo ldconfig
usuario@equipo:~$ sudo ln -s /usr/local/bin/snort /usr/sbin/snort
```

```
snort@snort:/snort_src/snort-2.9.8.2$ sudo ldconfig
snort@snort:/snort_src/snort-2.9.8.2$ sudo ln -s /usr/local/bin/snort /usr/sbin/snort
snort@snort:/snort_src/snort-2.9.8.2$
```

Figura. 3.7 Actualización de bibliotecas compartidas y creación de enlace simbólico.

Para verificar que Snort funciona correctamente, se ejecutará como usuario estándar utilizando el parámetro **-V**, lo que indica la versión del ejecutable y la de sus dependencias.

```
usuario@equipo:~$ snort -V
```

```
snort@snort:/$ snort -V
-+> Snort! <*-
o" )~ Version 2.9.8.2 GRE (Build 335)
**** By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
      Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
      Copyright (C) 1998-2013 Sourcefire, Inc., et al.
      Using libpcap version 1.5.3
      Using PCRE version: 8.31 2012-07-06
      Using ZLIB version: 1.2.8
```

Figura. 3.8 Versión de Snort junto con los archivos de configuración que utiliza.

Con esto concluye el proceso de instalación de Snort.

3.1.3 Configuración como IDS

Para que Snort no se ejecute con privilegios de administrador (**root**), se crean un grupo y un usuario (**snort:snort**) sin privilegios, que serán los encargados de ejecutar el servicio.

Para ello se utilizan los siguientes comandos:

```
usuario@equipo:~$ sudo groupadd snort
usuario@equipo:~$ sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
```

Donde los parámetros utilizados indican lo siguiente:

- r: Crea una cuenta de sistema.
- s: Indica el nombre de la Shell que se utilizará, con */sbin/nologin* se impide ingresar al sistema con este usuario.
- c: Nombre completo del usuario.
- g: Grupo al que pertenecerá el nuevo usuario.

```
snort@snort:/$ sudo groupadd snort
snort@snort:/$ sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
```

Figura. 3.9 Creación del grupo y el usuario snort.

Además, también es necesario crear la estructura de archivos y directorios requeridos por Snort. Primero debe crearse la carpeta “**snort**” dentro del directorio /etc: “**/etc/snort**”, la cual se utilizará para ubicar los archivos de reglas y configuraciones, dentro de esta carpeta, deberán crearse un par de carpetas más, la primera llevará por nombre “**rules**”, para quedar de la siguiente manera “**/etc/snort/rules**”, la segunda será “**preproc_rules**” y su ruta debe quedar así “**/etc/snort/preproc_rules**”.

Todas las alertas se crean en la ruta “**/var/log**”, por lo que también es necesario crear la carpeta “**snort**” en dicha ubicación, quedando como se muestra a continuación: “**/var/log/snort**”. Además, será necesaria una carpeta más en la siguiente ruta: “**/usr/local/lib**”, dicha carpeta será para las reglas dinámicas y debe llevar el nombre “**snort_dynamicrules**”.

Hecho lo anterior, dentro del directorio “**/etc/snort/rules**”, se deben crear los archivos necesarios para las reglas de la lista blanca, la lista negra y las reglas locales (que son aquellas definidas manualmente por el administrador), estos archivos son:

- white_list.rules
- black_list.rules
- local.rules

Todo lo anterior se ilustra en la Figura 3.10.

```
snort@snort:/etc$ sudo mkdir /etc/snort/
snort@snort:/etc$ sudo mkdir /etc/snort/rules
snort@snort:/etc$ sudo mkdir /etc/snort/preproc_rules
snort@snort:/etc$ sudo mkdir /var/log/snort/
snort@snort:/etc$ sudo mkdir /usr/local/lib/snort_dynamicrules
snort@snort:/etc$ sudo touch /etc/snort/rules/white_list.rules
snort@snort:/etc$ sudo touch /etc/snort/rules/black_list.rules
snort@snort:/etc$ sudo touch /etc/snort/rules/local.rules
snort@snort:/etc$ █
```

Figura. 3.10 Creación de la estructura de archivos.

A continuación se modifican los permisos para los directorios que se acaban de crear, donde al usuario que es dueño del directorio o archivo y al grupo al que pertenece se les otorgan permisos para lectura, escritura y ejecución (**7 en su forma octal**) y al resto de los usuarios solamente para lectura y ejecución (**5 en su forma octal**).

Por último, se cambia el propietario de estos directorios y se indica que pertenecerán al usuario **snort** y al grupo del mismo nombre, ver Figura 3.11.

```
snort@snort:/$ sudo chmod -R 5775 /etc/snort/
snort@snort:/$ sudo chmod -R 5775 /var/log/snort/
snort@snort:/$ sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules/
snort@snort:/$ sudo chown -R snort:snort /etc/snort/
snort@snort:/$ sudo chown -R snort:snort /var/log/snort/
snort@snort:/$ sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules/
snort@snort:/$
```

Figura. 3.11 Se modifican tanto los permisos como el propietario de los directorios.

Al llegar a este punto, se deben copiar los siguientes archivos de configuración desde la carpeta donde se descomprimió Snort originalmente (***/snort_src/snort-2.9.8.2/etc***) hacia el directorio ***/etc/snort***.

- classification.config
- reference.config
- snort.conf
- threshold.conf
- gen-msg.map
- unicode.map

Hasta este momento, los archivos se encuentran en los siguientes directorios:

- Archivo binario de Snort: ***/usr/local/bin/snort*** (ligado a ***/usr/sbin/snort***)
- Archivo de configuración de Snort: ***/etc/snort/snort.conf***
- Directorio de bitácoras de Snort: ***/var/log/snort***
- Directorio con las reglas de Snort: ***/etc/snort/rules***

La estructura de archivos de Snort debe lucir como se muestra en la Figura 3.12.

```
snort@snort:/$ tree /etc/snort/
/etc/snort/
├── classification.config
├── gen-msg.map
├── preproc.rules
├── reference.config
├── rules
│   ├── blank_list.rules
│   ├── local.rules
│   └── update_list.rules
├── snort.conf
├── threshold.conf
└── unicode.map

2 directories, 9 files
snort@snort:/$
```

Figura. 3.12 Estructura de directorios de Snort.

A continuación es necesario editar el archivo principal de configuración de Snort, "***/etc/snort/snort.conf***". Cuando Snort se ejecuta y se le pasa como parámetro la ubicación de dicho archivo, se le está indicando al programa que se ejecute como un IDS.

Pero antes de hacer esto, se deben comentar todas las líneas que hacen referencia a los archivos de reglas de Snort, ya que en lugar de utilizar cada archivo de manera individual, se utilizará **PulledPork** para administrar los conjuntos de reglas, y por lo tanto, todas las reglas se combinarán en un solo archivo.

Para lograr esto, se utilizara el siguiente comando:

```
usuario@equipo:~$ sudo sed -i 's/include \${RULE\_PATH}/#include \${RULE\_PATH}' /etc/snort/snort.conf
```


Una vez que se ejecuta el comando anterior, se observa el resultado abriendo el archivo `/etc/snort/snort.conf` con un editor de texto y verificando que al inicio de cada línea aparece el símbolo “#” como se aprecia en la Figura 3.13.

```
#####  
# 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/local.rules  
  
#include $RULE_PATH/app-detect.rules  
#include $RULE_PATH/attack-responses.rules  
#include $RULE_PATH/backdoor.rules  
#include $RULE_PATH/bad-traffic.rules  
#include $RULE_PATH/blacklist.rules  
#include $RULE_PATH/botnet-cnc.rules  
#include $RULE_PATH/browser-chrome.rules  
#include $RULE_PATH/browser-firefox.rules  
#include $RULE_PATH/browser-ie.rules  
#include $RULE_PATH/browser-other.rules  
#include $RULE_PATH/browser-plugins.rules  
#include $RULE_PATH/browser-webkit.rules  
#include $RULE_PATH/chat.rules  
#include $RULE_PATH/content-replace.rules  
#include $RULE_PATH/ddos.rules  
#include $RULE_PATH/dns.rules  
#include $RULE_PATH/dos.rules  
#include $RULE_PATH/experimental.rules  
#include $RULE_PATH/exploit-kit.rules  
#include $RULE_PATH/exploit.rules  
#include $RULE_PATH/file-executable.rules  
#include $RULE_PATH/file-flash.rules  
#include $RULE_PATH/file-identify.rules  
#include $RULE_PATH/file-image.rules
```

Figura. 3.13 Resultado en el archivo snort.conf tras la ejecución del comando.

A continuación se editará manualmente el mismo archivo de configuración, para esto se puede utilizar el editor de su preferencia.

```
usuario@equipo:~$ sudo nano /etc/snort/snort.conf
```

Lo primero es ubicar dentro del archivo la línea en donde se encuentra la configuración para la red local (**HOME_NET**), aquí se debe poner el segmento o segmentos de red para monitoreo. Enseguida se debe ubicar la configuración de la red externa (**EXTERNAL_NET**), aquí el valor será cualquier otra red, en este ejemplo la red local (**HOME_NET**) monitoreada será 10.30.1.0/24, como se puede apreciar en la Figura 3.14.

```
GNU nano 2.2.6 Archivo: /etc/snort/snort.conf Modificado
# 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 any
ipvar HOME_NET 10.30.1.0/24

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

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

# 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

^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^C Pos actual
^X Salir          ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt     ^T Ortografía
```

Figura. 3.14 Configuración de las redes a monitorear.

El siguiente paso es modificar las rutas hacia los archivos de reglas que utilizará Snort, como se muestra a continuación:

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

```
GNU nano 2.2.6 Archivo: /etc/snort/snort.conf Modificado
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 [64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.0/24,205.188.5.0/24,205.188.7.0/24,205.185

# 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 /etc/snort/so_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
#####

^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^C Pos actual
^X Salir          ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt     ^T Ortografía
```

Figura. 3.15 Rutas a los archivos de las reglas.

Nota: En la Figura anterior, el signo “#” enfrente de algunas líneas, indica que se encuentran comentadas y que no serán consideradas al ejecutarse el archivo.

Finalmente se debe habilitar el archivo **local.rules** que es donde se pueden agregar reglas de forma manual, lo anterior se logra quitando el símbolo “#” al inicio de la línea “**include \$RULE_PATH/local.rules**” en el archivo de configuración “**snort.conf**”.

```
GNU nano 2.2.6 Archivo: /etc/snort/snort.conf Modificado
# output log_unified2: filename snort.log, limit 128, nostamp
# syslog
# output alert_syslog: LOG_AUTH LOG_ALERT
# pcap
# output log_tcpdump: tcpdump.log
# 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/local.rules
#include $RULE_PATH/app-detect.rules
#include $RULE_PATH/attack-responses.rules
#include $RULE_PATH/backdoor.rules
#include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules
^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^C Pos actual
^X Salir          ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt     ^T Ortografía
```

Figura. 3.16 Archivo **local.rules** habilitado para pruebas.

Una vez listo el archivo de configuración, se realizará una prueba para comprobar que todo funciona adecuadamente y que las referencias que se modificaron dentro del archivo “**/etc/snort/snort.conf**” sean correctas.

Además, de momento Snort no tiene ninguna regla cargada (los archivos referenciados en **snort.conf** están vacíos), por lo que se creará una regla en el archivo “**/etc/snort/rules/local.rules**”, con lo que se generará una alerta cada vez que se detecte un ping.

La regla utilizada será la siguiente:

```
alert icmp any any -> $HOME_NET any (msg: "ICMP Test"; sid:10000001; rev:001;)
```

```
snort@snort:/$ sudo echo "alert icmp any any -> '$HOME_NET' any (msg: "ICMP Test"; sid:10000001; rev:001;)" >> /etc/snort/rules/local.rules
snort@snort:/$ cat /etc/snort/rules/local.rules
alert icmp any any -> $HOME_NET any (msg: "ICMP Test"; sid:10000001; rev:001;)
snort@snort:/$
```

Figura. 3.17 Regla que se agregó en el archivo **local.rules**.

Para iniciar la prueba se utiliza el siguiente comando:

```
usuario@equipo:~$ sudo snort -T -c /etc/snort/snort.conf
```

Donde:

- T: indica que se trata de una prueba del archivo de configuración.
- c: indica la ruta del archivo de configuración

En la Figura 3.18 se aprecia el comando al momento de ser ejecutado:

```
snort@snort:/$ sudo snort -T -c /etc/snort/snort.conf
Running in Test mode

--== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined : [ 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 ]
PortVar 'SHELLCODE_PORTS' defined : [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined : [ 1024:65535 ]
PortVar 'SSH_PORTS' defined : [ 22 ]
PortVar 'FTP_PORTS' defined : [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined : [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined : [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 70
00:7001 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 9090:90
91 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'GTP_PORTS' defined : [ 2123 2152 3386 ]
Detection:
  Search-Method = AC-Full-Q
  Split Any/Any group = enabled
  Search-Method-Optimizations = enabled
  Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/local/lib/snort dynamicengine/libsf engine.so... done
Loading all dynamic detection libs from /usr/local/lib/snort_dynamicrules...
WARNING: No dynamic libraries found in directory /usr/local/lib/snort_dynamicrules.
  Finished Loading all dynamic detection libs from /usr/local/lib/snort_dynamicrules
Loading all dynamic preprocessor libs from /usr/local/lib/snort_dynamicpreprocessor/...
Loading dynamic preprocessor library /usr/local/lib/snort_dynamicpreprocessor/libsf_sdf_preproc.so... done
Loading dynamic preprocessor library /usr/local/lib/snort_dynamicpreprocessor/libsf_sntp_preproc.so... done
Loading dynamic preprocessor library /usr/local/lib/snort_dynamicpreprocessor/libsf_imap_preproc.so... done
Loading dynamic preprocessor library /usr/local/lib/snort_dynamicpreprocessor/libsf_reputation_preproc.so... done
Loading dynamic preprocessor library /usr/local/lib/snort_dynamicpreprocessor/libsf_gtp_preproc.so... done
```

Figura. 3.18 Inicio de la prueba.

En las Figuras 3.19 y 3.20 se muestran el resultado al momento de cargar las reglas y el mensaje de éxito en caso de que todo haya marchado bien, respectivamente.

```
+++++
Initializing rule chains...
1 Snort rules read
  1 detection rules
  0 decoder rules
  0 preprocessor rules
1 Option Chains linked into 1 Chain Headers
0 Dynamic rules
+++++
+-----[Rule Port Counts]-----+
|      tcp      udp      icmp      ip
|  src         0         0         0         0
|  dst         0         0         0         0
|  any         0         0         1         0
|  nc          0         0         1         0
|  s+d         0         0         0         0
+-----+

```

Figura. 3.19 Verifica si Snort carga la regla correctamente.


```
Rule application order: activation->dynamic->pass->drop->sdrop->reject->alert->log
Verifying Preprocessor Configurations!

[ Port Based Pattern Matching Memory ]
[ Number of patterns truncated to 20 bytes: 0 ]

--== Initialization Complete ==--

-*> Snort! <*-
o" )-
  ""
  Version 2.9.8.2 GRE (Build 335)
  By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
  Copyright (C) 2014-2015 Cisco and/or its affiliates. All rights reserved.
  Copyright (C) 1998-2013 Sourcefire, Inc., et al.
  Using libpcap version 1.5.3
  Using PCRE version: 8.31 2012-07-06
  Using ZLIB version: 1.2.8

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.6 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>

Snort successfully validated the configuration!
Snort exiting
snort@snort:/$
```

Figura. 3.20 Mensaje de prueba exitosa del archivo de configuración.

Ahora que se comprobó que Snort carga la configuración y las reglas de manera correcta, se iniciará en modo IDS y se le indicará que la salida se muestre en la consola.

Se ejecutará en línea de comandos utilizando los siguientes parámetros:

- | | |
|---------------------------------|---|
| -A console | La opción “ <i>console</i> ” imprime alertas en forma rápida. |
| -q | Modo silencioso, no despliega banners ni reporte de estatus. |
| -u snort | Ejecuta Snort como el usuario “ snort ”. |
| -g snort | Ejecuta Snort con el grupo “ snort ”. |
| -c /etc/snort/snort.conf | La ruta al archivo snort.conf . |
| -i eth0 | La interfaz por la que escuchará el tráfico de red. |

```
usuario@equipo:~$ sudo /usr/local/bin/snort -A console -q -u snort -g snort
-c /etc/snort/snort.conf -i eth0
```

```
snort@snort:/$ sudo /usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i eth0
```

Figura. 3.21 Snort ejecutándose.

Desde la computadora donde se está ejecutando Snort, se realiza una prueba mediante un PING hacia un equipo cuya dirección IP es 10.30.1.25.

```
snort@snort-IDS:~$ ping 10.30.1.25
PING 10.30.1.25 (10.30.1.25) 56(84) bytes of data:
64 bytes from 10.30.1.25: icmp_seq=1 ttl=64 time=0.716 ms
64 bytes from 10.30.1.25: icmp_seq=2 ttl=64 time=0.432 ms
64 bytes from 10.30.1.25: icmp_seq=3 ttl=64 time=0.302 ms
64 bytes from 10.30.1.25: icmp_seq=4 ttl=64 time=0.347 ms
64 bytes from 10.30.1.25: icmp_seq=5 ttl=64 time=0.416 ms
64 bytes from 10.30.1.25: icmp_seq=6 ttl=64 time=0.409 ms
64 bytes from 10.30.1.25: icmp_seq=7 ttl=64 time=0.395 ms
64 bytes from 10.30.1.25: icmp_seq=8 ttl=64 time=0.363 ms
64 bytes from 10.30.1.25: icmp_seq=9 ttl=64 time=0.381 ms
64 bytes from 10.30.1.25: icmp_seq=10 ttl=64 time=0.405 ms
64 bytes from 10.30.1.25: icmp_seq=11 ttl=64 time=0.357 ms
64 bytes from 10.30.1.25: icmp_seq=12 ttl=64 time=0.388 ms
64 bytes from 10.30.1.25: icmp_seq=13 ttl=64 time=0.418 ms
^C
--- 10.30.1.25 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 11999ms
rtt min/avg/max/mdev = 0.302/0.409/0.716/0.098 ms
snort@snort-IDS:~$
```

Figura. 3.22 Ping hacia un equipo con dirección IP 10.30.1.25.

Lo anterior genera una serie de alertas, ya que como se definió previamente, la actividad detectada coincide con la regla creada y dicha dirección IP se encuentra en el segmento de red que está siendo monitoreado por Snort.

La salida de Snort se muestra a continuación.

```
snort@snort-IDS:~$ sudo /usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i eth0
05/23-18:23:20.585468 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:20.585795 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:21.584715 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:21.585039 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:22.586912 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:22.587155 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:23.585929 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:23.586198 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:24.584929 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:24.585270 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:25.584611 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:25.584951 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:26.494388 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.1 -> 10.30.1.58
05/23-18:23:26.584749 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:26.585050 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:27.584695 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:27.584980 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:28.584634 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:28.584939 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:29.497416 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.1 -> 10.30.1.58
05/23-18:23:29.584824 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:29.585132 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:30.585165 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:30.585436 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:31.584657 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:31.584955 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
05/23-18:23:32.584727 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.12 -> 10.30.1.25
05/23-18:23:32.585038 00000001:1 "ICMP Test" [**] [Priority: 0] {ICMP} 10.30.1.25 -> 10.30.1.12
```

Figura. 3.23 Alertas de Snort.

Las bitácoras de Snort se almacenan en el directorio: */var/log/snort*, y para poder leer su contenido se utiliza el comando:

```
usuario@equipo:~$ sudo snort -r snort.log.xxxxxxxxxx
```

```
snort@snort-IDS:~$ ls /var/log/snort/
snort.log.1464045527 snort.log.1464045786
snort@snort-IDS:~$
```

Figura. 3.24 Bitácoras de Snort.

3.1.4 Configuración del preprocesador HTTP Inspect

Http Inspect es un pre-procesador que brinda la capacidad de examinar el tráfico HTTP, lo que incluye *URL*'s, *cookies*, parámetros en *GET/POST*, entre otros. Por lo que sin lugar a dudas, se trata de un módulo bastante útil.

Para habilitar este pre-procesador, se debe modificar el archivo de configuración de Snort presente en la ruta */etc/snort/snort.conf*. Dicho pre-procesador cuenta con un módulo para configuración global (*http_inspect Global*) y otro para una configuración específica de cada servidor web (*http_inspect Server*) que se encuentre en el segmento de red monitoreado.

Cada uno de estos módulos cuenta con determinadas opciones que permiten controlar el comportamiento del pre-procesador http inspect, en la Figura 3.25 se muestra un ejemplo para las configuraciones "global" y "server", en el Anexo A se pueden consultar las opciones de configuración disponibles junto con su descripción.

```
usuario@equipo:~$ sudo nano /etc/snort/snort.conf
```

```
GNU nano 2.2.6 Archivo: /etc/snort/snort.conf
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 CS
  chunk_length 500000 \
  server_flow_depth 0 \
  client_flow_depth 0 \
  post_depth 65495 \
  oversized_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 85
  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 \

^G Ver ayuda      ^C Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^O Pos actual
^X Salir          ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt     ^T Ortografía
```

Figura. 3.25 Configuración del preprocesador Http inspect.

3.2 Fase 3: Instalación y configuración de Barnyard2

Barnyard2 es un intérprete de código abierto para los archivos binarios *unified2* generados por Snort. Su principal utilidad es la de permitir que Snort escriba datos en disco de manera eficiente y dejar la tarea de interpretar los datos en binario a un proceso diferente, lo que permitirá que Snort se dedique exclusivamente a la captura de tráfico.

Barnyard2 leerá los datos desde los archivos antes mencionados y posteriormente los copiará a una base de datos de MySQL.

3.2.1 Requerimientos

Es necesario contar con un servidor y un cliente de MySQL, la instalación de los mismos se hace mediante el siguiente comando:

```
usuario@equipo:~$ sudo apt-get install -y mysql-server libmysqlclient-dev  
mysql-client autoconf libtool
```

Al iniciar el proceso de instalación se pedirá crear una contraseña para el usuario **root** de MySQL (la cual deberá escribirse dos veces), en este ejemplo se utilizará la contraseña **MYSQLROOTPASSWORD**.

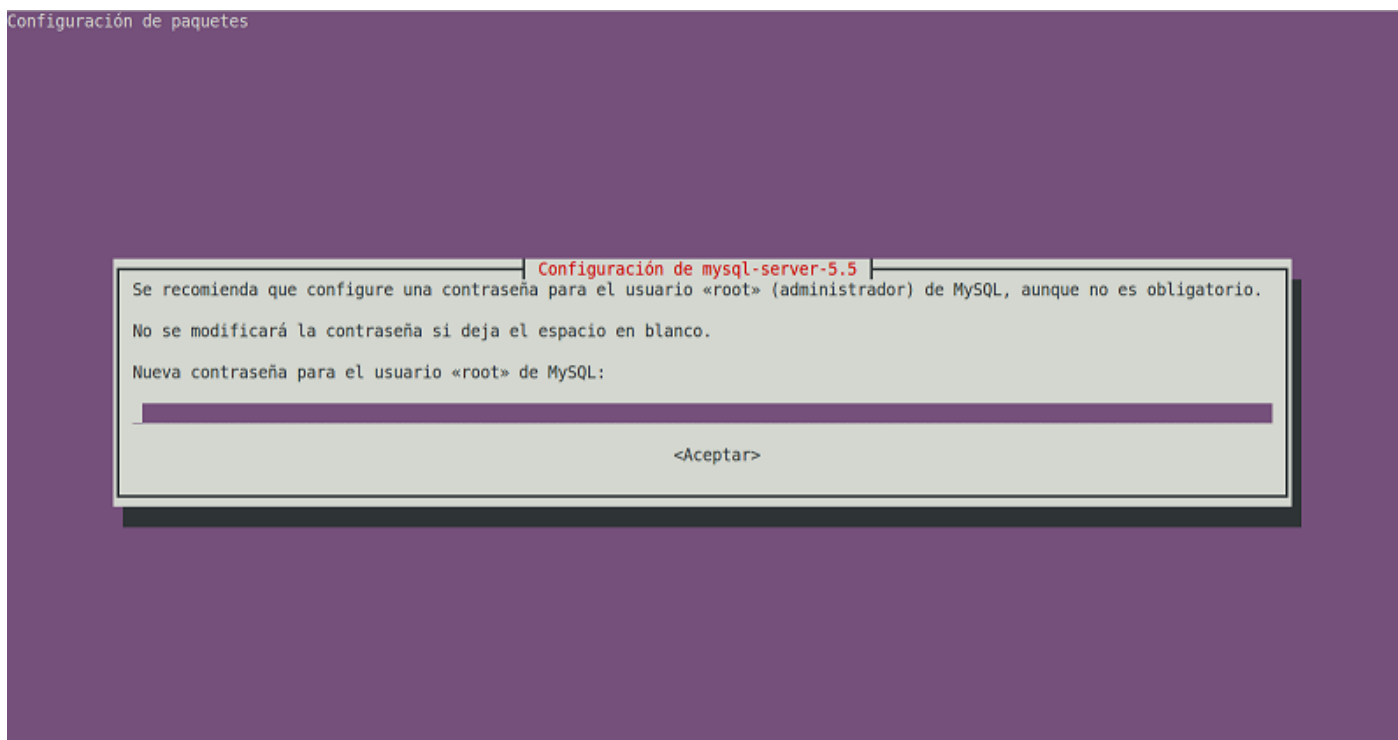


Figura. 3.26 Creación de la contraseña del usuario root en MySQL.

Posteriormente se creará un usuario para Snort en MySQL, cuya contraseña será **MYSQLSNORTPASSWORD** (ES IMPORTANTE NOTAR LA DIFERENCIA ENTRE ESTA CONTRASEÑA Y LA UTILIZADA POR EL USUARIO ROOT DE MYSQL).

Es necesario configurar Snort para que los datos capturados se almacenen en formato binario y que de esta forma puedan ser procesados por barnyard2, esto se consigue editando el archivo de configuración de Snort que se encuentra en el directorio */etc/snort/snort.conf*, en dicho archivo debe agregarse la siguiente línea.

```
output unified2: filename snort.u2, limit 128
```

Quedando de la siguiente manera:



Figura. 3.27 Archivo *snort.conf* modificado.

3.2.2 Instalación

Para comenzar con la instalación de Barnyard2 primero se debe descargar la versión más reciente desde el sitio oficial^[3], y como se ha hecho hasta este momento, guardarlo en el directorio */snort_src*; debido a que el código fuente viene comprimido, será necesario desempaquetarlo y compilarlo para poder instalar el programa.

Para descomprimir el paquete se utilizarán los mismos comandos que se han venido utilizando a lo largo de este ejemplo por lo que se omitirán.

Una vez desempaquetado el programa, será necesario utilizar el siguiente comando para compilarlo:

```
usuario@equipo:~$ sudo autoreconf -fvi -I ./m4
```

Dependiendo de la arquitectura del sistema operativo (x86 o x86_64), es necesario instalar la versión adecuada de la biblioteca utilizada por MySQL, para esto, se puede ejecutar alguno de los siguientes comandos:

```
usuario@equipo:~$ sudo ./configure --with-mysql --with-mysql-libraries=/usr/lib/x86_64-linux-gnu
usuario@equipo:~$ sudo ./configure --with-mysql --with-mysql-libraries=/usr/lib/i386-linux-gnu
```

Para finalizar el proceso de instalación ejecutar los comandos **make** y **make install**.

Para que Barnyard2 funcione de manera correcta, se deben crear las siguientes carpetas y archivos recordando asignar como propietario al usuario y al grupo **snort**.

```
usuario@equipo:~$ sudo cp /snort_src/barnyard2-2-1.13/etc/barnyard2.conf /etc/snort
usuario@equipo:~$ sudo mkdir /var/log/barnyard2
usuario@equipo:~$ sudo touch /var/log/snort/barnyard2.waldo
usuario@equipo:~$ sudo touch /etc/snort/sid-msg.map
```

3.2.3 Configuración

Para configurar barnyard2 correctamente, lo primero es crear el usuario **snort** dentro de MySQL junto con la base de datos necesaria para que Barnyard2 guarde las alertas generadas.

Después de ingresar cada uno de los siguientes comandos, se solicitará la contraseña del usuario **root** de MySQL (**MYSQLROOTPASSWORD**).

Lo primero es ingresar el siguiente comando para crear la base de datos (cuyo nombre será **snort**) en MySQL:

```
usuario@equipo:~$ echo "create database snort;" | mysql -u root -p
```

[3] <https://github.com/firnsy/barnyard2/releases>

Lo siguiente es dar de alta la contraseña “**MYSQLSNORTPASSWORD**” que será utilizada por el usuario “**snort**” de MySQL:

```
usuario@equipo:~$ echo "grant create, insert, select, delete, update on snort.* to
snort@localhost identified by 'MYSQLSNORTPASSWORD'" | mysql -u root -p
```

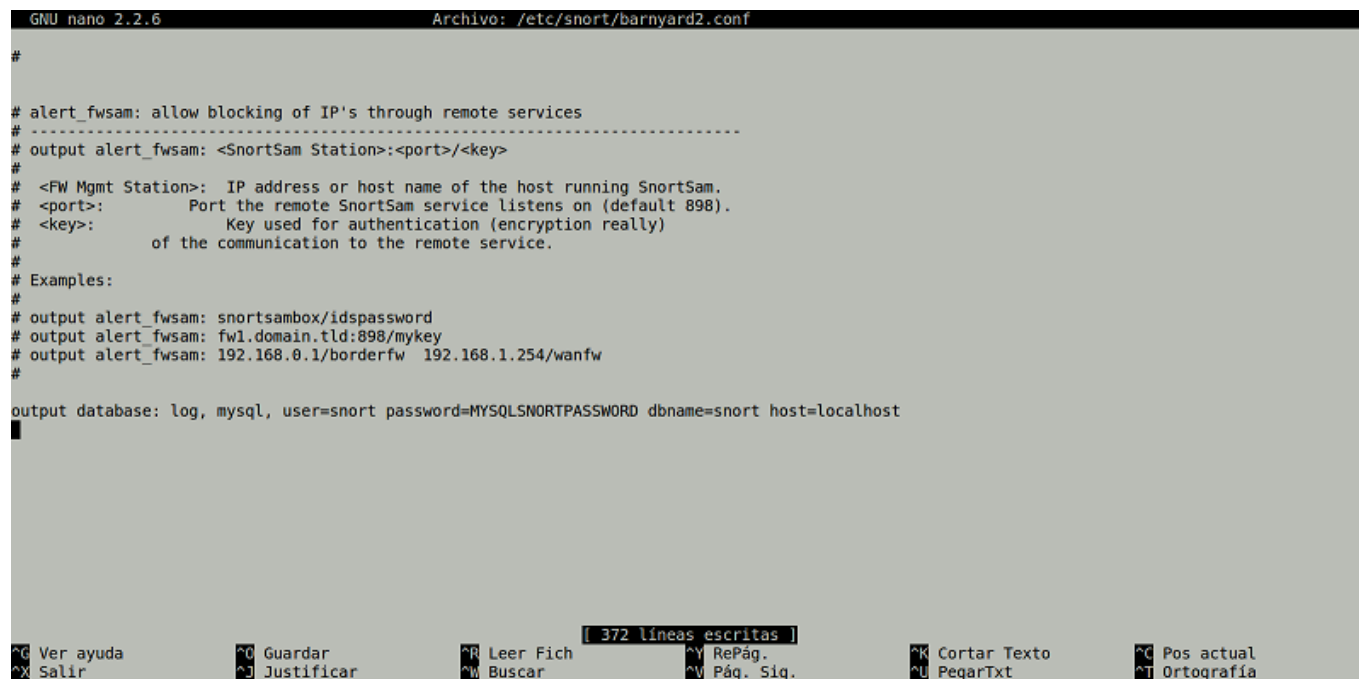
```
usuario@equipo:~$ mysql -u root -p -D snort < /snort_src/barnyard2-2-
1.13/schemas/create_mysql
```

También es necesario indicarle a Barnyard2 cómo se conectará con la base de datos “**snort**” que se encuentra dentro de MySQL, por lo tanto, se editará el archivo “**barnyard2.conf**” (en la ruta: “**/etc/snort/**”) agregando la siguiente línea al final del mismo:

**“output database: log, mysql, user=snort
password=MYSQLSNORTPASSWORD dbname=snort host=localhost”**

```
snort@snort-IDS:/snort_src/barnyard2-2-1.13/etc$ sudo nano /etc/snort/barnyard2.conf
```

Figura. 3.28 Abriendo el archivo **barnyard2.conf**.



```
GNU nano 2.2.6 Archivo: /etc/snort/barnyard2.conf
#
# alert_fwsam: allow blocking of IP's through remote services
# -----
# output alert_fwsam: <SnortSam Station>:<port>/<key>
#
# <FW Mgmt Station>: IP address or host name of the host running SnortSam.
# <port>: Port the remote SnortSam service listens on (default 898).
# <key>: Key used for authentication (encryption really)
# of the communication to the remote service.
#
# Examples:
#
# output alert_fwsam: snortsambox/idspassword
# output alert_fwsam: fw1.domain.tld:898/mykey
# output alert_fwsam: 192.168.0.1/borderfw 192.168.1.254/wanfw
#
output database: log, mysql, user=snort password=MYSQLSNORTPASSWORD dbname=snort host=localhost
[ 372 líneas escritas ]
^G Ver ayuda      ^C Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^C Pos actual
^X Salir         ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt      ^T Ortografía
```

Figura. 3.29 Modificando el archivo **barnyard2.conf**.

Debido a que la contraseña se encuentra almacenada en texto plano en el archivo **barnyard2.conf**, se deben modificar los permisos de dicho archivo por motivos de seguridad, con esto se busca evitar que cualquier usuario pueda tener acceso a él y leerlo.

```
usuario@equipo:~$ sudo chmod o-r /etc/snort/barnyard2.conf
```

En este punto se realizará una prueba para verificar que tanto **Snort** como **Barnyard2** funcionan adecuadamente. En primer lugar se ejecuta Snort en modo de alerta.

```
usuario@equipo:~$ sudo /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i eth0
```

```
snort@snort-IDS:/snort_src/barnyard2-2-1.13/etc$ sudo /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i eth0
```

Figura. 3.30 Snort en ejecución.

Al realizar un ping a la interfaz eth0 (donde Snort está escuchando el tráfico de red), se crea un nuevo archivo con el nombre snort.u2.xxxxxxxxxx en el directorio */var/log/snort*, lo que indica que todo marcha de manera correcta.

El siguiente paso es ejecutar Barnyard2 para pedirle que busque los eventos en este archivo y que los cargue en la base de datos “**snort**”:

```
usuario@equipo:~$ sudo barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort
```

Donde:

-c /etc/snort/barnyard2.conf	La ruta hacia el archivo <i>barnyard2.conf</i>
-d /var/log/snort	El directorio donde se buscarán los archivos enviados por Snort
-f snort.u2	El nombre del archivo dentro del directorio anterior (snort.u2.xxxxxxxxxx)
-w /var/log/snort/barnyard2.waldo	La ubicación del archivo Waldo
-u snort	Ejecutar Barnyard2 como el usuario especificado
-g snort	Ejecutar Barnyard2 como el grupo especificado

Enseguida se revisará la base de datos (snort) para corroborar que Barnyard2 almacenó los eventos:

```
usuario@equipo:~$ mysql -u snort -p -D snort -e "select count(*) from event"
```

Utilizar la contraseña del usuario “**snort**” de MySQL cuando sea requerida.

```
snort@snort-IDS:/$ mysql -u snort -p -D snort -e "select count(*) from event"
Enter password:
+-----+
| count(*) |
+-----+
|        62 |
+-----+
snort@snort-IDS:/$
```

Figura. 3.31 Comprobación de los eventos en la base de datos.

Como se puede apreciar, existen 62 registros (*count* regresa un resultado mayor que 0) lo que indica que tanto Snort como Barnyard2 están instalados y configurados correctamente.

3.3 Fase 4: Instalación y configuración de PuledPork

PuledPork es el script encargado de actualizar las reglas utilizadas por Snort, y en caso de no contar con ellas, las instala.

3.3.1 Requerimientos

Como todo lo que se ha instalado hasta este momento, **PuledPork** cuenta con una serie de dependencias que son necesarias antes de continuar, los requerimientos para **PuledPork** se instalan de la siguiente manera.

```
usuario@equipo:~$ sudo apt-get install -y libcrypt-ssleay-perl liblwp-useragent-determined-perl
```

3.3.2 Instalación

Como con los paquetes anteriores, se deberá descargar y guardar el código fuente en el directorio **/snort_src**, para después desempaquetar, compilar e instalar PuledPork, la descarga se puede realizar desde su sitio oficial ^[4].

Para desempaquetar el código fuente se utiliza el comando **tar** de la misma manera que se ha venido haciendo hasta ahorita, una vez terminado el proceso, se copia el script **pulledpork.pl** (el corazón de *pulledpork*) a la siguiente ubicación **/usr/local/bin**.

```
usuario@equipo:~$ sudo cp /snort_src/pulledpork-0.7.0/pulledpork.pl /usr/local/bin
usuario@equipo:~$ sudo chmod +x /usr/local/bin/pulledpork.pl
```

Enseguida se deben copiar los siguientes archivos:

- disablesid.conf
- dropsid.conf
- enablesid.conf
- modifysid.conf
- pulledpork.conf

Desde el directorio **etc/** de **PuledPork** hacia el directorio de configuración de Snort (**/etc/snort/**).

Hecho lo anterior, se crearán carpetas y archivos que son requeridos por PuledPork y también se comprobará que el script funciona de manera correcta revisando su versión:

```
usuario@equipo:~$ sudo mkdir /etc/snort/tmp
usuario@equipo:~$ sudo mkdir /etc/snort/rules/iplists
usuario@equipo:~$ sudo touch /etc/snort/rules/iplists/default.blacklist
usuario@equipo:~$ /usr/local/bin/pulledpork.pl -V
```

[4] <https://code.google.com/p/pulledpork/>

3.3.3 Configuración

Antes de continuar, es necesario registrarse en el sitio oficial de snort para obtener un **Oinkcode** único que permite descargar los conjuntos de reglas. La configuración de PuledPork se hará editando el archivo "**pulledpork.conf**" que debe encontrarse en la ruta **/etc/snort**.

Es necesario poner el **código oink** obtenido al registrarse en el sitio de Snort, en cualquier parte del archivo **pulledpork.conf** donde se encuentre la cadena **<oinkcode>**.

A continuación se presenta una lista con las modificaciones necesarias dentro del archivo **pulledpork.conf**.

Agregar el código oink donde sea necesario.

Modificar el enlace en la línea 21 a:

<https://snort.org/downloads/community/|community-rules.tar.gz>Community

Quitar el comentario para poder utilizar la versión libre de las reglas de Emerging Threats.

Cambiar a: temp_path=/etc/snort/tmp/

Cambiar a: rule_path=/etc/snort/rules/snort.rules

Cambiar a: local_rules=/etc/snort/rules/local.rules

Cambiar a: sid_msg=/etc/snort/sid-msg.map

Cambiar a: config_path=/etc/snort/snort.conf

Cambiar a: distro=Ubuntu-10-4

Cambiar a: black_list=/etc/snort/rules/iplists/default.blacklist

Cambiar a: IPRVersion=/etc/snort/rules/iplists/

Quitar el comentario y cambiar a: enablesid=/etc/snort/enablesid.conf

Quitar el comentario y cambiar a: dropsid=/etc/snort/dropsid.conf

Quitar el comentario y cambiar a: disablesid=/etc/snort/disablesid.conf

Quitar el comentario y cambiar a: modifiesid=/etc/snort/modifiesid.conf

Nota: debido a que la última versión de reglas disponible para Snort es la 2.9.7.6; es necesario indicar dentro del archivo que se utilizará dicha versión, sin importar que la versión de Snort utilizada sea una más actual.

Una vez realizados todos los cambios, se ejecuta **PulledPork** de forma manual utilizando los siguientes parámetros para verificar que funciona de manera correcta.

```
usuario@equipo:~$ sudo /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf
```

Donde:

-l Escribe una bitácora detallada en **/var/log**
-c /etc/snort/snort.conf La ruta al archivo de configuración **pulledpork.conf**

```
snort@snort-IDS:/usr/local/lib$ sudo /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf

http://code.google.com/p/pulledpork/

  _____
 /         \
|         |  PulledPork v0.7.0 - Swine Flu!
|         |
|         |
|         |  Copyright (C) 2009-2013 JJ Cummings
|         |  cummingsj@gmail.com
|         |  66
|         |  (")
 \         /  Rules give me wings!
  \_____/

-----

Checking latest MD5 for snortrules-snapshot-2976.tar.gz...
  They Match
  Done!
Rules tarball download of community-rules.tar.gz...
IP Blacklist download of http://labs.snort.org/feeds/ip-filter.blf...
Reading IP List...
Checking latest MD5 for opensource.gz...
  They Match
  Done!
Checking latest MD5 for emerging.rules.tar.gz...
  They Match
  Done!
Prepping rules from opensource.gz for work...
  Done!
Prepping rules from community-rules.tar.gz for work...
  Done!
Prepping rules from emerging.rules.tar.gz for work...
  Done!
Prepping rules from snortrules-snapshot-2976.tar.gz for work...
  Done!
Reading rules...
Reading rules...
```

Figura. 3.32 Comprobación del archivo *pulledpork.pl*.

```
Modifying Sids...
  Done!
Processing /etc/snort/enablesid.conf...
  Modified 0 rules
  Done
Processing /etc/snort/dropsid.conf...
  Modified 0 rules
  Done
Processing /etc/snort/disablesid.conf...
  Modified 0 rules
  Done
Setting Flowbit State...
  Enabled 98 flowbits
  Done
Writing /etc/snort/rules/snort.rules...
  Done
Generating sid-msg.map...
  Done
Writing v1 /etc/snort/sid-msg.map...
  Done
Writing /var/log/sid_changes.log...
  Done

Rule Stats...
  New:-----46
  Deleted:---16
  Enabled Rules:----28266
  Dropped Rules:----0
  Disabled Rules:---24191
  Total Rules:-----52457
IP Blacklist Stats...
  Total IPs:-----55909

Done
Please review /var/log/sid_changes.log for additional details
Fly Piggy Fly!
```

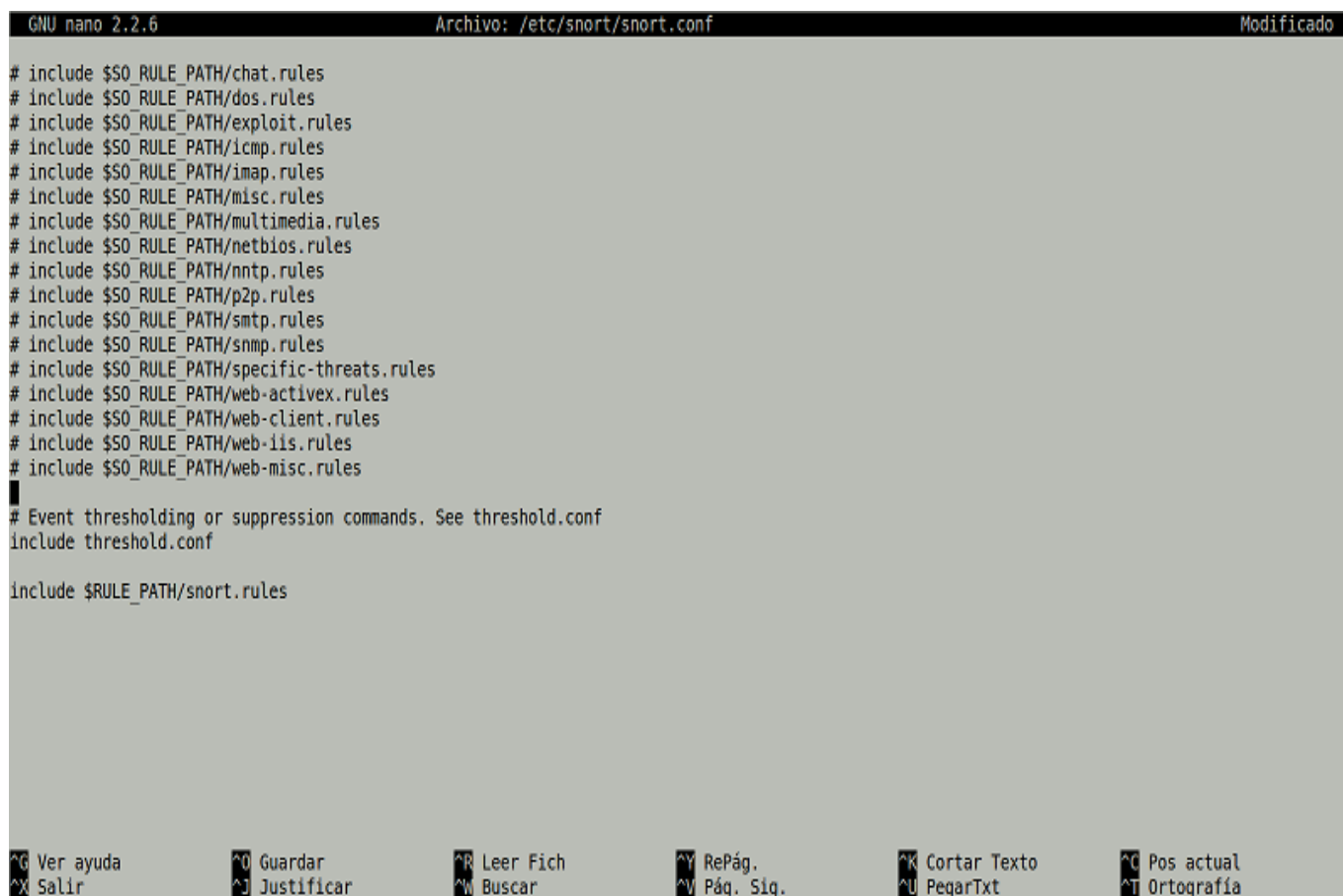
Figura. 3.33 Salida correcta al ejecutar *pulledpork.pl*.

Cuando **PulledPork** se ejecuta exitosamente, se podrá ver el archivo **snort.rules** en el directorio **/etc/snort/rules**.

Debido a que PulledPork combina todos los conjuntos de reglas que descarga en un sólo archivo, es necesario indicarle a Snort que incluya dicho archivo al momento de ejecutarse, de lo contrario Snort no cargará estas reglas cuando se inicie.

Esto se consigue añadiendo la línea: **include \$RULE_PATH/snort.rules** en el archivo **snort.conf**. Para esto se debe editar nuevamente el archivo **snort.conf** y agregar lo siguiente al final del mismo (en una línea nueva):

```
include $RULE_PATH/snort.rules
```



```
GNU nano 2.2.6 Archivo: /etc/snort/snort.conf Modificado
# 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

include $RULE_PATH/snort.rules

^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y RePág.      ^K Cortar Texto  ^C Pos actual
^X Salir         ^J Justificar   ^W Buscar      ^V Pág. Sig.   ^U PegarTxt     ^T Ortografía
```

Figura. 3.34 Línea agregada al archivo **snort.conf**.

3.3.3.1 Configuración de PulledPork para inicio automático

Para que PulledPork se ejecute de forma automática (para que las reglas de Snort se estén actualizando constantemente), debe añadirse una tarea nueva (cron) que ejecute su script, esto se consigue agregando la siguiente línea después del comando **crontab**.

```
usuario@equipo:~$ sudo crontab -e  
01 04 * * * /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf -l
```

```
snort@snort-IDS:/$ sudo crontab -e  
no crontab for root - using an empty one  
  
Select an editor. To change later, run 'select-editor'.  
 1. /bin/ed  
 2. /bin/nano <---- easiest  
 3. /usr/bin/vim.tiny  
  
Choose 1-3 [2]:
```

Figura. 3.35 Selección del editor de su preferencia.

```
GNU nano 2.2.6 Archivo: /tmp/crontab.bsNjLL/crontab Modificado  
  
# Edit this file to introduce tasks to be run by cron.  
#  
# Each task to run has to be defined through a single line  
# indicating with different fields when the task will be run  
# and what command to run for the task  
#  
# To define the time you can provide concrete values for  
# minute (m), hour (h), day of month (dom), month (mon),  
# and day of week (dow) or use '*' in these fields (for 'any').#  
# Notice that tasks will be started based on the cron's system  
# daemon's notion of time and timezones.  
#  
# Output of the crontab jobs (including errors) is sent through  
# email to the user the crontab file belongs to (unless redirected).  
#  
# For example, you can run a backup of all your user accounts  
# at 5 a.m every week with:  
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/  
#  
# For more information see the manual pages of crontab(5) and cron(8)  
#  
# m h dom mon dow command  
  
01 04 * * * /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf -l  
  
^G Ver ayuda      ^O Guardar      ^R Leer Fich    ^Y RePág.  
^X Salir          ^J Justificar   ^W Buscar       ^V Pág. Sig.  
^C Cortar Texto   ^U PegarTxt     ^T Pos actual  
^_ Ortografía
```

Figura. 3.36 Añadiendo el archivo *pulledpork.pl* a crontab.

3.4 Fase 5: Instalación y configuración de BASE

Basic Analysis and Security Engine (BASE) es una aplicación web que brinda una interfaz amigable para analizar las alertas de los eventos almacenados en la base de datos de Snort, lo que permite una administración más sencilla.

3.4.1 Requerimientos

Para que **BASE** pueda funcionar de manera correcta, es necesario instalar las dependencias necesarias entre las que se incluyen **apache2** (servidor web), el módulo de **php5** e **Image_Graph** (permite generar gráficas).

```
usuario@equipo:~$ sudo apt-get install -y apache2 libapache2-mod-php5  
php5 php5-mysql php5-common php5-gd php5-cli php-pear  
usuario@equipo:~$ sudo pear install -f Image_Graph
```

```
snort@snort-IDS:/$ sudo apt-get install -y apache2 libapache2-mod-php5 php5 php5-mysql php5-common php5-gd php5-cli php-pear  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.  
 libntdb1 python-ntdb  
Use 'apt-get autoremove' to remove them.  
Se instalarán los siguientes paquetes extras:  
 apache2-bin apache2-data libapr1 libaprutil1 libaprutil1-dbd-sqlite3  
 libaprutil1-ldap php5-json php5-readline  
Paquetes sugeridos:  
 apache2-doc apache2-suexec-pristine apache2-suexec-custom apache2-utils  
 php5-dev php5-user-cache  
Se instalarán los siguientes paquetes NUEVOS:  
 apache2 apache2-bin apache2-data libapache2-mod-php5 libapr1 libaprutil1  
 libaprutil1-dbd-sqlite3 libaprutil1-ldap php-pear php5 php5-cli php5-common  
 php5-gd php5-json php5-mysql php5-readline
```

Figura. 3.37 Instalación de los requerimientos de BASE.

Una vez completada la instalación de **Image_Graph** se ignoran las advertencias que muestra, siempre y cuando las últimas tres líneas digan que la instalación fue correcta (**Install ok**).

```
snort@snort-IDS:/$ sudo pear install -f Image_Graph  
WARNING: failed to download pear.php.net/Image_Graph within preferred state "stable", will instead download version 0.8.0, stability "alpha"  
WARNING: failed to download pear.php.net/Image_Canvas within preferred state "stable", will instead download version 0.3.5, stability "alpha"  
Did not download optional dependencies: pear/Numbers Roman, pear/Numbers Words, use --alldeps to download automatically  
WARNING: "pear/Image_Color" is deprecated in favor of "pear/Image_Color2"  
pear/Image_Graph can optionally use package "pear/Numbers Roman"  
pear/Image_Graph can optionally use package "pear/Numbers Words"  
downloading Image_Graph-0.8.0.tgz ...  
Starting to download Image_Graph-0.8.0.tgz (370,682 bytes)  
.....done: 370,682 bytes  
downloading Image_Canvas-0.3.5.tgz ...  
Starting to download Image_Canvas-0.3.5.tgz (55,263 bytes)  
...done: 55,263 bytes  
downloading Image_Color-1.0.4.tgz ...  
Starting to download Image_Color-1.0.4.tgz (9,656 bytes)  
...done: 9,656 bytes  
install ok: channel://pear.php.net/Image_Color-1.0.4  
install ok: channel://pear.php.net/Image_Canvas-0.3.5  
install ok: channel://pear.php.net/Image_Graph-0.8.0  
snort@snort-IDS:/$
```

Figura. 3.38 Instalando *Image Graph*.

Posteriormente se debe descargar ADODB ^[5] (se trata de un conjunto de bibliotecas de bases de datos para PHP y Python) en su versión 5.18 (ADODB 5.19 no es compatible con la versión de BASE utilizada para este proyecto) y guardarlo en la carpeta **/snort_src**.

Una vez descomprimido el paquete, se debe mover el contenido de la carpeta **adodb5** generada hacia **/var/adodb** (de no existir dicha carpeta, crearla).

3.4.2 Instalación

Descargar y descomprimir el código de BASE ^[6] de la misma manera que se ha venido realizando hasta este punto (por este motivo se omitirán los comandos utilizados para ello).

```
snort@snort-IDS:/snort_src$ ll
total 8636
drwxr-xr-x 7 root root 4096 may 24 15:56 /
drwxr-xr-x 25 root root 4096 may 24 15:41 ./
drwxr-xr-x 2 root root 4096 may 24 15:52 adodb5/
-rw-rw-r-- 1 snort snort 544455 may 24 15:47 adodb518a.tgz
drwxr-xr-x 10 root root 4096 may 24 10:07 barnyard2-2-1.13/
-rw-r--r-- 1 root root 958567 mar 5 2010 base-1.4.5.tar.gz
drwxr-xr-x 6 snort snort 4096 may 23 17:08 daq-2.0.6/
-rw-r--r-- 1 root root 514687 mar 28 23:56 daq-2.0.6.tar.gz
drwxr-xr-x 5 501 staff 4096 sep 11 2013 pulledpork-0.7.0/
-rw-r--r-- 1 root root 39294 ene 23 12:14 pulledpork-0.7.0.tar.gz
drwxr-xr-x 10 root root 4096 may 23 17:14 snort-2.9.8.2/
-rw-r--r-- 1 root root 6311793 mar 28 23:50 snort-2.9.8.2.tar.gz
-rw-r--r-- 1 root root 435017 may 23 18:56 v2-1.13.tar.gz
snort@snort-IDS:/snort_src$ sudo tar -xvzf base-1.4.5.tar.gz
```

Figura. 3.39 Preparando la instalación de BASE.

Debido a que Ubuntu 14 utiliza el servidor web apache versión 2.4, la carpeta de instalación para **BASE** será **“/var/www/html/base/”**, es por este motivo que se moverán todos los archivos desde la carpeta que se creó al descomprimir el paquete descargado hacia la ubicación antes mencionada.

Adicionalmente dentro de los archivos que fueron movidos hay uno que tiene el nombre **base_conf.php.dist** el cual debe renombrarse **base_conf.php** como se muestra en la Figura 3.40.

```
snort@snort-IDS:/snort_src/base-1.4.5$ sudo mkdir /var/www/html/base/
snort@snort-IDS:/snort_src/base-1.4.5$ sudo mv * /var/www/html/base/
snort@snort-IDS:/snort_src/base-1.4.5$ sudo mv /var/www/html/base/base_conf.php.dist /var/www/html/base/base_conf.php
snort@snort-IDS:/snort_src/base-1.4.5$
```

Figura. 3.40 Instalando BASE.

[5] <http://sourceforge.net/projects/adodb/files/adodb-php5-only/>
[6] <http://sourceforge.net/projects/secureideas/>

A continuación se debe editar el archivo de configuración de **BASE** modificándolo de la siguiente manera:

```
$BASE_urlpath = '/base';  
$DBlib_path = '/var/adodb/';  
$alert_dbname = 'snort';  
$alert_host = 'localhost';  
$alert_port = '';  
$alert_user = 'snort';  
$alert_password = 'MYSQLSNORTPASSWORD';
```

También se deben modificar los permisos al directorio de **BASE**, además de prevenir que el archivo **base_conf.php** sea leído por cualquier usuario, debido a que dicho archivo contiene la contraseña del usuario “**snort**” de MySQL en texto plano y finalmente se reinicia el servicio de Apache como se muestra en la siguiente Figura.

```
snort@snort-IDS:/$ sudo chown -R www-data:www-data /var/www/html/base/  
snort@snort-IDS:/$ sudo chmod o-r /var/www/html/base/base_conf.php  
snort@snort-IDS:/$ sudo service apache2 restart  
* Restarting web server apache2  
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message  
[ OK ]  
snort@snort-IDS:/$ █
```

Figura. 3.41 Cambiando permisos al directorio de BASE.

3.4.3 Configuración

Los pasos necesarios para la configuración se realizan a través de la interfaz web:

Primero, navegar a la URL `http://dirección_IP_asignada_a_snort/base/index.php` y hacer clic en el enlace que dice “**Setup page**”.



Figura. 3.42 Inicio de la configuración de BASE.

En la página que se despliega, hacer clic en el botón **“Create BASE AG”** que se encuentra en la esquina superior derecha de la página, esto creará las tablas necesarias para que el programa trabaje de manera correcta.

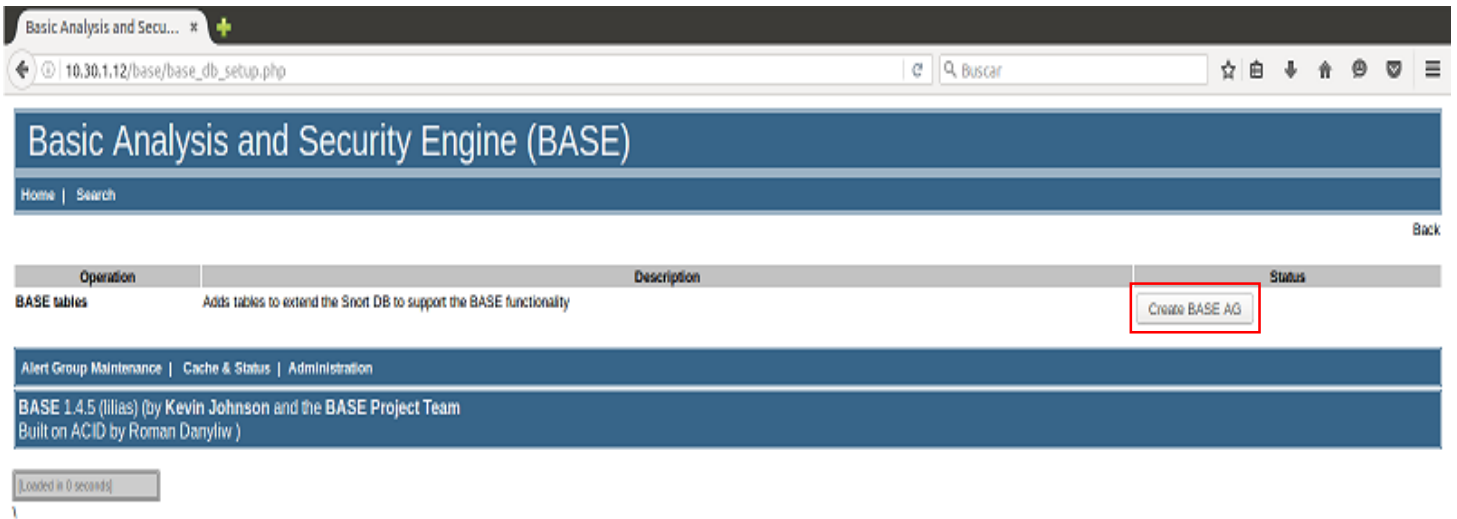


Figura. 3.43 Creando las tablas necesarias para BASE.

BASE mostrará una serie de mensajes donde se indica que las tablas fueron creadas manera correcta; finalmente hacer clic en el enlace **“Main page”**.

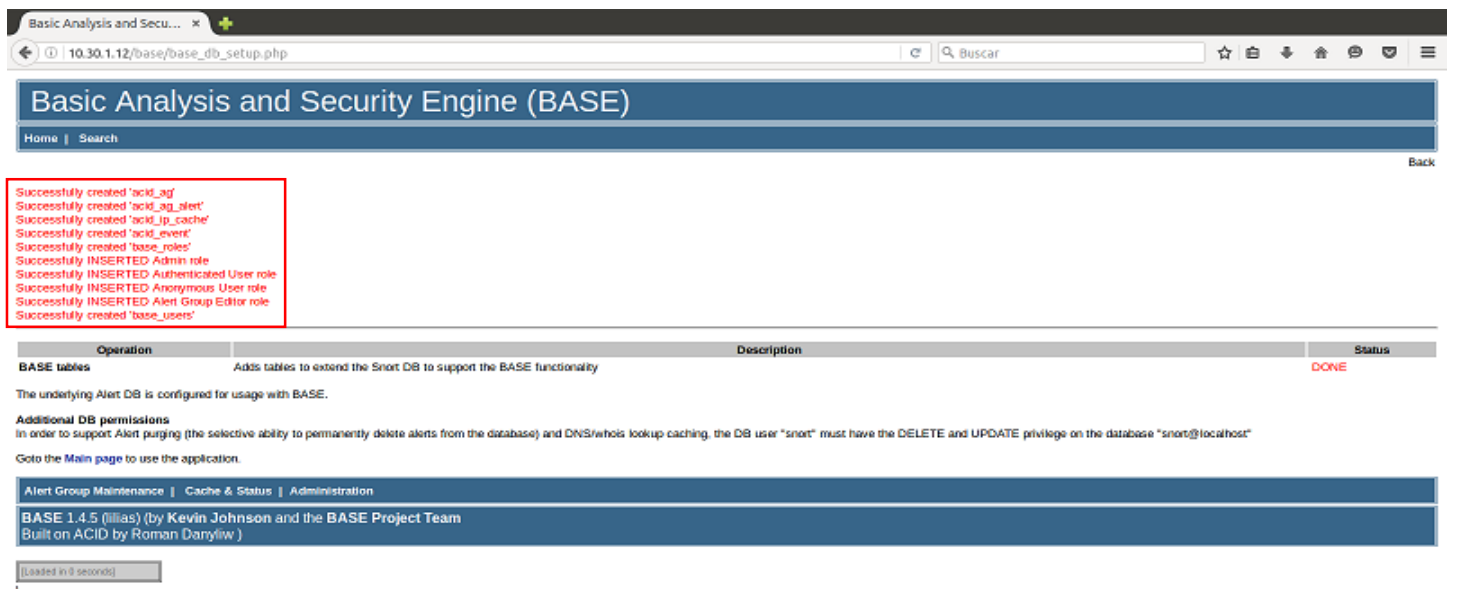


Figura. 3.44 Mensaje de tablas creadas correctamente.

Con esto, **BASE** se encuentra configurado y se puede acceder a él para analizar el comportamiento del IDS (Snort), ver el tráfico capturado y las alertas de los ataques detectados.

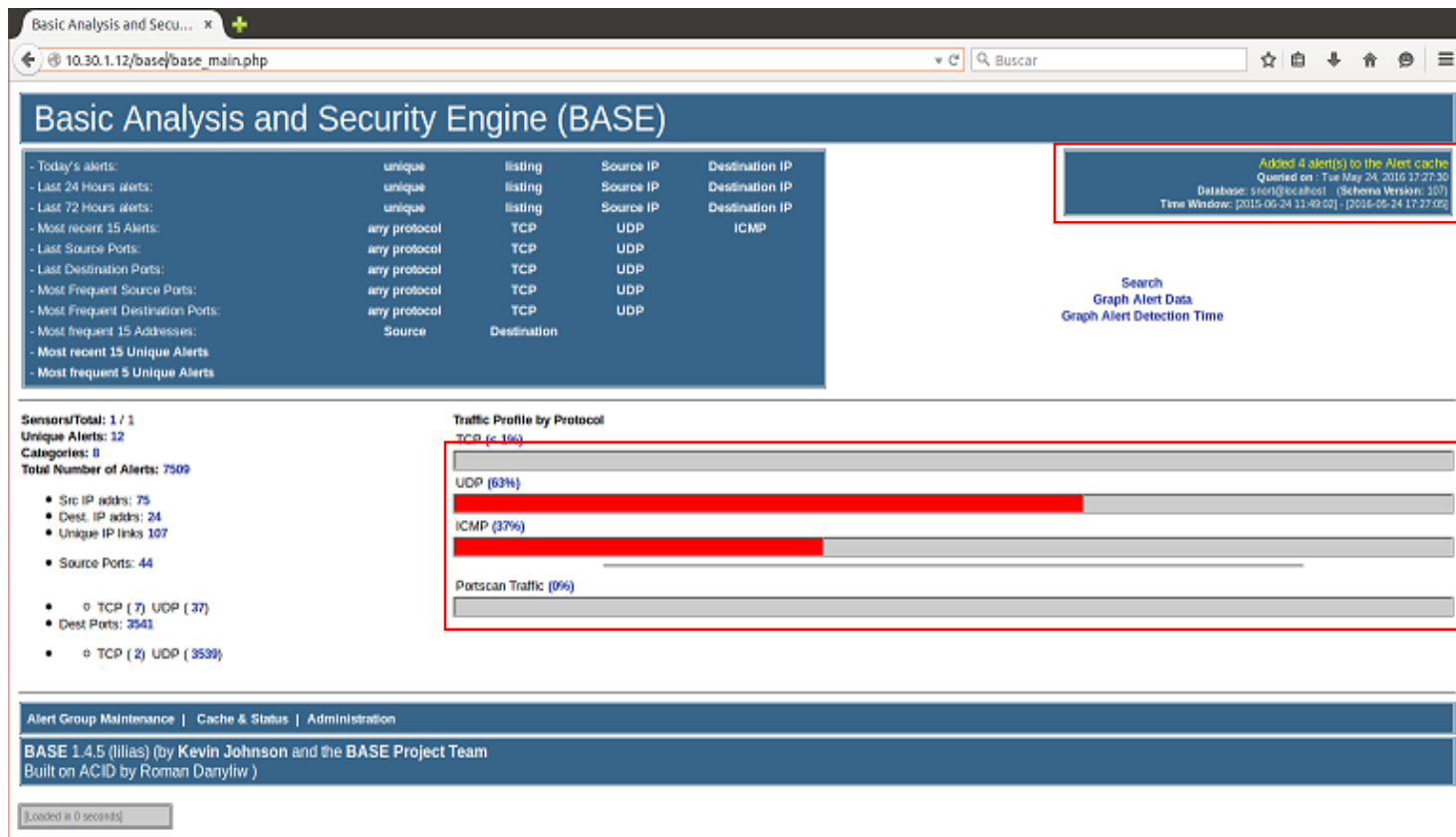


Figura. 3.45 BASE configurado y funcional.



Capítulo 4

Pruebas y Resultados

4.1 Pruebas

Para verificar el correcto funcionamiento del servidor IDS, se realizaron una serie de pruebas desde el propio sensor, y desde un equipo diferente que pertenece a la misma red cuya dirección IP es 10.30.1.58.

Estas pruebas consistieron en navegar a algunos sitios web con dominios de nivel superior que se sabe generan alertas de SNORT. Dichos sitios incluyen aquellos que representan una violación de políticas de seguridad, como son los **.xxx** o los que se consideran inseguros debido a que comúnmente son catalogados como focos de distribución de malware, por ejemplo, **.tk** y **.su**.

The screenshot shows the BASE web interface. At the top, there's a navigation bar with 'Home | Search' and a search box. Below that, a message states 'Added 1 alert(s) to the Alert cache' with a timestamp 'Queried on: Tue May 24, 2016 17:33:21'. A table on the left shows search criteria: Meta Criteria: any; IP Criteria: Source = 10.30.1.58; UDP Criteria: any; Payload Criteria: any. A 'Summary Statistics' box on the right lists metrics like Sensors, Unique Alerts, and Unique IP links. The main area displays a table of alerts, with a red box highlighting a specific alert: ID #6 (1-12992), signature 'ET DNS DNS Query to a .tk domain - Likely Hostile', timestamp 2016-05-24 17:08:55, source address 10.30.1.58, and destination address 10.30.12.235:53. Below this, a larger table lists 17 alerts with columns for ID, Signature, Timestamp, Source Address, Dest. Address, and Layer 4 Proto.

ID	Signature	Timestamp	Source Address	Dest. Address	Layer 4 Proto
#0 (1-13526)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:01	10.30.1.58 52303	8.8.8.8:53	UDP
#1 (1-13525)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:01	10.30.1.58 52303	10.30.12.235:53	UDP
#2 (1-12996)	[snort] ET DNS DNS Query to a .tk domain - Likely Hostile	2016-05-24 17:08:55	10.30.1.58 51348	8.8.8.8:53	UDP
#3 (1-12995)	[snort] ET DNS DNS Query to a .tk domain - Likely Hostile	2016-05-24 17:08:55	10.30.1.58 51348	10.30.12.235:53	UDP
#4 (1-12994)	[snort] ET DNS DNS Query to a .tk domain - Likely Hostile	2016-05-24 17:08:55	10.30.1.58 63308	10.30.12.235:53	UDP
#5 (1-12993)	[snort] ET DNS DNS Query to a .tk domain - Likely Hostile	2016-05-24 17:08:55	10.30.1.58 50994	8.8.8.8:53	UDP
#6 (1-12992)	[snort] ET DNS DNS Query to a .tk domain - Likely Hostile	2016-05-24 17:08:55	10.30.1.58 50994	10.30.12.235:53	UDP
#7 (1-12981)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:08:23	10.30.1.58 54681	10.30.12.235:53	UDP
#8 (1-12980)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:08:23	10.30.1.58 65178	10.30.12.235:53	UDP
#9 (1-12979)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:08:23	10.30.1.58 55281	10.30.12.235:53	UDP
#10 (1-12892)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:29	10.30.1.58 62858	8.8.8.8:53	UDP
#11 (1-12891)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:29	10.30.1.58 62858	10.30.12.235:53	UDP
#12 (1-12890)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:29	10.30.1.58 52448	10.30.12.235:53	UDP
#13 (1-12889)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:29	10.30.1.58 53259	8.8.8.8:53	UDP
#14 (1-12888)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:06:29	10.30.1.58 53259	10.30.12.235:53	UDP
#15 (1-12887)	[snort] MALWARE-OTHER dns request with long host name segment - possible data exfiltration attempt	2016-05-24 17:06:27	10.30.1.58 54127	8.8.8.8:53	UDP
#16 (1-12886)	[snort] MALWARE-OTHER dns request with long host name segment - possible data exfiltration attempt	2016-05-24 17:06:27	10.30.1.58 54127	10.30.12.235:53	UDP
#17 (1-12884)	[snort] MALWARE-OTHER dns request with long host name segment - possible data exfiltration attempt	2016-05-24 17:06:27	10.30.1.58 50230	10.30.12.235:53	UDP

Figura. 4.1 Alertas de actividad generadas por el equipo con dirección IP 10.30.1.58.

En la Figura anterior se puede observar que el equipo con dirección IP 10.30.1.58 realizó una serie de peticiones DNS (al puerto 53 de los destinos) y que generó las siguientes alertas:

- ET POLICY DNS Query For XXX Adult Site Top Level Domain
- ET DNS Query to a .tk domain – Likely Hostile

Al abrir una de las peticiones DNS a un sitio para adultos, se puede observar información detallada del paquete capturado, en donde se puede identificar que el sitio visitado fue: **filmepornoht.xxx**.

Basic Analysis and Secu... x

10.30.1.12/base/base_gry_alert.php?submit=%230(1-13526)&sort_order=

Meta Criteria: any
IP Criteria: Source = 10.30.1.58 ...Clear...
UDP Criteria: any
Payload Criteria: any

Added 7 alert(s) to the Alert cache

Alert #0
[First] >> Next #1-[1-12886]

ID #	Time	Triggered Signature
1 - 19526	2016-05-24 17:26:01	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain

Meta

Sensor Address	Interface	Filter
Dave:NULL	NULL	none

Alert Group: none

IP

Source Address	Dest. Address	Ver	Hdr Len	TOS	length	ID	fragment	offset	TTL	chksum
10.30.1.58	8.8.8.8	4	20	0	62	7012	no	0	128	996 = 0x3e4

Options: none

UDP

source port	dest port	length
52908	53	42

Payload

Plain Display length = 34

Download of Payload

```
000 : 2A 8E 01 00 00 01 00 00 00 00 00 00 00 00 0c 66 69 6c *.....fil  
010 : 60 65 70 67 72 67 67 68 64 03 78 78 78 00 00 01 mepornoht.xxx...  
020 : 00 01 ..
```

Download in pcap format

Figura. 4.2 Revisión de petición DNS a sitio para adultos.

En otra de las peticiones realizadas desde el mismo equipo, esta vez a un sitio con dominio **.tk**, se pudo verificar que la consulta realizada iba dirigida al sitio **ngni1.tk** mediante el servidor DNS de Google con dirección IP pública 8.8.8.8 - como se puede ver en la Figura 4.3 -.

The screenshot shows the alert details for ID #1-12996. The alert was triggered at 2016-05-24 17:08:55 with the signature "[snort] ET DNS DNS Query to a .tk domain - Likely Hostile". The IP section shows a source address of 10.30.1.58 and a destination address of 8.8.8.8. The UDP section shows source port 51348 and destination port 53. The payload section shows a plain display of "length = 26" and a hex dump of the payload: "000 : 6c 54 01 00 00 01 00 00 00 00 00 05 6e 67 6e 17.....ngn
010 : 69 31 02 74 6a 00 00 1c 00 01 ii.tk.....".

Figura. 4.3 Revisión de petición DNS a dominio .tk.

Para finalizar se navegó a un sitio de dominio **.su** desde el sensor IDS con dirección IP 10.30.1.12, actividad que generó el indicador **ET POLICY DNS Query for .su TLD (Soviet Union) Often Malware Related**.

The screenshot shows the alert list page in the IDS interface. The page title is "Basic Analysis and Security Engine (BASE)". The alert list shows several alerts, with the first two highlighted in red:

ID	Signature	Timestamp	Source Address	Dest. Address	Layer 4 Proto
#0(1-13636)	[url] [snort] ET POLICY DNS Query for .su TLD (Soviet Union) Often Malware Related	2016-05-24 17:35:30	10.30.1.12:55895	10.30.12.235:53	UDP
#1(1-13637)	[url] [snort] ET POLICY DNS Query for .su TLD (Soviet Union) Often Malware Related	2016-05-24 17:35:30	10.30.1.12:55895	10.30.12.235:53	UDP
#2(1-12947)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:07:25	10.30.1.12:48432	10.30.12.235:53	UDP
#3(1-12948)	[snort] ET POLICY DNS Query For XXX Adult Site Top Level Domain	2016-05-24 17:07:25	10.30.1.12:48432	10.30.12.235:53	UDP
#4(1-9517)	[snort] MALWARE-OTHER dns request with long host name segment - possible data exfiltration attempt	2015-06-30 16:37:25	10.30.1.12:47984	10.30.12.235:53	UDP
#5(1-9518)	[snort] MALWARE-OTHER dns request with long host name segment - possible data exfiltration attempt	2015-06-30 16:37:25	10.30.1.12:47984	10.30.12.235:53	UDP

At the bottom of the alert list, there is an "ACTION" section with a dropdown menu and buttons for "Selected", "ALL on Screen", and "Entire Query".

Figura. 4.4 Revisión de petición DNS a dominio .su.

Finalmente al ver el contenido de las peticiones hechas, se identificó que el sitio en cuestión fue: **3dnews.su**.

The screenshot shows a web interface for a network security tool. At the top, there's a search bar with the URL '10.30.1.12/base/base qry_alert.php?submit=%230{1-13636}&sort_order='. Below it, a sidebar contains filters for IP Criteria (Source = 10.30.1.12), UDP Criteria (any), and Payload Criteria (any). A red message states 'Added 1 alert(s) to the Alert cache'. The main area displays an alert with the following details:

ID #	Time	Triggered Signature
1 - 13636	2016-05-24 17:35:30	[url] [snort] ET POLICY DNS Query for .su TLD (Soviet Union) Often Malware Related

Below the alert, there are sections for Meta, IP, and UDP details.

Sensor Address	Interface	Filter
Dave:NULL	NULL	none

Source Address	Dest. Address	Ver	Hdr Len	TOS	length	ID	fragment	offset	TTL	chksum
10.30.1.12	10.30.12.235	4	20	0	58	19458	no	0	64	52349 = 0xcc7d

source port	dest port	length
55895	53	39

The Payload section shows a hex dump of the data, with a red box highlighting the domain name:

```
000 | BD CE 01 00 00 01 00 00 00 00 00 03 77 77 77 | .....WWW
010 | 06 33 64 6E 65 77 73 02 73 75 00 00 01 00 01 | .3dnews.su.....
```

Figura. 4.5 Visita a sitio 3dnews.su.

Esto se considera evidencia suficiente para considerar que, de forma general, la aplicación funciona de manera adecuada y que únicamente será necesario hacer modificaciones al tipo de alertas que mostrará dependiendo de las necesidades del cliente.

4.2 Resultados

Después del periodo de pruebas, del cual se dio un breve ejemplo en este trabajo, y el cual consistió de tener el sensor monitoreando la red local durante un periodo de una semana. Se obtuvieron resultados positivos y satisfactorios, lo que llevó a la decisión de implementar Snort en una serie de equipos dedicados, con mejores características de hardware, y con lo que se buscó obtener un mejor desempeño.

Dichos equipos fueron desplegados paulatinamente en las redes de los clientes de la empresa. Actualmente contamos con 13 sensores que utilizan Snort como motor IDS distribuidos en las redes de 4 clientes diferentes.

Se cumplió satisfactoriamente con el objetivo de tener una herramienta funcional para el servicio de monitoreo de red, sin que esto representara un impacto económico significativo, ya que únicamente es necesario invertir el hardware que se desea utilizar lo que permite generar ahorro para la empresa.

También se documentaron todos los procedimientos utilizados durante el desarrollo de este proyecto con la finalidad de generar un manual de instalación y configuración de los servicios y herramientas aquí descritas. Esto con el fin de tener material de consulta para el personal operativo; con lo que se busca que puedan replicar el proyecto en cualquier equipo de cómputo dependiendo de las necesidades de la empresa.

Es importante recalcar que gracias a la flexibilidad de SNORT, ahora el personal del SOC cuenta con una aplicación que le permite realizar sus actividades de gestión y respuesta a incidentes de una forma más productiva y eficiente, mejorando así la calidad del servicio que se brinda. Un ejemplo de esto es que hemos personalizado los sensores con los que contamos, ya que dependiendo del tráfico presente en cada red, se ha podido descartar actividad considerada como falso positivo y excluir tráfico generado por determinados equipos o segmentos de red de una manera relativamente sencilla.



Conclusiones

Implementar un **Sistema de Detección de Intrusiones** resulta en una herramienta muy útil que agrega capacidad de prevención y alerta anticipada a la seguridad en la red de una organización, ya que al monitorear de forma constante, recopila y analiza información de manera automática lo que ayuda a detectar la ocurrencia de eventos que intenten comprometer la seguridad de la red de alguna manera.

Es importante mencionar que para lograr lo anterior, además del motor IDS, se habilitaron una serie de características que buscan facilitar la operación y mantenimiento del dispositivo, por ejemplo:

- El complemento encargado de descargar automáticamente las reglas utilizadas por el IDS, con lo que se evita trabajar con un equipo desactualizado aumentando así su efectividad.
- La interfaz gráfica a la que se accede mediante un navegador web, en donde se despliega toda la información capturada por el IDS de una manera ordenada y amigable, haciendo más sencillo detectar actividad sospechosa.

Con esto lo que se busca, es permitir que el personal correspondiente (en este caso el equipo azul del SOC), se enfoque principalmente en sus tareas de monitoreo, detección y respuesta de incidentes.

Para llevar a cabo un proyecto de este tipo, se debe de contar con conocimientos técnicos sólidos en diversas áreas, tales como:

- Hardware: Las características físicas del equipo en el que se pudiera implementar este proyecto, pueden depender de factores como son; la cantidad de equipos que se planean desplegar en una red o el tamaño y cantidad de actividad de la misma.
- Sistemas Operativos: Sobre todo aquellos que tienen como base el kernel de Linux, ya que en muchas ocasiones resulta mucho más eficiente trabajar directamente sobre la consola de comandos que mediante una interfaz gráfica.
- Redes y seguridad informática: Para determinar el mejor punto de ubicación de estos dispositivos y realizar las configuraciones necesarias de manera adecuada, tomando en consideración el segmento o segmentos de red donde se colocarán.

Trabajar con aplicaciones de código libre puede resultar complejo en muchas ocasiones debido a la falta de información que puede haber respecto a la herramienta que se desea utilizar. Para implementar un servicio de este tipo, necesité documentarme de una manera muy extensa y así poder entender que era lo que se debía hacer. Afortunadamente, Snort cuenta con una amplia comunidad que le brinda soporte, por lo que existe bastante información que resulta muy útil.

Aun así, durante el desarrollo del proyecto surgieron algunos problemas, sobre todo relacionados con las configuraciones utilizadas, ya que al ser una serie de servicios que trabajan en conjunto, en ocasiones detalles muy sencillos como por ejemplo, una URL que ya no es válida dentro de un archivo fundamental para la descarga de reglas, no permitía obtener el resultado deseado.

Al encontrarme con este tipo de situaciones me di cuenta que es de suma importancia leer cuidadosamente la información que viene en los manuales y dentro de los archivos de configuración, ya que frecuentemente, en esos mismos archivos es donde se hallan los consejos o soluciones para los problemas de configuración que se pudieran tener.

Considero que a pesar de los contratiempos que surgieron, con paciencia y perseverancia, al final cumplí de manera satisfactoria con el propósito del proyecto, ya que desde su implementación y hasta la fecha, los servidores funcionan de manera óptima y han cumplido, hasta el momento, con los objetivos para los que fueron pensados.

Aunado al beneficio monetario que representan las herramientas de código libre, en mi opinión, resulta bastante útil poder realizar modificaciones a discreción y de acuerdo a nuestras necesidades, sin tener que solicitar algún tipo de permiso o licencia a un tercero como muchas veces ocurre con los aplicativos comerciales.

Otra de las ventajas surge de la naturaleza modular del proyecto, por ejemplo, se pueden implementar diferentes tipos de interfaces web para el monitoreo. Aunque en esta ocasión se utilizó BASE, existen otras opciones como lo es SNORBY. Siendo esto muy útil ya que cada interfaz ofrece información de manera diferente y en muchas ocasiones una puede resultar más práctica que las demás.

Dado que la EMPRESA DE CIBERSEGURIDAD se encuentra en crecimiento, se pretende continuar trabajando con estas herramientas por tiempo indefinido. Además, tomando como base este proyecto, se implementó Snort como un servidor IPS, ya que los cambios necesarios en las configuraciones ya probadas y documentadas son mínimos. Actualmente se han hecho algunas pruebas para utilizar este tipo de herramientas lo que puede ser un beneficio adicional a futuro.

Finalmente debo agregar que este proyecto me ayudó a tener un mejor panorama de cómo debe desarrollar un proyecto de forma adecuada en el ámbito profesional, el tener la responsabilidad de trabajar con equipos que se encontrarán en producción y que por lo tanto se espera que funcionen sin contratiempos, es muy diferente de una implementación experimental como se da normalmente en las escuelas.

En este sentido, ser autodidacta se vuelve una habilidad realmente necesaria, ya que en el trabajo no hay alguien que te instruya acerca de cómo se deben hacer las cosas, sino que es uno como ingeniero, quien debe dar solución a los problemas que se presentan.

Durante el desarrollo de esta actividad logré mejorar habilidades y ampliar los conocimientos adquiridos durante mi formación en la Facultad, que si bien había practicado anteriormente, nunca lo había hecho de manera profesional. También fue de utilidad para pulir y fortalecer los distintos conocimientos con los que ya contaba.

No se trató de un proyecto sencillo, ya que involucro muchas configuraciones y tiempo invertido en investigación e implementación de la solución, sin embargo, resulta muy gratificante conocer la forma en que funciona el motor de un IDS como lo es Snort; aunado al hecho de comprender la estructura de los componentes y las reglas que utiliza para la detección de incidentes de seguridad.



Glosario

SOC.- Security Operation Center (Centro de Operaciones de Seguridad) es un centro de trabajo destinado a brindar servicios de Ciber seguridad gestionada.

CERT.- Equipo de Respuesta a Emergencias de Cómputo (Computer Emergency Response Team).

Incident Handler.- Persona encargada de tomar las acciones pertinentes para bloquear a un atacante al momento de descubrir una violación de seguridad informática en una infraestructura dada.

IDS.- Intrusion Detection System (Sistema de Detección de Intrusiones) Es un software diseñado para alertar cuando alguien o algo intenta comprometer un sistema de información a través de actividades maliciosas o violaciones de políticas de seguridad.

SNORT.- Sistema para detección de intrusiones de código abierto para monitoreo de tráfico de red en tiempo real.

Ataque.- Intento de obtener acceso no autorizado a sistemas de servicios, recursos, información o de comprometer la integridad de un sistema.

Defacement.- Tipo de ataque en el que una entidad maliciosa modifica el contenido y apariencia de un sitio web.

Código malicioso.- Código de un programa que realiza acciones sin autorización y que puede tener un impacto adverso en la confidencialidad, integridad y disponibilidad de la información presente en un sistema informático.

Incidente.- Ocurrencia que presenta una amenaza a un sistema de información o a la información presente en la misma, y que puede necesitar de una respuesta para mitigar las consecuencias.

Indicador de Compromiso.- Piezas de datos forenses, tales como datos hallados en las bitácoras de un sistema o archivos, que identifican una actividad potencialmente maliciosa en un sistema o una red.

DMZ.- Zona desmilitarizada (Demilitirized Zone) es una sub-red física o lógica que separa una red local de cualquier red externa (usualmente el internet), típicamente en esta red se encuentran los servidores de recursos y/o servicios para que puedan ser accesibles desde internet sin permitir el acceso al resto de la red.

Evento.- Es un cambio en las operaciones diarias de una red o sistema de información que indica que se ha presentado una violación de seguridad.

Payload.- Datos esenciales transmitidos dentro de un paquete u otra medida de medición.

Vector de ataque.- Es el medio por el cual un atacante gana acceso a una computadora o servidor.

Dirección IP.- Dirección única que identifica a un dispositivo en internet o en una red local.

Lista negra.- También llamada “blacklist” es una lista compuesta por direcciones IP de servidores con comportamiento sospechoso o malicioso.

Host.- Equipo o conjunto de ellos, que ofrece un servicio al resto de los equipos conectados a una red, ya sea local o global.

Dominio.- También conocido como nombre de dominio es el nombre que identifica un sitio web y debe ser único en internet.

Agente de usuario.- (User-Agent) Incluye información como el nombre de una aplicación, su versión, el sistema operativo utilizado y el lenguaje de programación.

HTTP.- Hyper Text Transfer Protocol (Protocolo de Transferencia de Hipertexto) funciona como un protocolo de solicitud-respuesta que fue diseñado para permitir comunicación entre clientes y servidores.

Servidor Command and Control.- Un servidor command and control es la computadora central que manda instrucciones a una botnet (Ejército de computadoras zombie) y que recibe reportes de los equipos comprometidos.

Firewall.- Software o Hardware que comprueba la información procedente de internet o de una red local y a continuación bloquea o permite el paso de la misma por el equipo, en función de la configuración del firewall.

Proxy.- Se trata de un equipo que funge de intermediario entre un explorador web e internet, entre sus funciones ayudan a mejorar la seguridad, ya que filtran algunos contenidos web y software malicioso.

Deep web.- Se trata de una colección de sitios web que son públicamente visibles (usualmente mediante un navegador web especial) pero ocultan la dirección IP del servidor donde se encuentran hospedados, además de que no se pueden hallar dichos sitios mediante buscadores web (no están indizados).

Sniffer.- Programa utilizado para capturar tramas de red.

URL.- De **Uniform Resource Locator** (Localizador Uniforme de Recursos) estándar que permite denominar recursos en internet para que puedan ser localizados.

GET/POST.- Dos de los métodos más utilizados para peticiones y respuestas entre un cliente y un servidor.



Referencias

Bibliografía:

Richard Bejtlich. (2013). Practice of Network Security Monitoring - Understanding Incident Detection and Response. Estados Unidos: No Starch Press.

Referencias Electrónicas

Paul Cichonski, Tom Millar, Tim Grance, Karen Scarfone. (2012). Computer Security Incident Handling Guide. Enero de 2016, de National Institute of Standards and Technology Sitio web: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf>

Patrick Kral. (2011). Incident Handler's Handbook. Enero de 2016, de SANS Institute Sitio web: <https://www.sans.org/reading-room/whitepapers/incident/incident-handlers-handbook-33901>

David Kovar. (N/A). Blue Team Perspectives - The Business of Incident Response. Enero de 2016, de SANS Digital Forensics and Incident Response Sitio web: https://digital-forensics.sans.org/summit-archives/Prague_Summit/Blue_Team_Perspectives_David_Kovar.pdf

Joaquín García Alfaro. (N/A). Detección de ataques en red con Snort. Enero de 2016, de Universidad Autónoma de Barcelona Sitio web: <http://www.deic.uab.es/material/26118-snort.pdf>

Noah Dietrich. (2015). Snort 2.9.8.x on Ubuntu 12, 14, and 15. Enero de 2016, de Snort.org Sitio web: https://s3.amazonaws.com/snort-org-site/production/document_files/files/000/000/090/original/Snort_2.9.8.x_on_Ubuntu_12-14-15.pdf

Glosario

Concepto: SOC

Liga: <http://www.innotecsystem.com/content/page/servicios-gestionados-seguridad>

Fecha de Consulta: Febrero de 2016

Concepto: CERT

Liga: <http://www.innotecsystem.com/content/page/servicios-gestionados-seguridad>

Fecha de Consulta: Febrero de 2016

Concepto: Incident Handler

Liga: <https://www.truesec.be/services/incident-handling/>

Fecha de Consulta: Febrero de 2016

Concepto: IDS

Liga: <https://www.techopedia.com/definition/3988/intrusion-detection-system-ids>

Fecha de Consulta: Febrero de 2016

Concepto: SNORT

Liga: <http://searchmidmarketsecurity.techtarget.com/definition/Snort>

Fecha de Consulta: Febrero de 2016

Concepto: Ataque

Liga: https://niccs.us-cert.gov/glossary#letter_a

Fecha de Consulta: Febrero de 2016

Concepto: Defacement

Liga: <http://www.trendmicro.com/vinfo/us/security/definition/website-defacement>

Fecha de Consulta: Febrero de 2016

Concepto: Código malicioso

Liga: https://niccs.us-cert.gov/glossary#letter_m

Fecha de Consulta: Febrero de 2016

Concepto: Incidente

Liga: https://niccs.us-cert.gov/glossary#letter_i

Fecha de Consulta: Febrero de 2016

Concepto: Indicador de Compromiso

Liga: <http://searchsecurity.techtarget.com/definition/Indicators-of-Compromise-IOC>

Fecha de Consulta: Febrero de 2016

Concepto: DMZ

Liga: <http://searchsecurity.techtarget.com/definition/DMZ>

Fecha de Consulta: Febrero de 2016

Concepto: Evento

Liga: <http://whatis.techtarget.com/definition/security-event-security-incident>

Fecha de Consulta: Febrero de 2016

Concepto: Payload

Liga: <http://searchsecurity.techtarget.com/definition/payload>

Fecha de Consulta: Febrero de 2016

Concepto: Vector de ataque

Liga: <http://searchsecurity.techtarget.com/definition/attack-vector>

Fecha de Consulta: Febrero de 2016

Concepto: Dirección IP

Liga: <http://techterms.com/definition/ipaddress>

Fecha de Consulta: Febrero de 2016

Concepto: Lista negra

Liga: <https://apps.mexagon.net/clientes/knowledgebase/98/Listas-negras-Blacklist.html>

Fecha de Consulta: Febrero de 2016

Concepto: Host

Liga: <http://www.mastermagazine.info/termino/5270.php>

Fecha de Consulta: Febrero de 2016

Concepto: Dominio

Liga: <http://www.masadelante.com/faqs/dominio>

Fecha de Consulta: Febrero de 2016

Concepto: Agente de usuario

Liga: <http://www.alegsa.com.ar/Dic/agente%20de%20usuario.php>

Fecha de Consulta: Febrero de 2016

Concepto: HTTP

Liga: http://www.w3schools.com/tags/ref_httpmethods.asp

Fecha de Consulta: Febrero de 2016

Concepto: Servidor command and control

Liga: <http://whatis.techtarget.com/definition/command-and-control-server-CC-server>

Fecha de Consulta: Febrero de 2016

Concepto: Firewall

Liga: <http://windows.microsoft.com/es-mx/windows/what-is-firewall#1TC=windows-7>

Fecha de Consulta: Febrero de 2016

Concepto: Proxy

Liga: <http://windows.microsoft.com/es-xl/windows-vista/what-is-a-proxy-server>

Fecha de Consulta: Febrero de 2016

Concepto: Deep Web

Liga: <http://www.pcadvisor.co.uk/how-to/internet/what-is-dark-web-how-access-dark-web-deep-joc-3593569/>

Fecha de Consulta: Febrero de 2016

Concepto: Sniffer

Liga: <http://www.mundocisco.com/2009/08/que-es-un-sniffer.html>

Fecha de Consulta: Febrero de 2016

Concepto: URL

Liga: <http://definicion.de/url/>

Fecha de Consulta: Febrero de 2016

Concepto: GET/POST

Liga: http://www.w3schools.com/tags/ref_httpmethods.asp

Fecha de Consulta: Febrero de 2016



Anexos

Anexo A - Opciones de configuración para http_inspect Global

Opciones disponibles para la configuración del módulo Global

iis_unicode_map <mapa> unicode.map	Se trata de un archivo que le indica a HTTP Inspect cómo debe decodificar caracteres Unicode.
proxy_alert	Alerta sobre el uso del servidor proxy HTTP. Este preprocesador generará alertas sobre los usuarios que no utilizan los proxies pre-configurados o utilizan algún tipo de proxy independiente.
compress_depth <número>	Especifica la cantidad máxima de paquetes a descomprimir, el rango de valores esta entre 1 y 65535. por defecto es 1460
decompress_depth <número>	Especifica la cantidad máxima de datos descomprimidos a obtener desde un paquete de datos comprimido, el rango de valores esta entre 1 y 65535. por defecto es 2920

Opciones disponibles para la configuración del módulo Server

profile <perfil>	<p>Indica el perfil que utilizará HTTP Inspect para realizar los procesos de auditoria de las peticiones y respuestas HTTP, cada uno de estos perfiles están diseñados para que funcionen de una forma óptima dependiendo del tipo de servidor web, sin embargo, es una propiedad opcional y no es necesaria para el correcto funcionamiento del módulo, los valores validos son:</p> <p>all: Normaliza las URI utilizando todas las técnicas de ataque disponibles y alerta sobre formas de evasión más serias, con este perfil, se intenta detectar todos los tipos de ataques sin importar el tipo de servidor HTTP.</p> <p>Apache: Perfil para Servidores web Apache.</p> <p>iis: Perfil para servidores web ISS.iss4_0, iss5_0: Son idénticos al perfil ISS, la diferencia está en que alerta si una URL tiene doble codificación.</p>
ports {<puerto><puerto>}	<p>Se trata de los puertos HTTP para decodificar peticiones en el servidor HTTP. Sin embargo, el tráfico HTTPS es cifrado y no puede ser decodificado por HTTP Inspect.</p> <p>Para tratar con esto es necesario utilizar el preprocesador SSL.</p>
extended_response_inspection	Con el uso de esta propiedad todos los campos HTTP son inspeccionados, todos estos campos son extraídos y almacenados

	en buffers, posteriormente se aplican distintas reglas sobre dichos buffers.
enable_cookie	Permite la extracción de cookies desde peticiones HTTP y respuestas HTTP, estas cookies extraídas son almacenadas en el buffer de cookies
inspect_gzip	<p>Con esta opción HTTP inspect descomprimirá ficheros comprimidos (gzip/deflate) que viajen en las respuestas.</p> <p>Para que esta propiedad funcione correctamente se debe activar la opción extended_response_inspection, por otro lado también depende de los valores indicados en las propiedades compress_depth y decompress_depth definidas en el módulo global, ya que el proceso de descompresión finalizará cuando se alcance uno de estos dos límites o cuando los datos sean descomprimidos por completo.</p>
unlimited_decompress	Con esta opción no se aplican restricciones para descomprimir datos comprimidos (inclusive aunque vengan partidos en distintos paquetes) la des-compresión finalizará solamente cuando los datos sean descomprimidos por completo o cuando una secuencia de final de paquete sea recibida. Sin embargo la des-compresión en un único paquete aún se encuentra limitada por el valor de la propiedad compress_depth
enable_xff	Con esta propiedad Snort podrá parsear y registrar la dirección IP original de cliente presentada en el X-Forwarded-For o True-Client-IP
server flow depth <número>	<p>Indica el tamaño de respuesta del servidor a inspeccionar, cuando se encuentra activada la propiedad extended_response_inspection la inspección se aplicará al cuerpo de la respuesta HTTP y no a los Headers, cuando la propiedad extended_response_inspection esta desactivada, la inspección aplicará a toda la respuesta incluyendo los headers. Esta opción es utilizada para balancear las necesidades de desempeño del IDS y el nivel de inspección necesario sobre los datos de la respuesta de servidores web.</p> <p>El rango de valores va desde -1 hasta 65535, donde el valor -1 indicará a Snort que ignore todo el tráfico del lado del servidor para los puertos definidos en la propiedad ports cuando la propiedad extended_response_inspection esta desactivada, sin embargo cuando esta activada, Snort ignorará el cuerpo de la respuesta HTTP, pero no sus headers. Inversamente al valor de -1, con el valor de 0, Snort inspeccionará todos los campos HTTP definidos en la propiedad ports (lo que afectará el rendimiento general del</p>

	<p>IDS). Valores superiores a 0 le indicarán a Snort el número de bytes que debe inspeccionar en la respuesta del servidor.</p> <p>El valor por defecto es de 300 bytes, ya que frecuentemente los datos de los headers son menores a esta longitud, sin embargo este valor puede variar dependiendo de las particularidades específicas de cada entorno, por esta razón, en muchas ocasiones se ha recomendado establecer el máximo valor.</p>
<p>client_flow_depth <número></p>	<p>Indica el tamaño de la petición del cliente que Snort inspeccionará, a diferencia de server_flow_depth, esta propiedad es aplicada al primer paquete de la petición HTTP y no está basado en el flujo de una sesión TCP. El rango de valores para esta propiedad está entre -1 y 1460 y el valor por defecto es de 300, con este valor Snort no intentará inspeccionar Cookies demasiado grandes que frecuentemente aparecen al final de los headers de muchas peticiones de clientes. El valor de -1 causa que Snort ignore todo el tráfico del lado del cliente para los puertos especificados en la propiedad ports, Inversamente al significado del valor -1, cuando se indica el valor 0, Snort inspeccionara todo el tráfico HTTP del lado del cliente en los puertos indicados en la propiedad ports, valores superiores a 0 indican el número de bytes que inspeccionará Snort en el primer paquete de la petición del cliente. Del mismo modo que server_flow_depth, dependiendo de las particularidades del entorno en algunas ocasiones es recomendable establecer el valor máximo de 1460.</p>
<p>post_depth <número></p>	<p>Especifica la cantidad de datos a inspeccionar en un post message del cliente, el valor puede ser establecido entre -1 y 65495. El valor por defecto es -1 lo que causa que Snort ignore todos los datos en el post message. Inversamente al valor -1, el valor 0 indicará a Snort que inspeccione todos los datos en el post message del cliente, valores superiores a 0 indican el número de bytes a inspeccionar en el post message del cliente.</p>
<p>ascii <yes no></p>	<p>Esta opción indica cuando decodificar caracteres ASCII codificados como por ejemplo "%2f" "%2e". Es muy frecuente que en las URL existan caracteres ASCII codificados, por lo tanto es recomendado tener esta característica desactivada para evitar que http inspect genere alertas relacionadas con esto.</p>
<p>multi_slash <yes no></p>	<p>Esta opción normaliza múltiples diagonales en una línea, (URL o URI) dejando solamente uno valido, por ejemplo home//////////page es normalizado a home/page, esta opción permite generar una alerta cuando se detecta este tipo de comportamientos cuando esta activada.</p>

<p>directory <yes no></p>	<p>Esta opción normaliza directorios transversales y autorreferenciales, por ejemplo: /home/./page, se normaliza a /home/page, o /home/./page se normaliza en /home/page.</p> <p>Especificando “yes” en esta opción se genera una alerta, sin embargo en algunos casos puede resultar en falsos positivos, puesto que algunos sitios web hacen referencia a directorios transversales.</p>
<p>allow_proxy_use</p>	<p>Especificando esta opción, indica a HTTP Inspect que no genere alertas si la opción global proxy_alert esta activada, si la opción proxy_alert no se encuentra activada, esta propiedad no hace nada. La opción allow_proxy_use es una forma de restringir el uso no autorizado del proxy de un servidor autorizado</p>
<p>oversize_dir_length <número></p>	<p>Establece el número máximo de caracteres permitido para la longitud de un directorio en una URL, si un directorio URL es más amplio que este tamaño se genera una alerta, un valor recomendado es 300.</p>
<p>inspect_uri_only</p>	<p>Se trata de una opción que permite inspeccionar solamente la porción URI de una petición HTTP, dado que normalmente entre el 90% y 95% de los ataques se encuentran en dicha porción de las peticiones, se capturarán la mayoría de los ataques, por otro lado esta opción mejora el desempeño cuando se encuentra habilitada, sin embargo, no tiene en cuenta ninguna regla que utilice el “uricontent”, por ejemplo si se define la siguiente regla:alert tcp any -> any 80 (msg:”content”; content: “foo”;) y posteriormente se realiza la siguiente petición: get /pagina.htm http/1.0\r\n\r\n Lo anterior no generará ninguna alerta.</p>
<p>max_header_length <número></p>	<p>Se trata del valor máximo permitido para la cabecera de una petición, las peticiones que excedan esta restricción generarán una alerta de “Long Header”, por defecto se encuentra desactivada, para activar esta propiedad se debe establecer un valor mayor a 0 y menor o igual 65535, especificando el valor de 0, indica que se desactiva la alarma.</p>
<p>normalize_headers</p>	<p>Activa la normalización de headers HTTP, usando la misma configuración de los parámetros de normalización comunes como son multi slash o transversal directory, es útil para normalizar Referrer URI’s que podrían aparecer en el header HTTP.</p>
<p>normalize_cookies</p>	<p>Activa la normalización de campos de Cookies usando la misma configuración de los parámetros de normalización comunes como son multi slash o transversal directory, es útil para normalizar Cookies codificadas</p>

normalize_utf	Activa la normalización para el cuerpo de las respuestas HTTP, donde el header "Content-Type" tiene un conjunto de caracteres "utf-16le", "utf-16be", "utf-32le", o "utf-32be". HTTP inspect normalizará estos valor a su forma de codificación de 8-bits generando una alarma en el caso de que bytes extras sean mayores a 0.
max_headers <número>	Indica el número máximo de campos que puede contener el Header de una petición, las peticiones que superen este valor generarán una alerta de "Max Headers". La opción se encuentra desactivada por defecto, para activarla, es necesario especificar un valor mayor a cero y menor o igual a 1024. Especificar el valor de 0 es igual que desactivar esta opción.
no_alerts	Desactiva todas las alertas generadas por HTTP Inspect, sin embargo no tiene efecto alguno sobre las reglas definidas en Snort.
http_methods <métodos>	Especifica métodos adicionales en las peticiones HTTP a parte de los chequeados por defecto en este preprocesador (GET y POST) Por ejemplo: http_methods { PUT TRACE CONNECT }

Anexo B - Configuración de interfaces

El equipo habilitado como sensor IDS requiere de dos interfaces de red, una que siempre funcionará en modo promiscuo; es decir, que estará escuchando el tráfico presente en el segmento de red que se desea monitorear. La segunda interfaz, será a la que se le asigne una dirección IP con la que se podrá acceder a la GUI de monitoreo.

A continuación se muestra un ejemplo de configuración, realizada en el archivo **interfaces**, localizado en la ruta **/etc/network**.

```
# interfaces(5) file used by ifup(8) and ifdown(8)

auto eth0
iface eth0 inet manual
up ifconfig $IFACE 0.0.0.0 up
up ip link set $IFACE promisc on
down ip link set $IFACE promisc off
down ifconfig $IFACE down

auto eth1
iface eth1 inet static
address 10.30.1.12
gateway 10.30.1.1
netmask 255.255.255.0
dns-nameservers 10.30.12.235 8.8.8.8

auto lo
iface lo inet loopback
```

Anexo C - Instalación manual de las reglas de Snort

Las reglas de Snort pueden instalarse de forma manual (sin utilizar PulledPork) y es necesario un Oinkcode (Gratuito al registrarse en el sitio oficial de Snort).

Es necesario quitar los comentarios en todas las líneas “*#include*” dentro del archivo **snort.conf**, debido a que las reglas descargadas serán una serie de archivos en vez del archivo único creado por PulledPork.

```
usuario@equipo:~$ sudo sed -i 's/\#include \$RULE_PATH/include \$RULE_PATH/'  
/etc/snort/snort.conf
```

Descargar las reglas utilizando el código oink obtenido del sitio, se recomienda procurar utilizar la versión más reciente de las reglas.

```
usuario@equipo:~$ cd /snort_src  
usuario@equipo:~$ sudo wget https://www.snort.org/reg-rules/snortrules-snapshot-  
2956.tar.gz/<oinkcode> -O snortrules-snapshot-2956.tar.gz  
usuario@equipo:~$ sudo tar xvfz snortrules-snapshot-2956.tar.gz -C /etc/snort
```

Copiar los nuevos archivos de **/etc/snort/etc** a **/etc/snort** (y eliminar la carpeta **/etc/snort/etc**)

```
usuario@equipo:~$ sudo cp /*.conf* ../  
usuario@equipo:~$ sudo cp /*.map ../  
usuario@equipo:~$ cd /etc/snort  
usuario@equipo:~$ sudo rm -Rf /etc/snort/etc
```

Respetar la configuración establecida anteriormente

```
ipvar HOME_NET 10.30.1.0/24  
ipvar EXTERNAL_NET !$HOME_NET  
var RULE_PATH /etc/snort/rules  
var SO_RULE_PATH /etc/snort/so_rules  
var PREPROC_RULE_PATH /etc/snort/preproc_rules  
var WHITE_LIST_PATH /etc/snort/rules  
var BLACK_LIST_PATH /etc/snort/rules
```

Realizar una prueba del archivo de configuración con el siguiente comando:

```
usuario@equipo:~$ sudo snort -T -c /etc/snort/snort.conf
```

Anexo D - Reconstruyendo la base de datos de Snort

Ocasionalmente **Barnyard2** puede tener problemas y no iniciar de manera correcta, para resolver borrar la base de datos de Snort puede ayudar.

Detener los servicios de **Snort** y **Barnyard2** (en caso de que se estén ejecutando):

```
usuario@equipo:~$ sudo service snort stop
usuario@equipo:~$ sudo service barnyard2 stop
```

Borrar la base de datos de Snort:

```
usuario@equipo:~$ mysql -u root -p
> drop database snort;
> exit
```

Y recrearla como se hizo originalmente durante la instalación de Barnyard2 (Utilizar la contraseña de root para MySQL cuando sea requerida).

```
usuario@equipo:~$ sudo echo "create database snort;" | mysql -u root -p
usuario@equipo:~$ sudo echo "grant create, insert, select, delete, update on
snort.* to snort@localhost identified by 'MYSQLSNORTPASSWORD'" | mysql -u root -p
usuario@equipo:~$ mysql -u root -p -D snort < /snort_src/barnyard2-2-
1.13/schemas/create_mysql
```

Finalmente habrá que re-configurar BASE desde la interfaz web

1. Ir a http://ip_de_snort/base/index.php y hacer clic en el link “**setup page**”.
2. Clic en el botón “**Create BASE AG**” en la esquina superior de la página.
3. Clic en la línea “**Main page**”.

Reiniciar el equipo y verificar los servicios se estén ejecutando.

Adicionalmente se pueden eliminar también los archivos en **/var/log/snort** (será necesario crear un nuevo archivo **barnyard2.waldo** para que barnyard2 se ejecute de forma correcta).

```
usuario@equipo:~$ sudo rm /var/log/snort/*
usuario@equipo:~$ sudo touch /var/log/snort/barnyard2.waldo
```


Anexo E - Scripts de inicio automático para Snort y Barnyard2

Primero se debe crear el script de inicio automático para Snort, por lo que se crea un archivo con el nombre **“snort.conf”** en la siguiente ubicación:

```
usuario@equipo:~$ sudo nano /etc/init/snort.conf
```

Dentro del archivo recién creado se deben añadir las siguientes líneas:

```
description "Snort NIDS Service"  
stop on runlevel [!2345]  
start on runlevel [2345]  
script  
exec /usr/local/bin/snort -q -u snort -g snort -c /etc/snort/snort.conf -i eth0 -D  
end script
```

Se define una breve descripción del contenido del script, también se definen los niveles de ejecución (runlevel 2, 3, 4 o 5) en los que se iniciará o se detendrá el servicio; y finalmente se tiene la instrucción a ejecutar entre las líneas **script** y **end script**.

Se cambian los permisos del script para que pueda ser ejecutado y se verifica que funciona correctamente.

Se repite el procedimiento para el script de inicio de Barnyard2:

```
usuario@equipo:~$ sudo nano /etc/init/barnyard2.conf
```

Contenido en el script:

```
description "Barnyard2 service"  
stop on runlevel [!2345]  
start on runlevel [2345]  
script  
exec /usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort -f  
snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort -D  
end script
```

Volviendo ejecutable el script de Barnyard2 y verificando que funcione:

Por último se reinicia el equipo y se verifica que ambos servicios iniciaron correctamente:

```
usuario@equipo:~$ sudo service snort status  
usuario@equipo:~$ sudo service barnyard2 status
```

