PONTIFICIA **UNIVERSIDAD CATÓLICA** DEL PERÚ

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN PARA LA ADMINISTRACIÓN DE LAS HERRAMIENTAS DE SEGURIDAD EN UNA RED LOCAL

Tesis para optar por el Título de Ingeniero Informático, que presenta el bachiller:

Dennis Stephen Cohn Muroy

ASESOR: Ingeniero Corrado Daly Scaletti

Lima, octubre del 2008

Resumen

Desde el año en que se estableció la primera red de computadoras (ARPANET), hasta nuestros días, Internet ha pasado a través de un largo proceso evolutivo. Siendo utilizado actualmente como fuente de conocimiento, medio de comunicación y una amplia plataforma para hacer negocios (e-business). Lastimosamente, también es un canal a través del cual se perpetran ataques que han ocasionado pérdidas de información no sólo a las empresas de diversos tamaños, sino también a las personas naturales.

Como mecanismo de prevención, es necesario hacer uso de una serie de herramientas de tipo software y/o hardware, así como políticas de seguridad a fin de proteger la confidencialidad, integridad y disponibilidad de la información.

Sin embargo, las soluciones presentes en el mercado, a pesar de poseer un adecuado desempeño en cuanto a la prevención y la detección de los ataques, carecen de un entorno intuitivo y de fácil uso, lo cual influye en el registro de reglas débiles o erróneas; provocando agujeros en el perímetro de la seguridad de la red local.

Es por ello, que en el presente documento se plantea como solución analizar, diseñar e implementar una aplicación para facilitar la administración de las herramientas Iptables, Squid y Snort; utilizadas para proteger la información dentro de una red local.

Dedicado a:

Todas aquellas personas quienes buscan usar la ciencia en pos de ayudar a los demás.

Agradecimientos:

A mi madre, a mi asesor, a mis profesores y a mis amigos quienes me apoyaron a lo largo de estos cinco años.

A mis compañeros del grupo Linux-IDES por presentarme las bondades del FOSS.

ÍNDICE

Índice de Imágenes

Índice de Tablas

Introducción

Desde el año 1969, época en que se estableció la primera red de computadoras a la cual se le denominó ARPANET, hasta nuestros días, Internet ha pasado a través de un largo proceso evolutivo. Siendo en la actualidad una de las principales fuentes de conocimiento, comunicación y una amplia plataforma para hacer negocios (e-business). Lastimosamente, también es un canal a través del cual se perpetran ataques que han ocasionado pérdidas de información no sólo a las empresas de diversos tamaños, sino también a las personas naturales.

Como mecanismo de prevención, es necesario hacer uso de una serie de herramientas de tipo software y/o hardware, así como políticas de seguridad a fin de asegurar la confidencialidad, integridad y disponibilidad de la información. Sin embargo, las soluciones presentes en el mercado, a pesar de poseer un adecuado desempeño en cuanto a la prevención y la detección de los ataques, carecen de un entorno intuitivo y de fácil uso, confundiendo a los administradores al efectuar las configuraciones necesarias; esto trae como consecuencia el registro de reglas débiles o erróneas, provocando agujeros en el perímetro de la seguridad de la red local.

Es por ello, que en el presente documento se plantea como solución analizar, diseñar e implementar una aplicación para facilitar la administración de las herramientas Iptables, Squid y Snort; utilizadas para proteger la información dentro de una red local.

Capítulo 1: Generalidades

En el presente capítulo se explicarán los conceptos necesarios para poder comprender el problema que se desea resolver a través del presente proyecto, se mostrará el esquema seguido para realizar el proyecto y se dará a conocer las alternativas de solución existentes en la actualidad.

1.1. Definición del Problema

Desde sus orígenes en el año 1969, en donde se estableció la primera red de computadoras a la cual se le denominó ARPANET, hasta nuestros días, Internet ha pasado a través de un largo proceso evolutivo. Actualmente, es una de las principales fuentes de conocimiento y de intercambio de información; además se ha convertido en una plataforma que brinda grandes oportunidades para hacer negocios (e-business).

Lastimosamente, también es usado como un medio para perpetrar ataques que han ocasionado pérdidas de información no sólo a las

empresas de diversos tamaños, sino también a las personas naturales. Dichos ataques han ido creciendo en los últimos años, según cifras brindadas por [WWW0001], “Un 70% de las organizaciones anunciaron al menos un incidente de seguridad durante 2000, frente a un 42% anunciado en 1996”.

Dentro de las ideas que a lo largo del tiempo han surgido para contrarrestar esta situación, se llegó a presentar el supuesto de que el computador más seguro era aquel que se hallaba desconectado de la red y guardado en una habitación cerrada con un guardián en la puerta (Eugene H. Spafford); sin embargo, esta alternativa ya no puede considerarse como válida al hallarnos en una época en la que es mucho más evidente que para poder prosperar tanto en el ámbito tanto social como económico como educativo, las personas no pueden aislarse unas de otras.

En consecuencia, es necesario hacer uso de una serie de herramientas de tipo software y/o hardware, así como políticas de seguridad que aseguren se respeten los siguientes principios de la seguridad de la información: confidencialidad, integridad y disponibilidad (Apuntes del curso de Seguridad Control y Auditoría de Sistemas de Información, 2007).

En la actualidad, para proteger tanto a los usuarios como a la información que viaja por la red, existen en el mercado herramientas que, a pesar de presentar un adecuado desempeño en cuanto a la prevención y la detección de los ataques, carecen de un entorno intuitivo y de fácil uso, confundiendo a los administradores al efectuar las configuraciones necesarias; esto trae como consecuencia el registro de reglas débiles o erróneas provocando agujeros en el perímetro de la seguridad de la red local. Según se menciona en [CIS2007] y en la lista de las veinte (20) mayores vulnerabilidades del 2007 publicada por [WWW0002], el factor humano es una de las principales brechas de seguridad dentro de cualquier sistema.

Es por ello, que en el presente documento se plantea como solución analizar, diseñar e implementar una aplicación que integre la

administración de las herramientas libres ***Iptables*** (cortafuegos – NAT), ***Squid*** (proxy web) y ***Snort*** (Sistema de Detección de Intrusos) – aplicaciones muy usadas actualmente para administrar la seguridad en una red local de computadoras – permitiendo su fácil configuración para apoyar el aseguramiento de la red local.

1.2. Marco Conceptual del Problema

A continuación se expondrán algunos conceptos que ayudarán a comprender la función y utilidad de las herramientas de seguridad sobre las que se construirá la aplicación.

a) Cortafuegos (*firewall*)

Un cortafuegos es un dispositivo, seguro y confiable, que pertenece a una red; ubicado comúnmente en el punto en donde dos o más redes se comunican. Todos los paquetes trasmitidos entre estas redes son analizados por el cortafuegos el cual determina, basándose en una serie de reglas definidas por el administrador de la red, el tráfico que será bloqueado y el tráfico que podrá ser transmitido. En la Imagen 1.1 se puede apreciar el perímetro defensivo establecido por el cortafuegos entre dos (2) redes, una interna y la otra externa.

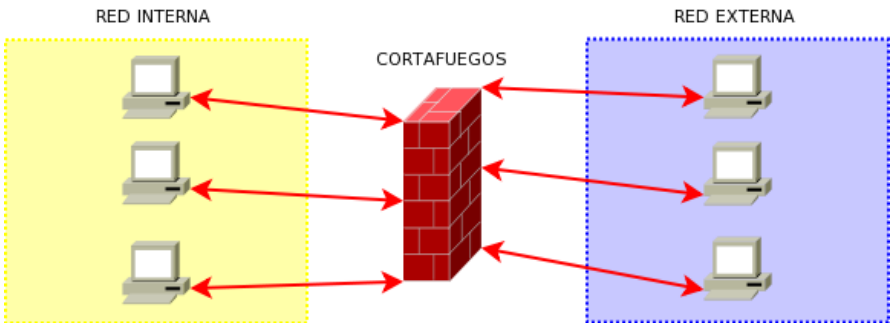


Imagen 1.1: Perímetro defensivo establecido por el cortafuegos

Funcionamiento de un cortafuegos:

En la Imagen 1.2 se puede observar el funcionamiento de un cortafuegos que hace uso de una serie de reglas establecidas para determinar si un paquete de datos debe ser transmitido o rechazado.

- El cortafuegos recibe un paquete o datagrama (1).
- El cortafuegos verifica el origen y el destino y comienza a compararlos con cada una de las reglas establecidas (2).
- En caso el cortafuegos cumpla con la regla (3) procede a verificar los filtros establecidos para dicha regla (4a); caso contrario, verifica si existen más reglas contra las cuales comparar el paquete (4b).
- Si el paquete no cumple con ninguna de las reglas definidas, se ejecuta la regla predeterminada.
- El cortafuegos transmite (5a) o rechaza el paquete al destino (5b) según se haya especificado en las reglas.
- El paquete es recibido por el destinatario(5).

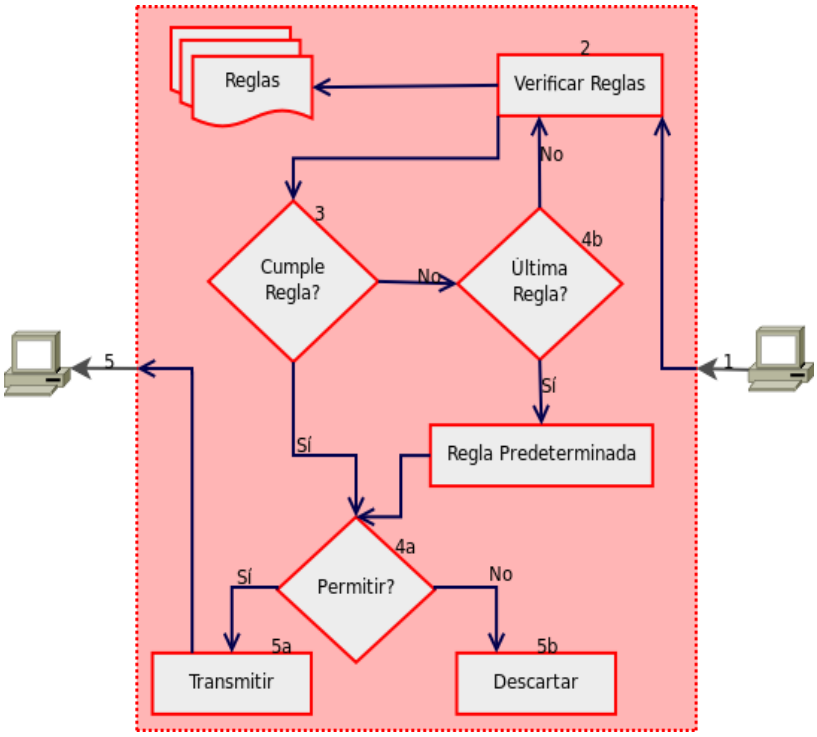


Imagen 1.2: Funcionamiento de un cortafuegos

Tipos de cortafuegos:

Los cortafuegos pueden ser clasificados en dos (2) tipos: Filtros de Paquete y Cortafuegos Proxy.

Filtros de Paquete:

También conocidos como cortafuegos a nivel de red. Estos

cortafuegos analizan rápidamente los paquetes que se están transmitiendo y recibiendo. Debido a que únicamente examinan el tipo de paquete, las direcciones IP de origen y de destino, puertos (TCP, UDP) , el tipo de mensaje y las interfaces de entrada y salida.

Este tipo de cortafuegos trabaja en base a tablas en donde almacena las reglas de filtrado. A continuación se presenta un ejemplo de tabla de filtrado:

Regla	Acción	IP Origen	IP Destino	Puerto Origen	Puerto Destino	Protocolo
1	Permitir	*	192.168.0.1	*	20, 21	TCP
2	Descartar	*	*	*	8080	UDP

Tabla 1.1: Ejemplo de tabla utilizada por un cortafuego de tipo filtro de paquetes

Un ejemplo de este tipo de cortafuegos es el Iptables; herramienta de software libre desarrollada por Netfilter, [WWW0003].

Durante las primeras etapas del procesamiento de paquetes, esta herramienta decide en base a reglas si un determinado paquete debe de ser retransmitido a otra computadora, debe de ser procesado por la misma máquina o debe de ser rechazado. Un diagrama sobre el procesamiento de paquetes a través de Iptables puede ser apreciado en la Imagen 1.3.

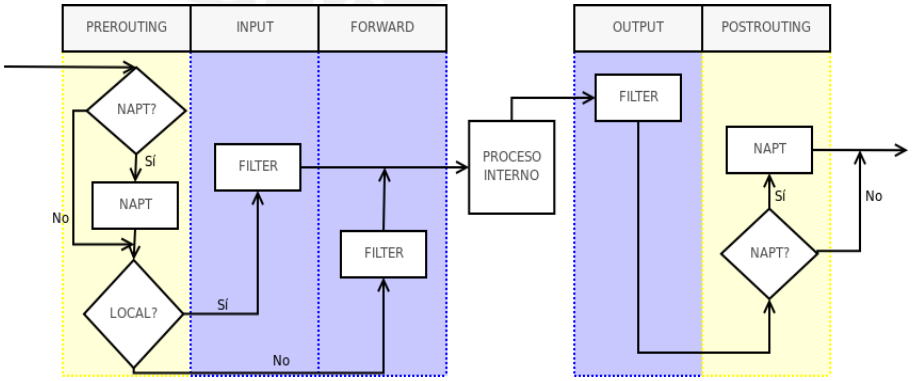


Imagen 1.3: Procesamiento de paquetes a través de Iptables

Las reglas son almacenadas en cadenas las cuales a su vez se encuentran agrupadas en tres (3) tablas cada una de las cuales tiene un objetivo determinado.

Las cadenas, de acuerdo a la tabla a la cual pertenecen son:

- **Tabla de Filtrado:** Esta tabla contiene las cadenas INPUT, OUTPUT y FORWARD, encargadas de las funciones de filtrado de paquetes.
- **Tabla de NAPT:** Esta es la tabla que brinda las reglas necesarias para la reescritura de direcciones o de puertos de los paquetes que ingresen a la NAT o salgan de ella. Cuenta con las siguientes cadenas PREROUTING, POSTROUTING y OUTPUT.
- **Tabla de reescritura:** Conocida como mangle, permite sobreescribir valores de la cabecera de los paquetes IP para afectar la performance de los servicios. Hace uso de todas las cadenas: PREROUTING, INPUT, FORWARD, OUTPUT y POSTROUTING.

Cortafuegos Proxy:

Un cortafuegos de tipo Proxy es aquel que cumple una doble función – cliente y servidor. Actúa como servidor al recibir las peticiones de las máquinas pertenecientes a su misma red; y como cliente al redirigir dichas peticiones a los servidores finales.

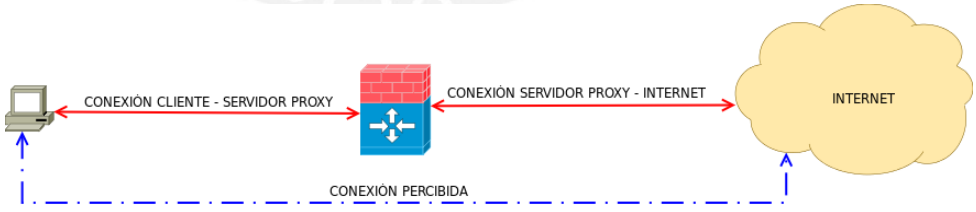


Imagen 1.4: Conexión real vs. conexión percibida al usar un proxy

Su finalidad es el de mantener la información de la red como privada, limitando el acceso a recursos que administrador del sistema considere como peligrosos. Provee mayor nivel de filtrado que el Filtro

de Paquetes, ya que las restricciones que a establecerse pueden ser por ejemplo de acuerdo al servicio, nombres o fechas de los archivos.

Sin embargo, la principal desventaja que puede presentar frente al Filtrado de Paquetes, es la posibilidad de crear cuellos de botella en redes de alta velocidad, para evitar ello, se requiere que su procesamiento sea mayor o igual al del ruteador.

Como ejemplo tenemos la aplicación Squid que es una herramienta de software libre que cumple las tareas de un Proxy-caché, según [WES2004]. Dentro de las funciones que esta solución nos ofrece tenemos:

- Uso de menor ancho de canal al navegar por Internet.
- Reduce el tiempo de carga de las páginas web.
- Proteger los equipos dentro de la red local funcionando como un cortafuegos proxy.
- Previene el acceso a una lista determinada de páginas.
- Recoge estadísticas del tráfico web dentro de la red.

El trabajo del Squid es ser un proxy y una caché. Como proxy es un intermediario en las transacciones web; como caché Squid almacena el contenido web de páginas recientemente visitadas para su posible uso posterior.

b) Traductores de Direcciones de Red (NAT – *Network Address Translation*)

El NAT nació como una solución al problema de la escasez de direcciones IPv4, permitiendo prolongar su uso y de esa forma retrasar la migración al protocolo IPv6.

Tomando como referencia lo mencionado en [HUN2002], los NAT son una extensión a la función de ruteo que permite modificar la dirección de origen de los paquetes que se están enviando. Consiguiendo con ello, que una dirección privada cuente con acceso a Internet.

Esto es posible, ya que antes que los paquetes creados dentro de la red privada sean enviados fuera de la misma, la dirección IP de origen debe de ser convertida a una dirección de Internet válida.

Dentro de los principales beneficios al utilizar redes privadas podemos hallar:

- Independencia con el Proveedor de Servicios de Internet (ISP). Si se cambia de ISP no es necesario reenumerar los hosts dentro de la red. En el peor caso solo sería necesario actualizar los valores registrados en la NAT.
- Permite detectar los ataques de IP spoofing. Al no poder utilizar direcciones privadas para identificar sitios públicos, es posible detectar un ataque de IP spoofing si una máquina de fuera de la red intenta comunicarse haciendo uso de una dirección privada.

Sin embargo, el uso de la NAT puede traer algunas consecuencias:

- Se reduce el desempeño del ruteador al tener éste que dedicarse a desempeñar una tarea adicional.
- Interfiere con los mecanismos de identificación punto a punto que permiten identificar las direcciones de origen.

c) Traductores de Direcciones y Puertos de Red (NAPT – *Network Address and Port Translation*)

Según información recabada de [NEG2004], el NAPT, también llamado PAT, es una herramienta que al igual que un NAT, permite llevar a cabo el enmascaramiento de direcciones IP; sin embargo, su funcionalidad involucra incluso el poder enmascarar puertos.

La principal ventaja que presenta el NAPT frente al NAT, es que mientras el segundo solo permite enmascarar una dirección privada con una dirección pública (relación de uno a uno), el primero permite enmascarar la combinación de IP privada y puerto con una combinación IP pública y puerto (esto permite relacionar puertos de más de una máquina dentro de la red privada con los puertos del

NAPT).

El Iptables, aplicación mencionada anteriormente, también desempeña las funciones de un NAPT.

d) Sistema de Detección de Intrusos (IDS – Intrusion Detection System)

Utilizando como base [COL2005], un IDS es una herramienta que realiza un monitoreo de los eventos que ocurren en un computador o en una red, buscando intrusiones no autorizadas que puedan comprometer la confidencialidad, integridad o disponibilidad de los datos. Las intrusiones detectadas son por lo general causadas por atacantes que ingresan al sistema desde la Internet o usuarios del sistema que desean adquirir privilegios adicionales a los que se les ha asignado.

Estas herramientas se pueden clasificar de acuerdo a la siguiente lista:

- **Sistemas de Detección de Intrusos basados en Red (NIDS):** Estos IDS's son posicionados en sectores claves de la red, siendo su función primordial, analizar el tráfico en tiempo real.

- **Sistemas de Detección de Intrusos basados en Host (HIDS):** Este tipo de IDS se caracteriza por residir dentro de una computadora (host). Se encargan de monitorear las actividades que ocurren en el Sistema Operativo, reportando cada actividad sospechosa al usuario y registrándolo en una bitácora.

A la vez, los IDS también pueden pertenecer a una de las siguientes categorías:

- **Basados en Firmas:** También conocidos como basados en

conocimiento. Su método de detección se basa en modelos que describen la forma en que un ataque es llevado a cabo. Estos modelos reciben el nombre de Firmas (*Signatures*).

- **Basados en Anomalías Estadísticas:** También conocidos como basados en comportamiento. Estos IDS's hacen uso de un mecanismo de aprendizaje a través del cual almacenan en un registro el conjunto de actividades que son comúnmente realizadas en el equipo, definiéndolas como un patrón normal de comportamiento. Una vez el IDS abandone la fase de aprendizaje, cualquier actividad que se halle fuera del patrón definido será considerada como anómala.

- **Basados en Anomalías en el Protocolo:** Estos IDS's tienen conocimiento de cada protocolo que deberán de monitorear. El IDS almacena un modelo por cada protocolo en el que se indica cual es el uso adecuado que se le debe dar. El IDS buscará cualquier anomalía presente en un protocolo que no se ajuste al modelo establecido.

- **Basados en Anomalías en el Tráfico:** El objetivo de este tipo de IDS es el de hallar cambios en los patrones normales del tráfico de una red, para ello lleva a cabo comparaciones sobre un modelo que representa el tráfico ordinario de una red.

- **Basados en Reglas:** Este tipo de IDS's hacen uso de un sistema experto, que hace uso de reglas y datos a ser analizados. Si el conjunto de datos analizar cumplen un patrón definido por las reglas, el IDS considerará que hay una actividad anómala en proceso.

La herramienta Snort es un ejemplo de un Sistema de Detección y Prevención de Intrusos. Permite llevar a cabo análisis de datos que viajan a través de una red en tiempo real, según [WWW0004]. Dentro de las funciones que nos brinda, tenemos:

- Análisis de protocolo.

- Búsqueda de contenido
- detecta una gran variedad de ataques.

Hace uso de un lenguaje flexible de reglas que describe el tráfico que debe de bloquear o permitir su paso, así como de un motor de detección que usa una arquitectura modular basada en componentes (plugins).

Asimismo, cuenta con tres métodos de configuración:

- Modo Sniffer, para lectura de paquetes que circulan por la red, mostrándolos.
- Modo Loggin, registra los paquetes en el disco.
- Modo Sistema de Detección de Intrusos basado en reglas.

1.3. Plan de Proyecto

Siendo un proyecto un esfuerzo temporal – con un fecha de inicio y de final – que se lleva a cabo de forma gradual, siguiendo una serie de pasos, para obtener un producto, servicio o resultado único; es necesario elegir una adecuada metodología de gestión que permita dirigir el proyecto desde su inicio hasta su final de forma exitosa.

Para el presente proyecto de tesis, se ha decidido hacer uso de la metodología de gestión Scrum, metodología de gestión ágil que toma como base varios principios establecidos por el PMI.

a) La metodología Scrum:

Scrum es una metodología simple desarrollada para la gestión de proyectos de diversa complejidad. Siendo concebido pensando más en los miembros del equipo que en la tecnología disponible.

Dentro de las ventajas presentes por el uso de esta metodología tenemos:

- Adecuado manejo de los requerimientos cambiantes.
- Incentiva la motivación del equipo de desarrollo.

- El cliente se haya involucrado con el proyecto en un mayor grado.
- Hace uso del empowerment dentro del desarrollo del proyecto.
- Permite superar satisfactoriamente los fallos presentados durante el tiempo de vida del proyecto.

Roles

Las personas involucradas en un proyecto que hace uso de Scrum asumen uno e los siguientes roles:

Equipo Scrum

Definido dentro del PMBOK como Miembros del Equipo de Proyecto; se halla conformado por las personas dedicadas al desarrollo y diseño de la aplicación. Normalmente está constituido por un grupo de 5 a 9 individuos.

Dueño del Producto

Cumple los roles de Cliente y de Usuario definidos dentro del PMBOK. La persona que asume este rol puede ser tanto un cliente como un miembro del equipo especializado en la lógica del negocio y procesos que se desarrollan; se encarga de proporcionar al Equipo Scrum con conocimientos relacionados al negocio. Asimismo, administra los Backlogs de los productos – es un listado de requerimientos pendientes en donde todas las especificaciones del producto están enumeradas – los cuales están a la vista de todos los miembros de la organización.

Scrum Master

Definido dentro del PMBOK como Director del Proyecto; esta persona tiene diariamente breves reuniones, denominados Scrums Diarios, con el equipo. Asimismo es el encargado de canalizar la información referente a nuevos requerimientos o modificaciones con la finalidad de disminuir el número de interrupciones a los desarrolladores.

Al final de cada ciclo – Sprint – y previamente a comenzar el siguiente, el Scrum Master lleva a cabo una reunión – Scrum Retrospective –

con los integrantes del equipo en donde se evalúan las experiencias vividas a lo largo del Sprint y las conclusiones a las que se ha llegado.

Proceso

A continuación se presentarán las actividades llevadas a cabo en una dirección de la gestión de un proyecto que hace uso de Scrum.

Creación del Backlog

Esta etapa involucra los Procesos de Iniciación y Planificación definidos en el PMBOK. El Dueño del Producto junta todas las especificaciones y requerimientos, incluyendo mejoras y correcciones de errores; asimismo, procede a definir cuales serán las metas a alcanzar. Luego de recabar los requerimientos, el Backlog es dividido en grupos, cada uno de los cuales corresponde a un entregable completamente funcional que aporta valor al negocio.

Etapa del Sprint

Esta etapa involucra los Procesos de Planificación y Ejecución definidos en el PMBOK. Un Sprint puede tener una duración entre dos (2) y cuatro (4) semanas.

El primer día de cada Sprint es utilizado para la creación del Sprint Backlog. Una vez el equipo es consciente de los requerimientos y se definen las fechas, el Dueño del Producto los autoriza a comenzar con la elaboración del producto.

Scrum Diario

Esta etapa involucra los Procesos de Planificación, Seguimiento y Monitoreo definidos en el PMBOK. Todos los días a la misma hora, el Scrum Master sostiene una reunión breve con el equipo con el objetivo de mitigar cualquier obstáculo que les impida el poder laborar de forma fluida.

Se analizan los siguientes puntos:

- El avance logrado desde la última reunión.
- El avance esperado hasta la siguiente reunión.

- Los obstáculos que pueden impedir se cumpla con el avance esperado.

Demostración y Evaluación

Esta etapa involucra los Procesos de Finalización definidos en el PMBOK. Cada Sprint culmina con una demostración de la funcionalidad desarrollada frente a un grupo de personas adicionales al Dueño del Producto (por ejemplo usuarios o ejecutivos de la organización contratante).

El resultado de la evaluación de las funcionalidades formarán la base para el siguiente Sprint.

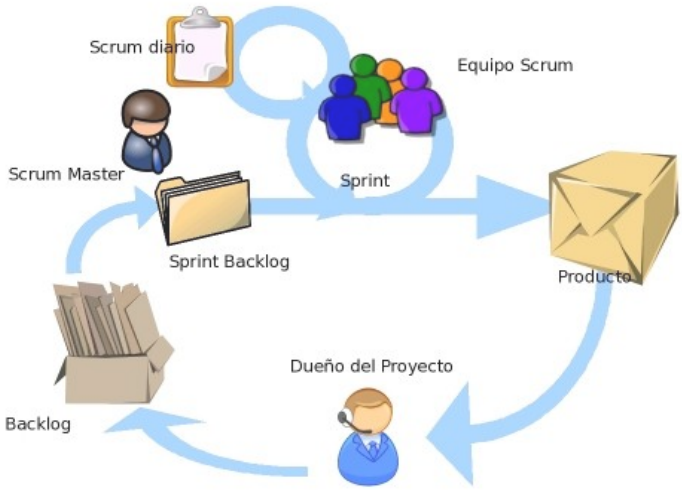


Imagen 1.5: Etapas del Scrum

b) Adaptando el Scrum al proyecto

Para poder hacer uso de la metodología Scrum en el proceso de gestión del presente proyecto será necesario efectuar algunos cambios con la finalidad de adaptarla a la realidad del proyecto (por ejemplo: limitación en el número de miembros que conforman el proyecto).

- El Equipo Scrum estará conformado por un (1) único miembro, quien deberá de efectuar las labores de Scrum Master.

- Las labores de Dueño de Producto serán llevadas a cabo por el Asesor.
- Cada Sprint tendrá una duración de veintiún (21) días y se llevarán a cabo cinco (5) Sprints a lo largo de toda la Tesis.
- Durante el Sprint se mantendrán cuatro reuniones con el Dueño de Proyecto. Una al inicio las demás al final de cada semana – cada siete (7) días.
- La evaluación de funcionalidades será efectuada por el Dueño del Producto al final de cada Sprint.

c) Calendario de entregas

Tomando como base la metodología de gestión definida en el punto anterior, se ha elaborado el siguiente calendario de entregas, vigente a lo largo del desarrollo del presente proyecto.

Tarea	Fecha Inicio	Fecha Fin
Elaboración del Capítulo 1		
Definición del problema	21/01/2008	27/01/2008
Marco conceptual	27/01/2008	03/02/2008
Plan de proyecto	03/02/2008	07/02/2008
Estado del arte	07/02/2008	10/02/2008
Sustentación de la solución	11/02/2008	11/02/2008
Revisión Capítulo 1	11/02/2008	17/02/2008
Correcciones del Capítulo 1	17/02/2008	23/02/2008
Elaboración del Capítulo 2		
Definición de metodología	24/02/2008	09/03/2008
Identificación de requerimientos	10/03/2008	13/03/2008
Análisis de la solución	13/03/2008	15/03/2008
Revisión Capítulo 2	15/03/2008	24/03/2008
Correcciones Capítulo 2	24/03/2008	27/03/2008
Elaboración del Capítulo 3		
Elaboración de la arquitectura de la solución	28/03/2008	29/03/2008
Elaboración de las pantallas	29/03/2008	30/03/2008
Revisión Capítulo 3	31/03/2008	07/04/2008
Correcciones Capítulo 3	07/04/2008	09/04/2008
Elaboración del Capítulo 4		
Definición de la construcción	10/04/2008	11/04/2008
Definición de las pruebas	11/04/2008	13/04/2008
Revisión Capítulo 4	14/04/2008	21/04/2008

Tarea	Fecha Inicio	Fecha Fin
Correcciones Capítulo 4	21/04/2008	27/04/2008
Sprint 1		
Administrar usuarios	04/05/2008	17/05/2008
Administrar configuraciones	17/05/2008	24/05/2008
Visualizar configuraciones	17/05/2008	24/05/2008
Generar reportes de sesiones	24/05/2008	25/05/2008
Sprint 2		
Administrar módulos	25/05/2008	01/05/2008
Administrar servicios	01/05/2008	08/05/2008
Administrar zonas	07/05/2008	15/06/2008
Sprint 3		
Administrar configuración Iptables	29/06/2008	06/07/2008
Administrar reglas Iptables	06/07/2008	12/07/2008
Administrar registro histórico de Iptables	12/07/2008	18/07/2008
Generar reporte Iptables	18/07/2008	20/07/2008
Sprint 4		
Administrar configuración Squid	20/07/2008	25/07/2008
Administrar reglas Squid	25/07/2008	01/08/2008
Administrar registro histórico de Squid	01/08/2008	08/08/2008
Generar reporte Squid	08/08/2008	10/08/2008
Sprint 5		
Administrar configuración Snort	25/08/2008	02/09/2008
Administrar reglas Snort	02/09/2008	07/09/2008
Generar reporte Snort	07/09/2008	14/09/2008
Elaboración del Capítulo 5		
Elaboración de la conclusiones y sugerencias	15/09/2008	20/09/2008

Tabla 1.2: Calendario del Proyecto

1.4. Estado del Arte

Dentro de las aplicaciones actuales que permiten mostrar una interfaz gráfica para la configuración de las herramientas Iptables, Squid y/o Snort tenemos:

a) Webmin

Webmin es una interfaz basada en web que permite administrar sistemas Unix. A través de esta herramienta es posible configurar cuentas de usuario, servicios (como por ejemplo Apache, DNS, entre

otros), bitácoras, proxy (como Squid), etc. Estas configuraciones pueden ser ejecutadas fácilmente tanto de forma local como remota.

Webmin está conformada por una interfaz web y varios programas CGI escritos en lenguaje Perl 5 que son los que interactúan directamente con los archivos de configuración.

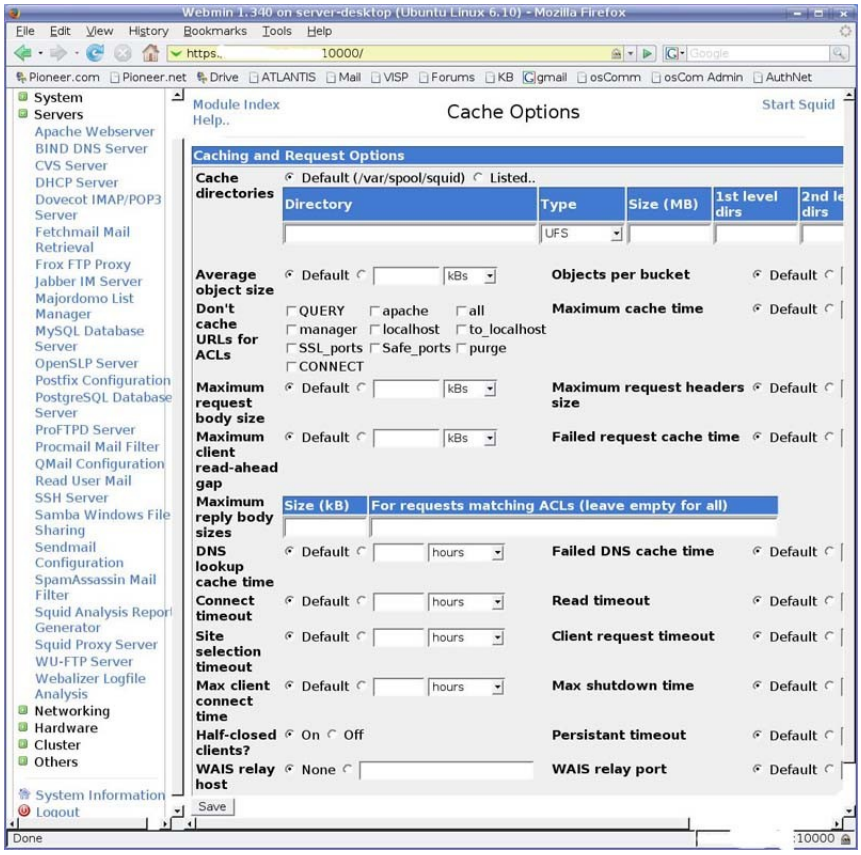


Imagen 1.6: Ejemplo de la pantalla del Webmin para la configuración del Squid

b) Firestarter

Firestarter es una aplicación de código abierto que permite administrar la configuración del cortafuegos. El objetivo de este software es combinar la facilidad de uso con herramientas potentes, para los usuarios de equipos Linux como administradores de sistema.

Características

- Aplicativo de código abierto y gratuito.

- Fácil de utilizar, cuenta con una interfaz gráfica.
- Un ayudante brinda soporte para efectuar la primera configuración.
- Especial para el uso en computadoras de escritorio, portátiles y servidores.
- Monitor en tiempo real de los eventos de intrusión que hayan ocurrido.
- Permite establecer políticas tanto de entrada, como de salida.
- Permite abrir y cerrar puertos con tan solo unos pocos clicks.
- Permite habilitar fácilmente el reenvío de paquetes a través de puertos.
- Permite especificar el tráfico a ser bloqueado.
- Permite visualizar conexiones de redes activas.

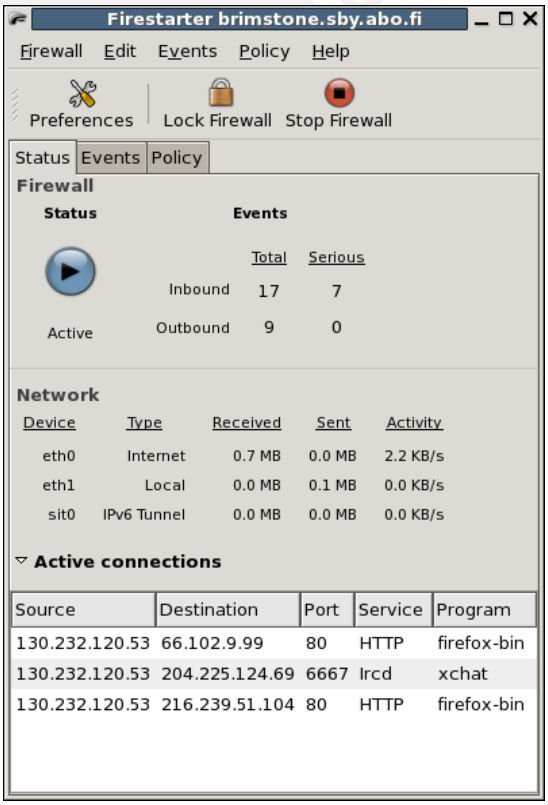


Imagen 1.7: Ejemplo de una pantalla de configuración del Firestarter

c) Guarddog

Guarddog es una herramienta para sistemas Linux que facilita la

configuración de los cortafuegos. Esta solución está dirigida a dos tipos/grupos de usuarios: los usuarios novatos y los de un nivel intermedio que no conocen conceptos avanzados de redes TCP/IP y seguridad.

Características:

- Interfaz de usuario de fácil uso.
- Maneja un protocolo basado en aplicaciones.
- Permite dar mantenimiento a las reglas de generadas para el cortafuegos.
- Las redes y nodos pueden ser divididos en zonas. Cada zona puede manejar sus propias políticas de seguridad.
- Soporta los siguientes protocolos: FTP, SSH, Telnet, Linuxconf, Corba, SMTP, DNS, Finger, HTTP, HTTPS, NFS, POP2, POP3, SUN RPC, Auth, NNTP, NETBIOS Name Service, NETBIOS Session Service, IMAP, Socks, Squid, pcANYWHEREstat, X Window System, Traceroute, ICQ, PowWow, IRC, PostgreSQL, MySQL, Ping, Quake, QuakeWorld, Quake 2, Who Is, Webmin, ICMP Source Quench, ICMP Redirect, Real Audio, Line Printer Spooler, syslog, NTP, NetMeeting, Gnutella, LDAP, LDAP-SSL, SWAT, Diablo II, Nessus, DHCP, AudioGalaxy, DirectPlay, Halflife, XDMCP and Telstra's BigPond Cable, CDDB, MSN Messenger, VNC, PPTP, Kerberos, klogin, kshell, NIS, IMAPS, POP3S, ISAKMP, CVS, DICT, AIM, Fasttrack, Kazaa, iMesh, Grokster, Blubster, Direct Connect, WinMX, Yahoo! Messenger, AH, ESP, Jabber, EsounD, Privoxy, eDonkey2000, EverQuest, ICP, FreeDB, Elster, Yahoo games, Legato NetWorker backups, Novell Netware 5/6 NCP, Bittorrent, rsync, distcc, Jabber over SSL, PGP key server, Microsoft Media Server y gkrellm.
- Los protocolos no soportados puede ser registrados directamente.
- Soporta configuraciones de ruteo.
- Se ejecuta sobre KDE 2 o 3, y Linux 2.2, 2.4, 2.6.
- Permite importar y exportar reglas del cortafuegos para ser

utilizadas en otras máquinas.

- Hace uso como base de la filosofía: “Lo que no está explícitamente permitido, está prohibido”.
- Licenciado bajo los términos de la licencia GPL.

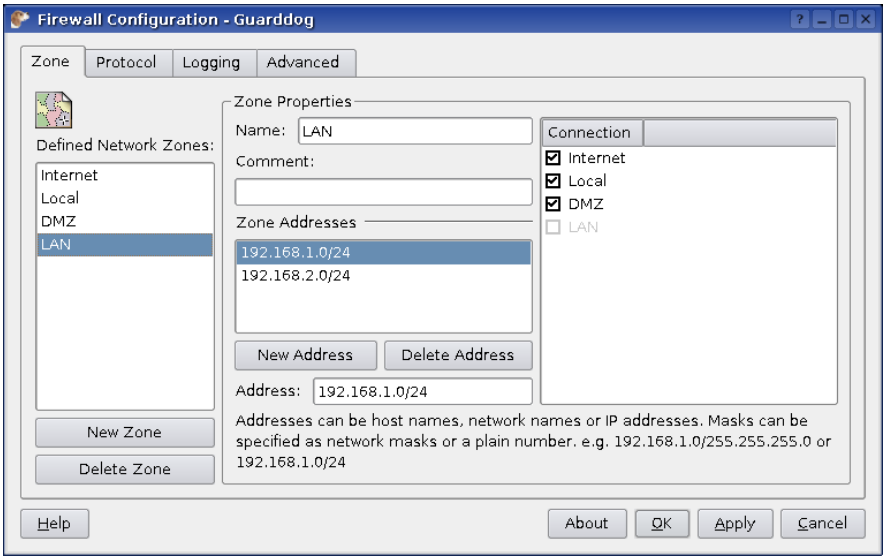


Imagen 1.8: Ejemplo de una pantalla de Guardog para configuración de zonas

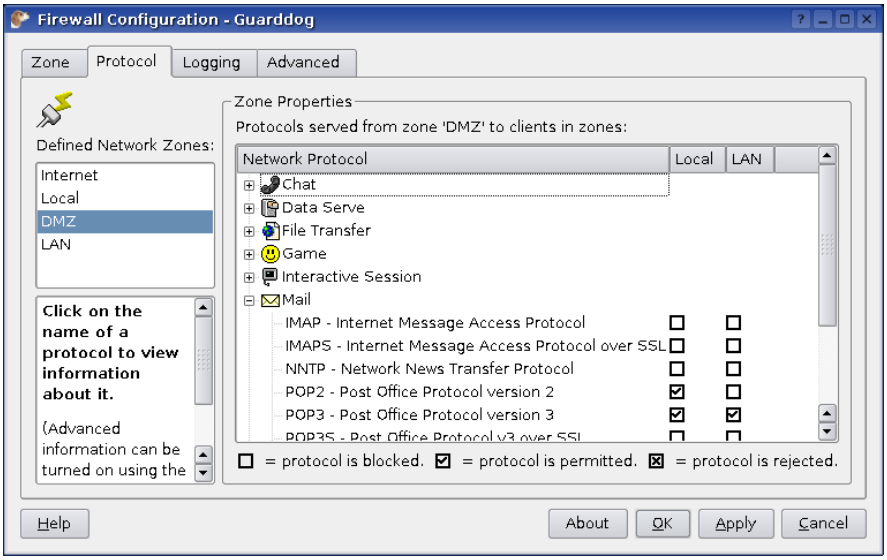


Imagen 1.9: Ejemplo de una pantalla de Guardog para configuración de protocolos

d) BASE

BASE – Basic Analysis and Security Engine – es una aplicación que provee una interfaz web que permite consultar y analizar las alertas que provienen del IDS Snort.

Características:

- Hace uso de un sistema de autenticación basado en roles, permitiendo configurar que tanta información deben de poder visualizar los demás usuarios.
- Permite modificar los archivos de configuración para no tener que trabajar con ellos directamente.

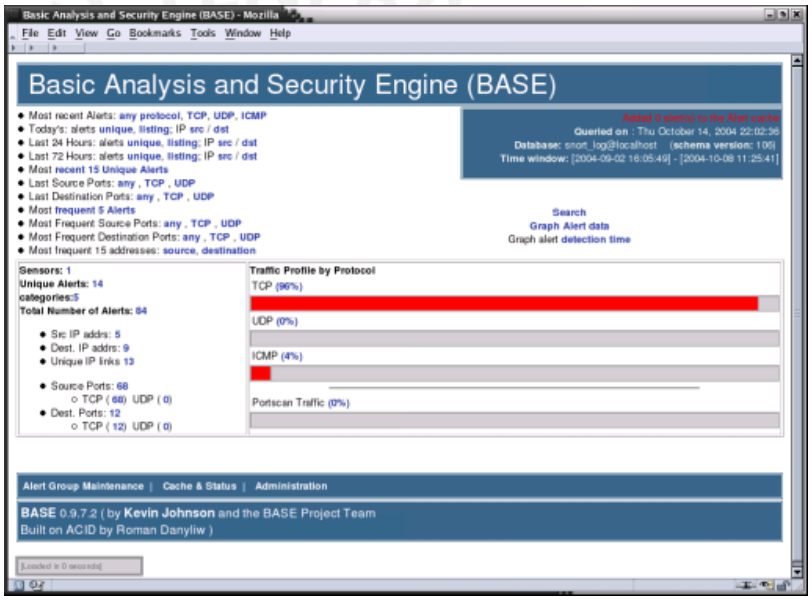


Imagen 1.10: Ejemplo de una pantalla de configuración de BASE

1.5. Descripción y sustentación de la Solución

Para disminuir los problemas de seguridad dentro de una red local ocasionados por una inadecuada configuración de los equipos a cargo de la seguridad, se propone como solución el análisis, diseño e implementación de una aplicación de fácil uso, que permita configurar y administrar funciones, tanto básicas como avanzadas, de las herramientas de seguridad Iptables, Squid y Snort.

El sistema busca brindar mayor seguridad a la información que manejan las PYMES, empresas medianas y usuarios con escasos conocimientos en temas referentes a seguridad de la información; a través de la disminución de las vulnerabilidades provocadas por una inadecuada configuración de los equipos de seguridad dentro de la red.

Características de la solución:

- La solución propuesta integrará las funciones de administración de las tres (3) herramientas de software libre destinadas al aseguramiento de redes: Iptables (cortafuegos), Squid (proxy – cache) y Snort (sistema de detección de intrusos).
- La aplicación presentará una interfaz gráfica amigable de fácil uso que permitirá llevar a cabo de una forma sencilla las tareas de configuración de las tres (3) herramientas.
- El aplicativo permitirá administrar de forma sencilla las reglas que utilizan las diversas herramientas que enmascara, a fin de que el usuario final pueda llevar a cabo configuraciones sin necesidad de interactuar directamente con la sintaxis de cada una de las tres aplicaciones.
- La aplicación generará archivos de configuración y mantendrá un histórico de las reglas generadas para cada una de las herramientas.
- La aplicación brindará opciones para exportar los archivos de configuración y de reglas definidas a fin de poder ser utilizados en otros equipos.
- Las reglas generadas por las aplicaciones funcionarán sobre interfaces ethernet, haciendo uso del protocolo IPv4.
- La solución será accesible vía web desde cualquier computadora que el usuario haya registrado previamente.
- El usuario podrá visualizar a manera de reportes, la información almacenada por las tres (3) herramientas, dentro de la bitácora del sistema.

Beneficios de su implementación:

- El usuario final contará con una herramienta simple y de fácil uso

que le permitirá administrar tareas de seguridad complejas de una forma rápida.

- Al poder ser exportadas las configuraciones y reglas, podrán ser implantadas rápidamente en otros equipos, lo cual traerá un ahorro de tiempo.
- Al contar con un histórico de reglas, el usuario tiene la posibilidad de retornar el sistema a una configuración previa a la actual, en caso ésta última no sea la más adecuada.
- Adicionalmente, al poseer una aplicación con una interfaz de fácil manejo, el usuario/administrador tendrá completo conocimiento de las reglas que está estableciendo; esto ayudará a mermar la cantidad de errores de seguridad provocados por configuraciones inadecuadas de los equipos.
- Finalmente, se contará con una serie de reportes que, tras un previo análisis, permitirán al usuario determinar que tan eficiente es la configuración establecida.

Capítulo 2: Análisis

En este capítulo se hará mención de la metodología que se utilizará a lo largo de todo el proyecto, la identificación de requerimientos y el análisis de la solución propuesta.

2.1. Definición de la metodología de la solución

a) La metodología Crystal Clear

Metodología de tipo ágil, creada por Alistair Cockburn a pedido del grupo de consultoría de IBM en el año 1991. A diferencia de muchas metodologías para la gestión del desarrollo del software, Crystal Clear se centra más en el factor humano que en los procesos y artefactos.

Esta metodología presenta las siguientes propiedades, siendo exigibles las tres (3) primeras:

- **Entregas Frecuentes:** Es imperante que al final de cada

iteración, se haga entrega a los usuarios el código debidamente testeado. Con ello se busca que el equipo de desarrollo pueda efectuar correcciones en la aplicación en caso la funcionalidad no cumpla con las necesidades del usuario. Es más sencillo efectuar los cambios sobre los entregables de cada iteración que sobre un proyecto completamente integrado.

- **Mejoras Reflejadas:** Consiste en identificar los puntos en los que el proyecto marcha bien, en los que marchan mal y los que pueden ser mejorados. Dichas mejoras serían aplicadas en la siguiente iteración. Esta propiedad debe de estar presente a lo largo de todo el proyecto.
- **Ósmosis en la Comunicación:** Propiedad imprescindible de esta metodología en la que los miembros abocados a la programación se distribuyen en dentro de una misma habitación o dentro de un ambiente que permita una comunicación fluida y sin obstáculos entre todo el equipo.
- **Seguridad Personal:** Consiste en brindar apoyo a los demás miembros del equipo. El equipo de trabajo está conformado por personas, cada una de las cuales tienen sus propias fortalezas, debilidades y problemas. Los aspectos negativos pueden solucionarse apoyando a los demás miembros y/o buscando apoyo en ellos.
- **Enfoque:** Destinar tiempo adecuado para poder cumplir con las prioridades que se tienen asignadas. Es importante apoyar a los demás miembros en las dudas y problemas que presenten; sin embargo, siempre hay que destinar un tiempo a las actividades asignadas.
- **Acceso sencillo para usuarios expertos:** Tener facilidad de acceso a usuarios expertos quienes verificarán que a lo largo de cada iteración se están cumpliendo sus expectativas. El acceso puede ser por visitas semanales y llamadas telefónicas; uno o más de un usuario experto dentro del equipo; desarrolladores especializados que han asumido el papel de usuarios (no es muy recomendado el uso de esta última).

- **Ambiente técnico con pruebas automatizadas** continuas a medida que se programa, gestión de la configuración (acompañada de una adecuada gestión de cambios), integración frecuente de los avances realizados.

El Proceso

Los proyectos son vistos como un ciclo continuo, ya que al concluir un proyecto, siempre habrá otro esperando a ser comenzado. Un ciclo de proyecto, dentro de Crystal Clear, está conformado por tres etapas:

Trazado de actividades

Esta primera etapa puede abarcar desde un par de días hasta un par de semanas. Dentro de este período, se llevan a cabo las siguientes tareas:

- **Forjar el corazón del equipo**, que consiste en definir los tres principales roles – el desarrollador líder, el patrocinador y el usuario clave.
- **Análisis exploratorio de 360°**, se lleva a cabo un estudio de viabilidad llevando a cabo una revisión de alto nivel de los pilares del esfuerzo aplicado en la etapa de desarrollo: el valor del negocio esperado, los requerimientos capturados, el ámbito del modelo, la tecnología a utilizarse, el plan de proyecto, la conformación del equipo y la metodología o acuerdos a utilizarse.
- **Trazando la metodología**, conformado por los convenios definidos al inicio del proyecto que pueden ir siendo mejorados y corregidos al inicio de cada nueva iteración.
- **Construyendo el plan de proyecto inicial**, se elabora el Mapa del Proyecto – diagrama equivalente al diagrama de precedencia – y el Plan de Proyecto – indicando los ciclos de entrega y los períodos que cada iteración involucra.

Dos o más ciclos de entrega

Cada ciclo de entrega involucra las siguientes actividades:

- **Recalibrar el plan de proyecto**, se evalúa tanto los requerimientos como el plan de proyecto al inicio de cada ciclo

de entrega con el objetivo de corregir desviaciones producidas respecto al alcance del proyecto.

- **Desarrollo en Iteración(es)**, cada iteración presenta una duración mayor a una semana y menor a tres meses. Se efectúan las siguientes labores:
  - Planear la iteración: se definen prioridades de las tareas
  - Ciclo Programación – testeo – integración
  - Ritual de fin de iteración
- **Entrega a usuarios reales**, se entrega la aplicación, resultado de la iteración, a un grupo de usuarios reales para así conocer si los resultados obtenidos hasta la fecha son un fiel reflejo de los requerimientos presentados por el usuario.
- **Reflexión** sobre el producto creado y los convenios usados.

Cierre de proyecto

En esta etapa final se llevan a cabo las siguientes tareas:

- **Pruebas** de aceptación
- Se prepara el **producto final** y el ambiente del usuario para llevar a cabo la implantación de la solución.
- **Recopilación de los conocimientos** adquiridos a fin de poder ser utilizados en un proyecto que se presente a futuro.

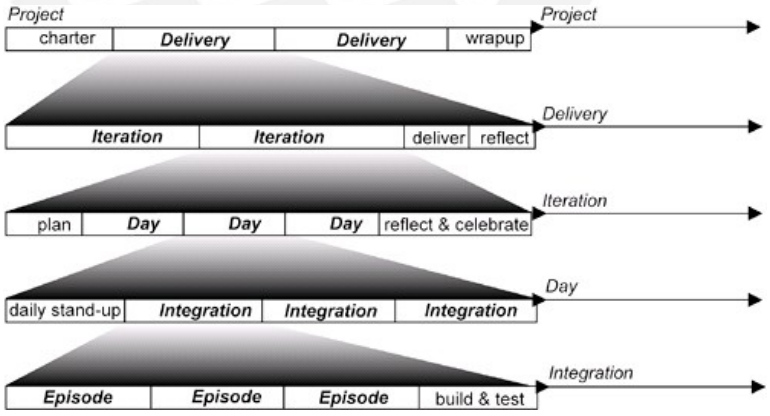


Imagen 2.1: Etapas de la metodología Crystal Clear

Los Roles

Dentro de esta metodología es posible identificar los siguientes roles.

Patrocinador: Es el encargado de definir cómo se distribuirá el capital dentro del proyecto, suele ser quien contrata los servicios del organismo desarrollador. En consecuencia, su decisión tendrá importancia al momentos de definir las prioridades en cuanto las funcionalidades a implementar.

Usuario experto: Adicionalmente a conocer las funciones desempeñadas por un usuario del sistema, cumple las funciones de un experto de negocio, es decir, conoce los procesos que se llevan a cabo dentro del negocio.

Líder de Diseño/Desarrollo: Miembro del equipo con gran conocimiento en temas de desarrollo de software. Es quien lleva el control del avance del equipo.

Equipo de Diseño/Desarrollo: Demás integrantes del equipo abocados a la tarea de desarrollar e implementar la solución. Suele haber un líder de diseño/desarrollo por cada tres miembros del equipo.

Coordinador del proyecto: Este rol puede ser ocupado tanto por el patrocinador como por el líder de diseño/desarrollo. Se encargará de enumerar las funciones a ser entregadas al final de cada ciclo de entrega. Lleva el control de las funciones a entregarse acorde al tiempo establecido en el cronograma.

Tester: Este rol es desempeñado por todos los miembros del equipo a lo largo de todo el proceso de implementación. Su función es la de reportar todos los bugs y errores detectados en el sistema siendo desarrollado.

Redactor: Este rol es llevado a cabo por los mismos miembros del equipo, quienes se encargarán de la redacción del manual de usuario a ser entregado junto con la solución.

El cuadro presentado a continuación brindará un breve resumen de la documentación generada por cada uno de los roles de la metodología.

	Trazado de Actividades	Ciclos de Entrega				Cierre Proyecto
		Recalibrar plan proyecto	Iteración	Entrega a usuarios reales	Reflexión sobre el producto y los acuerdos	
Patrocinador	Misión proyecto					
Equipo	Acuerdos equipo		Diseño Pantallas Ámbito Modelo Diagramas Diseño Código fuente	Empaquetar solución		
Coordinador	Mapa proyecto Plan lanzamiento Lista riesgos Plan iteración Cronograma		Estado del proyecto			
Líder Desarrollo	Arquitectura					
Usuario Experto	Casos de uso Requisitos Catálogo actores					
Tester			Reporte Pruebas / Errores			Reporte pruebas aceptación
Redactor				Manual de Usuario		

Tabla 2.1: Documentos generados por cada rol en la metodología Crystal Clear

b) Adaptando Crystal Clear al proyecto

La metodología a utilizar durante la presente tesis para llevar a cabo una adecuada gestión en el desarrollo del proyecto será Crystal Clear debido a las razones que se mencionan a continuación:

- Esta metodología se centra en el factor humano más que en los artefactos.
- Las constantes entregas de módulos a los usuarios al final de cada ciclo, permite recibir rápidas retroalimentaciones por parte del usuario lo cual da a conocer si el avance de la solución brinda las características esperadas.
- Existe una constante comunicación con el dueño del producto (patrocinador) lo cual permite resolver rápidamente cualquier duda presente en un requerimiento.
- Esta metodología ha sido creada pensando en pequeños equipos de trabajo; estando el equipo de desarrollo, del presente proyecto, conformado por un único miembro.

No obstante, para poder hacer uso de esta metodología de forma adecuada es necesario adaptarla según las necesidades del proyecto.

- El Equipo de Desarrollo estará conformado por un (1) único miembro, quien efectuará las labores de líder y equipo de diseño/desarrollo.
- Cada Iteración tendrá una duración de semana y media, siendo el ciclo de entrega de tres (3) semanas.
- El proyecto estará conformado por cinco (5) ciclos de entrega a lo largo de cada uno se implementará un módulo de funcionalidad independiente, con el objetivo que pueda ser entregado al usuario experto para ser validado.
- Al inicio del ciclo de entrega se llevará a cabo una reunión con el asesor con el objetivo de reestructurar el plan de proyecto en caso se presenten atrasos o correcciones en la funcionalidad previamente presentada.

Dentro de los artefactos que serán generados dentro de cada etapa al hacer uso de esta metodología, tenemos:

Análisis

- Mapa de Proyecto: Diagrama en donde se especifica las dependencias entre las funcionalidades a ser implementadas.

Establece el orden en que deben de implementarse las funcionalidades y cual es la dependencia existente entre los puntos a implementar.

- Plan de Lanzamiento: Se genera en base al Mapa del Proyecto. Se delimita las fechas en que se entregará cada funcionalidad.
- Catálogo de Riesgos: Consiste en un listado de riesgos que pueden amenazar la realización del proyecto o el cumplimiento de uno o muchos de sus objetivos. Dentro de la información básica que el catálogo debe de brindar tenemos el nombre del riesgo, consecuencias que puede provocar, probabilidad que ocurra y que medida preventiva o correctiva se debe de utilizar.
- Catálogo de actores: Contiene una tabla de dos columnas, en la primera se enumeran los actores y en la segunda las acciones que desempeñarán en el sistema a implementar.
- Lista de Requerimientos: Es un conglomerado de información que indica lo que se planea construir, el actor que ha de utilizarlo, el valor que provee dicha funcionalidad.
- Especificación de Casos de Uso: Es un documento en donde se describe los casos de uso presentes en el sistema. Para cada caso de uso debe describirse el flujo ideal – funcionalidad principal que el caso de uso representa – y cada uno de los flujos alternativos y excepcionales – funcionalidades alternativas o flujos que presentan errores – si es que los hubiera

Diseño

- Modelo del Dominio: Diagrama de clases que muestra las principales entidades del sistema y la forma en que se interrelacionan.
- Pantallas: Gráficos que muestran el aspecto que tendrá cada pantalla del sistema.
- Arquitectura del Sistema: Este diagrama muestra los principales componentes e interfaces del sistema.

Construcción

- Catalogo de Pruebas: Documento en donde se lleva el listado de pruebas que deben de llevarse a cabo de forma exitosa para poder asumir que una funcionalidad se haya implementada de forma exitosa.
- Manual de Usuario: Documentación en donde se da a conocer al usuario los pasos que debe de seguir para poder llevar a cabo exitosamente una acción en el sistema.

2.2. Identificación de los requerimientos

A continuación se presenta la lista de requerimientos tanto funcionales como no funcionales que el aplicativo deberá de cumplir.

a) Requerimientos funcionales

La lista de requerimientos funcionales se halla dividida en cuatro (4) grupos cada uno de los cuales representa cada módulo en los que el proyecto se hallará subdividido.

Dentro de la información que muestra la cabecera de cada tabla de requerimientos, encontramos los siguientes campos: Número de requerimiento, la descripción del requerimiento, la prioridad del requerimiento – que pude tomar valores del uno (menor prioridad) al cinco (mayor prioridad) – y el tipo de requerimiento – que puede ser Exigible (E) o Deseable (D).

Módulo Central			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FC001	La aplicación permitirá administrar el listado de los módulos del sistema, habilitándolos o deshabilitándolos.	4	E
FC002	La aplicación permitirá asignar un alias a las diferentes interfaces detectadas en el equipo.	4	E
FC003	La aplicación permitirá activar y desactivar la opción de reenvío de paquetes.	3	E
FC004	La aplicación permitirá exportar e importar las configuraciones y reglas establecidas por cada módulo del sistema.	4	E

Módulo Central			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FC005	La aplicación permitirá mantener la lista de usuarios que podrán iniciar sesión remota en el sistema.	3	E
FC006	La aplicación permitirá administrar una lista de direcciones IP hábiles que podrán iniciar sesión remota en el sistema.	4	E
FC007	La aplicación permitirá manejar una bitácora que permita auditar las sesiones iniciadas, almacenando información como usuario, IP de origen, sistema operativo y hora.	4	E
FC008	La aplicación permitirá iniciar y detener los servicios relacionados con cada uno de los módulos activados.	3	E
FC009	La aplicación permitirá mantener la lista de los nodos y las zonas existentes en la red	4	E

Tabla 2.2: Requerimientos Funcionales - Módulo Central

Módulo Iptables			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FI001	La aplicación permitirá configurar las opciones básicas del Iptables.	5	E
FI002	La aplicación podrá trabajar con una serie de reglas predeterminadas para determinados programas.	5	E
FI003	La aplicación permitirá administrar la tabla nat y la tabla filter.	5	E
FI004	La aplicación permitirá mantener las reglas de Iptables.	5	E
FI005	La aplicación permitirá establecer políticas para cada cadena registrada en Iptables.	4	E
FI006	Las reglas establecidas por la aplicación podrán ser aplicables a un rango de direcciones IP.	2	E
FI007	La aplicación podrá registrar reglas de filtrado por MAC	2	E
FI008	La aplicación permitirá administrar el registro histórico de reglas y configuraciones generadas por el módulo Iptables	5	E
FI009	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Iptables.	3	E

Tabla 2.3: Requerimientos Funcionales - Módulo Iptables

Módulo Squid			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FQ001	La aplicación permitirá configurar las opciones básicas del Squid.	5	E

Módulo Squid			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FQ002	La aplicación podrá trabajar con una serie de reglas preestablecidas para determinados programas.	5	E
FQ003	La aplicación permitirá habilitar y configurar las funcionalidades básicas de la caché y del proxy.	4	E
FQ004	La aplicación permitirá registrar una lista de control de acceso.	5	E
FQ005	La aplicación permitirá registrar grupos de control de acceso.	5	E
FQ006	La aplicación permitirá administrar reglas de control de acceso a través de las cuales se indicarán las listas y grupos de control a quienes se les permitirá acceder o se les negará el acceso a determinados recursos.	5	E
FQ007	La aplicación permitirá administrar el registro histórico de reglas y configuraciones generadas por el módulo Squid.	4	E
FQ008	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Squid.	3	E

Tabla 2.4: Requerimientos Funcionales - Módulo Squid

Módulo Snort			
Código	**Requerimiento**	**Prioridad**	**Tipo**
FS001	La aplicación permitirá configurar las opciones básicas del Snort para que trabaje en modo NIDS	5	E
FS002	La aplicación permite definir la lista de preprocesadores que Snort utilizará.	3	E
FS003	El módulo trabajará en forma predeterminada con las reglas proporcionadas por la comunidad de usuario de Snort, debidamente validadas por *Sourcefire*, empresa a cargo de la implementación y distribución de Snort.	5	E
FS004	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Snort.	3	E

Tabla 2.5: Requerimientos Funcionales - Módulo Snort

b) Requerimientos no funcionales

Código	Requerimiento	Tipo
FN001	La aplicación permitirá acceso remoto vía web.	E
FN002	La versión en desarrollo de la aplicación soportará interfaces	E

Código	Requerimiento	Tipo
	ethernet.	
FN003	La aplicación hará uso del protocolo IPv4.	E
FN004	La aplicación presentará una interfaz que sea fácil de utilizar.	E
FN005	La aplicación estará escrita en el lenguaje PHP 5.	E
FN006	El instalador de la aplicación estará escrito en el lenguaje Perl.	E
FN007	La aplicación se ejecutará sobre el servidor web Apache versión 2.	E
FN008	La aplicación se ejecutará prioritariamente en sistemas operativos GNU/Linux basados en RedHat.	E

Tabla 2.6: Requerimientos No Funcionales

2.3. Identificación *de Casos de Uso*

Tomando como base el listado de requerimientos funcionales del proyecto, se han identificado los casos de uso listados a continuación.

Módulo Central		
Código	**Caso de uso**	**Requerimientos Asociados**
CUC01	Mantener usuarios	FC005
CUC02	Mantener servicios	FC008
CUC03	Mantener módulos	FC001, FC004
CUC04	Mantener subred	FC009
CUC05	Iniciar sesión	FC005, FC006
CUC06	Generar reporte de sesiones	FC007
CUC07	Administrar directivas de seguridad	FC006
CUC08	Mantener configuración de red	FC002, FC003, FC009

Tabla 2.7: Casos de Uso - Módulo Central

Módulo Iptables		
Código	**Caso de uso**	**Requerimientos Asociados**
CUI01	Configurar Iptables	FI001
CUI02	Mantener reglas Iptables	FI002, FI003, FI004, FI005, FI006, FI007
CUI03	Mantener registro histórico	FI008
CUI04	Generar reporte Iptables	FI009

Tabla 2.8: Casos de Uso - Módulo Iptables

Módulo Squid		
Código	**Caso de uso**	**Requerimientos Asociados**
CUQ01	Configurar Squid	FQ001, FQ003
CUQ02	Mantener reglas Squid	FQ002, FQ004, FQ005, FQ006
CUQ03	Mantener registro histórico	FQ007
CUQ04	Generar reporte Squid	FQ008

Tabla 2.9: Casos de Uso - Módulo Squid

Módulo Snort		
Código	**Caso de uso**	**Requerimientos Asociados**
CUS01	Configurar Snort	FS001, FS002
CUS02	Mantener reglas Snort	FS003
CUS03	Generar reporte Snort	FS004

Tabla 2.10: Casos de Uso - Módulo Snort

2.4. Dominio del modelo

A continuación se presenta una breve descripción de las clases, separadas por módulos, que conforman el dominio del modelo del proyecto de tesis.

Módulo Central		
Código	**Clase**	**Descripción**
CC001	Servidor	Esta clase contiene la información básica sobre las configuraciones de red del servidor en donde se ejecuta la aplicación.
CC002	Usuario	Esta clase contiene información del usuario que iniciado sesión en la aplicación.
CC003	Interfaz	Esta clase representa a una zona dentro de la red local, conectada al servidor a través de una interfaz de red.
CC004	Nodo	Esta clase representa a un nodo dentro de una zona de la red local.
CC005	IPv4Valida	Esta clase representa una IP desde la cual es permitida iniciar sesión en la aplicación.
CC006	RegistroHistorico	Esta clase contiene información de lo intentos de inicio de sesión en la aplicación.
CC007	Subred	Esta clase representa una subred dentro de la red

Módulo Central		
Código	**Clase**	**Descripción**
		local.

Tabla 2.11: Dominio del Modelo - Módulo Central

Módulo Iptables		
Código	**Clase**	**Descripción**
CI001	Iptables	Esta clase representa las configuraciones básicas de la herramienta Iptables.
CI002	ReglaIptables	Esta clase representa cada una de las reglas que pueden ser registradas en la aplicación Iptables.
CI003	Cadena	Esta clase representa una cadena de Iptables.
CI004	Table	Esta clase representa una de las tablas propias de la aplicación Iptables.
CI005	HistoricoIptables	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Iptables.
CI006	Accion	Esta clase representa las acciones a ejecutar sobre cada regla que se defina.
CI007	Categoria	Esta clase representa las categorías en las que se clasifican las reglas predefinidas.
CI008	DetalleReglaPredefinida	Esta clase representa la regla predefinida tal cual es entendida por la aplicación Iptables.
CI009	Estado	Esta clase representa los estados de las conexiones.
CI010	FechaActivacionIptables	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro del registro histórico de configuraciones de Iptables.
CI011	Politica	Esta clase representa las políticas predefinidas de cada cadena del Iptables.
CI012	Protocolo	Esta clase representa cada uno de los protocolos soportados por la aplicación.
CI013	ReglaPredefinida	Esta clase representa la regla predefinida que el administrador de la red quiere utilizar.

Tabla 2.12: Dominio del Modelo - Módulo Iptables

Módulo Squid		
Código	**Clase**	**Descripción**
CQ001	Squid	Esta clase representa las configuraciones básicas de la herramienta Squid.

Módulo Squid		
Código	**Clase**	**Descripción**
CQ002	ReglaPredefinida	Esta clase representa una regla predefinida que puede ser utilizada lista para las configuraciones de las opciones de proxy de la herramienta Squid.
CQ003	ListaControlAcceso	Esta clase agrupa un conjunto de reglas Squid, tanto para la cache como para el proxy.
CQ004	ReglaSquid	Esta clase representa cada una de las reglas que pueden ser registradas en el Squid.
CQ005	HistoricoSquid	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Squid.
CQ006	Accion	Esta clase representa la acción a ejecutar sobre una regla registrada.
CQ007	FechaActivacionSquid	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro del registro histórico de configuraciones del Squid.
CQ008	PuertoSquid	Clase que representa el puerto sobre el que se ejecuta el Squid.
CQ009	TipoACL	Representa los tipos de Lista de Control de Acceso.
CQ010	TipoAcceso	Clase que representa el tipo de acceso (web o caché)
CQ011	Valor	Listado de valores sobre los que se ejecutan las reglas registradas.

Tabla 2.13: Dominio del Modelo - Módulo Squid

Módulo Snort		
Código	**Clase**	**Descripción**
CS001	Snort	Esta clase representa las configuraciones básicas de la herramienta Snort.
CS002	Servicio	Esta clase representa los servicios que la herramienta Snort se halla manejando; así como los puertos a los que se hallan relacionados.
CS003	Preprocesador	Esta clase representa los preprocesadores que el Snort está utilizando.
CS004	ReglaPredefinida	Esta clase representa cada una de las reglas que pueden ser registradas en la herramienta Snort.
CS005	HistoricoSnort	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Snort.
CS006	FechaAplicacionSnort	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro

Módulo Snort		
Código	**Clase**	**Descripción**
		del registro histórico de configuraciones del Snort.
CS007	Libreria	Esta clase representa las librerías que el Snort está utilizando.
CS008	Parametro	Esta clase representa los parámetros que el tipo de preprocesador puede recibir.
CS009	TipoLibreria	Esta clase representa los tipos de librerías disponibles.
CS010	TipoPreprocesador	Esta clase representa los tipos de preprocesadores disponibles.
CS011	TipoServicio	Esta clase representa los tipos de servicios disponibles.
CS012	TipoValor	Esta clase representa el tipo de fichero que el Snort utiliza como librería.

Tabla 2.14: Dominio del Modelo - Módulo Snort

2.5. Análisis de la solución

A continuación se presentará un breve análisis de la viabilidad del proyecto; para ello se efectuará tanto un análisis de costo – beneficio de la construcción e implantación de la solución desde el punto de vista del usuario final; así como un análisis de factores técnicos y económicos que deben de ser identificados por parte del equipo de trabajo.

Análisis de Costo – beneficio

La implementación de la solución traerá como beneficio una serie de mejoras al usuario, tanto en el proceso de administrar las diversas herramientas de seguridad; así como llevar a cabo un mejor control de los paquetes que viajan dentro de la red local.

Costo	Beneficio
El precio del computador en donde se instalen los aplicativos para la administración de la seguridad en la red local.	Las herramientas utilizadas para la administración de la seguridad tienen un costo menor a muchas soluciones comercializadas actualmente por diversas empresas.
El costo del personal capacitado para la implementación de la solución.	Menores tiempos requeridos para la configuración de las herramientas de seguridad.
	Menores riesgos que se efectúe una inadecuada configuración de la seguridad debido a errores de usuario.
	Mejora del nivel de seguridad de la información que se transmite dentro de la red local.

Tabla 2.15: Análisis Costo Beneficio

Análisis de factores tecnológicos

A lo largo del proyecto se hará uso de diversas tecnologías que permitirán facilitar el trabajo para poder llevar un adecuado control del proyecto durante el proceso de planificación y delegación de tareas, la etapa de codificación para mantener un adecuado control de las versiones del código que se va generando y una administración adecuada de los errores que se vayan detectando.

Herramienta para la gestión del proyecto

Las herramientas de gestión de proyectos, tienen como objetivo cubrir diversas necesidades al momento de llevar a cabo tareas de planificación y administración. Dentro de las funciones que podemos hallar, tenemos las de calendarización, control de costos, administración de recursos, comunicación, entre otras.

El siguiente cuadro comparativo mostrará tres (3) de estas aplicaciones a fin de poder elegir la más adecuada a utilizar a lo largo del proyecto.

	MS Project	dot-project	GanttProject
Multiplataforma		X	X
Escritorio	X		X
Basado en web		X	
Multiusuario (colaborativo)		X	
Manejo de lista de tareas	X	X	X
Lista de Eventos		X	
Calendarios	X	X	

Tabla 2.16: Comparación de herramientas de gestión de proyecto

Luego de analizar las características que presenta cada una de las herramientas mencionadas en función de las opciones necesarias para el proyecto, se ha elegido el dot-project como la herramienta de gestión a utilizar.

Controlador de versiones

Un sistema de control de versiones es una herramienta comúnmente utilizada dentro de la ingeniería y en el desarrollo de aplicaciones. Permite efectuar un seguimiento de los cambios realizados sobre el código y los documentos dentro de un proyecto.

El siguiente cuadro comparativo mostrará dos (2) tecnologías utilizadas para manejar el control de versiones, el CVS y el SVN.

	CVS	SVN
Commits Atómicos		X
Renombrar Archivos y directorios		X
Mover Archivos y Directorios		X
Copiar Archivos y Directorios		X
Replicar directorios remotos	X	X
Permisos en los directorios	X	
Revisión del registro histórico	X	X
Documentación	X	X
Facilidad de configuración	X	
Facilidad de uso	X	X
Multiplataforma	X	X
Velocidad		X

Tabla 2.17: Comparación de herramientas de control de versiones

Como herramienta de Control de Versiones se ha elegido al SVN debido a que brinda muchas más opciones frente al CVS lo cual será útil a lo largo del desarrollo del proyecto.

Herramienta para seguimiento de errores

Esta aplicación permite a los desarrolladores llevar un registro de todos los errores detectados en el programa en desarrollo, a fin de poder aplicarles un adecuado seguimiento desde que el error es detectado, hasta el momento en que es corregido.

A continuación se mostrará un cuadro comparativo entre dos (2) aplicaciones muy utilizadas para administrar los errores detectados.

	Mantis	Bugzilla
Fácil instalación	X	
Fácil configuración	X	
Aplicación web	X	X
Aplicación ligera	X	X

Tabla 2.18: Comparación de herramientas para seguimiento de errores

La herramienta Mantis, es la herramienta escogida para llevar a cabo un adecuado seguimiento a los errores detectados en la aplicación.

Lenguaje de Programación

La elección de un adecuado lenguaje de programación puede influir en el desarrollo de la solución en cuanto las características de portabilidad, consumo de recursos, entre otras. Debido a que se requiere que la solución funcione vía web, se se establecerá un cuadro comparativo entre tres (3) lenguajes utilizados para crear aplicaciones que cumplen con dicho requerimiento.

	PHP	Java	ASP
Ejecución bajo Linux	X	X	
Bajo consumo de memoria	X		
Bajo consumo de procesador	X		
Rapidez de ejecución	X	X	X
Gran variedad de librerías disponibles		X	X
API abierta y documentada	X	X	
Seguridad	X	X	X

Tabla 2.19: Comparación de lenguajes de programación

Tras comparar las características de cada uno de los lenguajes, se ha optado por hacer uso del lenguaje PHP.

Herramienta CASE

Son herramientas que brindan facilidades para el mantenimiento y el desarrollo de aplicaciones. Se presentará un cuadro comparativo de cuatro (4) herramientas CASE para la elaboración de diagramas UML.

	Umbrello	Argo UML	Star UML	Rational Rose
Soporte de repositorio				
Generar código	X	X	X	X
Ingeniería reversa	X		X	X
Soporte de UML 2.0	X		X	X
Selección fácil de clases y métodos	X		X	X
Diseño del modelo de datos	X		X	
Navegabilidad entre modelos			X	X
Exportar a imagen	X	X	X	X
Robusto			X	X
Ligero	X	X	X	
Soporte Linux	X	X		
Uso del formato XMI	X			

Tabla 2.20: Comparación de herramientas CASE

Luego de revisar las características de cada una de las herramientas, debido a los puntos que cada uno complementa, se ha optado por utilizar Umbrello.

Entornos de Desarrollo Integrados (IDE)

Son aplicaciones utilizadas para crear programas. En la mayoría de casos, un IDE está conformado por un editor de código fuente, un compilador o un intérprete y usualmente una opción para ejecutar paso a paso. Asimismo puede incluir funcionalidades adicionales, como integración con una herramienta de control de versiones, opciones de autocompletado de texto, editor de interfaces gráficas, entre otras.

A continuación se presentará un cuadro en donde se comparará cuatro (4) IDEs que pueden ser utilizados para el desarrollo del proyecto.

	Kate	KDevelop	Netbeans (PHP Pluggin)	Quanta +	Eclipse (PHP Pluggin)
Administración de Proyectos		X	X	X	X
Poder visualizar el árbol de directorios	X	X	X	X	X
Depurador		X	X		X
Barra de herramientas HTML			X	X	
Herramientas de arrastrar y colocar			X	X	
Inicio rápido	X	X		X	
Autoguardado	X	X		X	X
Integración con un manejador de versiones		X	X	X	X
Lista de pendientes (TODO)			X	X	X
Autocompletado	X	X	X	X	X
Integración con HTML			X	X	
Integración con CSS				X	
Soporte PHP	X	X	X	X	X
Comparación de archivos	X	X		X	

Tabla 2.21: Comparación de Entornos de Desarrollo Integrados

Luego de comparar las características que cada una de las aplicaciones posee, se ha optado por utilizar Quanta + como herramienta IDE.

Base de Datos

Son herramientas ampliamente utilizadas para almacenar datos para su

ser utilizados posteriormente. A continuación, se presentará un cuadro comparativo de tres (3) bases de datos ampliamente utilizadas en proyectos de diversa índole.

	Sqlite 3	MySQL 5	PgSql 8
Bajo consumo de recursos	X		
Fácil instalación	X	X	X
No depende de un servicio	X		
Fácil configuración	X	X	
Manejo de consultas	X	X	X
Manejo de transacciones	X	X	X
Soporte de integridad referencial		X	X
Drivers de conectividad con PHP	X	X	X
Portabilidad	X	X	X
Soporte de procedimientos		X	X

Tabla 2.22: Comparación de Bases de Datos

Luego de comparar las características de cada una de las aplicaciones, se ha optado por utilizar Sqlite 3, especialmente por su bajo consumo de recursos.

Análisis económico

El llevar a cabo cualquier proyecto, involucra salida de capital económico para poder solventar gastos como el pago al personal a cargo de la implementación de la solución, licencias, entre otros factores. A continuación se mostrará una tabla que resume los principales egresos a fin de implementar la solución.

Concepto					**Subtotal (S/.)**	
Mano de obra		Horas	Costo por hora			
	Generalidades	192	10	00	1920	00
	Análisis de la solución	152	20	00	3040	00
	Diseño de la solución	64	20	00	1280	00
	Construcción de la Solución	442	15	00	6630	00
	Conclusiones	26	15	00	390	00
					13260	**00**
Otros Gastos						
	Máquina de desarrollo				400	00
	Máquina de pruebas				200	00
	Luz, Internet				800	00
	Artículos de oficina (hojas, lapiceros, tinta)				150	00
					1550	**00**
Total					**14810**	**00**

Tabla 2.23: Análisis Económico

Capítulo 3: Diseño

En este capítulo se definirá la arquitectura que se utilizará para implementar la solución, brindándose información tales como las capas en las que se dividirá la solución, aspectos de seguridad, entre otros. Asimismo se definirán los criterios utilizados para elaborar el diseño de las interfaces de las diversas pantallas del sistema.

3.1. Arquitectura de la Solución

La arquitectura a utilizar será Web, con ello el usuario podrá tener acceso remoto a una interfaz gráfica para poder llevar a cabo la configuración de las herramientas de seguridad. Como principal requerimiento, el usuario deberá de hacer uso de un navegador de Internet.

La aplicación se hallará conformada por un módulo central que permitirá, dentro de su lista de funciones, administrar una serie de módulos complementarios cada uno de los cuales aportará nuevas funcionalidades

al sistema. Esto trae como beneficio la posibilidad de crear nuevos módulos con el objetivo de agregar nuevas funcionalidades a la herramienta. A lo largo del proyecto de tesis se implementarán tres (3) módulos complementarios, módulo Iptables, módulo Squid y módulo Snort.

Tanto la aplicación como las herramientas se hallarán ubicadas en un servidor GNU/Linux que se halle ejecutando el servicio web Apache, el cual, contará con el módulo PHP.

En la Imagen 3.1 se puede apreciar la arquitectura propuesta para el presente proyecto.

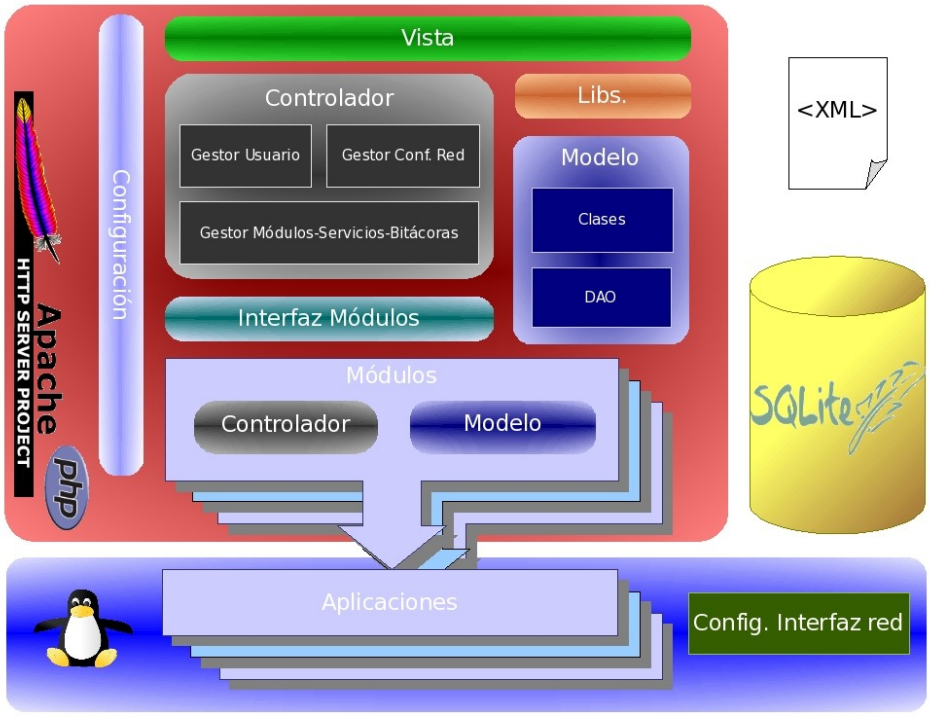


Imagen 3.1: Arquitectura de la solución

a) Descripción de la arquitectura de capas de la solución.

La aplicación está conformada por capas, cada una de las cuales aporta una funcionalidad al sistema. La lista de capas se detallan en la tabla siguiente:

Capa	Función que desempeña
Vista	Esta capa se encarga de presentar al usuario una interfaz con la que podrá interactuar, permitiéndole ingresar valores a la aplicación y visualizar las respuestas de la misma.
Controlador	Dentro de esta capa se hallan las clases que permiten manejar la lógica de negocio de todas las funcionalidades involucradas en el sistema. Asimismo será la encargada de controlar la interacción entre el módulo central de la aplicación con cada uno de los módulos complementarios.
Modelo	Esta es la capa que almacena los objetos de negocio de los que hace uso la aplicación a implementar, así como las clases que permiten llevar a cabo la persistencia contra una base de datos.
Módulo	Esta capa contendrá los diversos módulos integrados al sistema, cada uno de los cuales aportará nuevas funcionalidades al mismo.

Tabla 3.1: Capas de la Aplicación

b) Esquema de seguridad.

Al ser una aplicación que efectúe tareas de configuración de herramientas de seguridad dentro de una red local, este sistema debe de brindar la seguridad necesaria para que el acceso a las configuraciones sea restringido únicamente a personal autorizado.

Por ello se han propuesto las siguientes soluciones:

Servidor Web Virtual: La aplicación se ejecutará sobre un servidor web virtual (virtual host), que permitirá establecer el puerto que los usuarios utilizarán para comunicarse con la aplicación.

Cifrado de contraseñas: Para aminorar los riesgos, en caso un usuario no autorizado consiga tener acceso a las contraseñas almacenadas en el sistema, se ha optado por utilizar un mecanismo de cifrado sobre las contraseñas que serán almacenadas por la aplicación.

Perfiles y permisos: La aplicación manejará dos tipos de perfiles (administrador y usuario). El primero se caracteriza por poseer un

control absoluto sobre cada una de las funciones que la aplicación posee; mientras que el segundo puede efectuar labores de mantenimiento y generación de reportes.

Inicios de sesión: Se propone que todo aquel usuario que desee acceder al sistema debe de haber sido identificado previamente haciendo uso de una pantalla de inicio de sesión, en donde ingresará su nombre de usuario y contraseña.

Filtro de direcciones IP: Se propone establecer un filtro de direcciones IP, a nivel de aplicación, y de Iptables. Permitiendo el ingreso al sistema únicamente a través de máquinas cuya dirección IP haya sido registrada y autorizada.

Bitácoras de inicio de sesión: Se haría uso de una bitácora que almacenaría los intentos exitosos y fallidos de inicio de sesión con fines de auditoría.

c) Patrones a utilizar.

Modelo Vista Controlador (MVC): El patrón MVC está siendo utilizado para separar la capa de presentación (Vista) de los objetos del negocio (Modelo) y la lógica del negocio (Controlador), permitiendo así independizar la implementación de la lógica de la aplicación del diseño de las pantallas.

Data Mapper: Patrón utilizado para manejar la persistencia. Usando este mecanismo, los objetos que pertenecen al modelo desconocen la existencia de una base de datos. Una clase intermedia es la encargada de transferir los datos del modelo a la base de datos y viceversa.

Campo de Identidad: Consiste en almacenar el Id dentro del objeto; ello permite efectuar consultas haciendo uso de los Id, lo cual conlleva a un ahorro de tiempo.

Mapeo de Llave Foránea: Consiste en representar las llaves foráneas como relaciones entre dos objetos.

Solitario: Este patrón es utilizado en el ahorro de recursos; su objetivo es el de evitar instanciar una nueva clase si es que esta ya ha sido instanciada anteriormente.

Fachada: El patrón fachada permite establecer una interfaz a través de la cual se lleve a cabo la comunicación entre dos módulos. Es utilizada en el presente proyecto para garantizar la comunicación entre los módulos complementarios y el módulo central.

Template View: Este patrón permite generar páginas web dinámicas insertando etiquetas (ejemplo: etiquetas PHP) dentro del código HTML.

Front Controller: Este patrón establece la creación de una clase Controlador por cada Vista que se genere. Cada una de estas clases efectuará diferentes procedimientos dependiendo del evento que se accione en su respectiva Vista.

Server Session State: La utilidad de este patrón radica en poder almacenar información del cliente en el servidor, para poder hacer uso de estos valores más adelante.

d) Estructura del archivo de configuración del sistema.

Los archivos de configuración, tanto del módulo central, como de los módulos complementarios brindan la siguiente información.

- Las rutas a los directorios que conforman el módulo.
- El nombre de la página predeterminada del módulo.
- El nombre y la ruta del servicio del sistema al cual el módulo se halla asociado.

3.2. Diseño de la Interfaz Gráfica

A continuación se definirán los criterios utilizados para la elaboración de las pantallas y se dará a conocer los tipos de pantallas que conforman la aplicación.

a) Criterios utilizados para el diseño de la interfaz gráfica

La interfaz gráfica es la parte de la aplicación con la que el usuario interactúa. El diseño de la ventana debe de ser tal que el usuario se sienta cómodo y seguro al momento de utilizarlo. Para las pantallas del presente proyecto, se utilizaron los criterios listados a continuación:

Estética: La pantalla debe de ser estéticamente agradable. Muchos usuarios se muestran disconformes con una solución que no les es agradable a la vista. Como consecuencia esto puede llevar a que el usuario se niegue a hacer uso de la aplicación lo cual significaría el fracaso del proyecto.

Es por esa razón que uno de los factores que se han considerado en cuanto al diseño de la pantalla, es la combinación de los colores. Se ha definido el verde como color básico, siendo éste el principal color utilizado dentro del diseño.

En la Imagen 3.2 se puede apreciar los principales colores que son utilizados en el diseño de las pantallas.

Imagen 3.2: Mapa de colores

Intuitivo: Las opciones que se presenten en pantalla deben de ser fáciles de comprender y aprender por parte del usuario. La aplicación no debe de confundir al usuario en cuanto uso.

Balance: Los componentes mostrados en la pantalla deben de estar agrupados de tal forma que no dejen espacios vacíos; sin embargo, la pantalla no debe de verse sobrecargada.

Simetría: La alineación de los elementos tanto de forma horizontal como vertical, mejora la apariencia de la pantalla y el orden de la misma.

La siguiente imagen muestra un ejemplo de la simetría de los elementos dentro de una pantalla.

Imagen 3.3: Simetría de los elementos

Predecible: Tomando como premisa que el usuario siempre ha de usar algunas opciones más que otras; las opciones más importantes o aquellas cuyo uso pueda ser más frecuente deben de ser ubicados en puntos de fácil acceso.

A continuación se presenta la imagen del menú que la aplicación utilizará.

Imagen 3.4: Menú de la aplicación

Secuencial: Las opciones que se presenten deben de hallarse ordenadas. Por ejemplo si se habla de un menú que muestra opciones para ejecutar un proceso. Las opciones del menú deben de hallarse ordenadas de acuerdo a los pasos del proceso.

Económico: Los pasos para que un usuario pueda efectuar una acción deben de ser los menores posibles.

Agrupado: Los componentes presentados en la pantalla deben de hallarse agrupados de manera que guarden relación entre ellos. De esta forma se evita la confusión del usuario al desconocer la funcionalidad de alguno de los componentes.

En la siguiente imagen se puede ver un ejemplo de como se agruparán los componentes dentro de la aplicación.

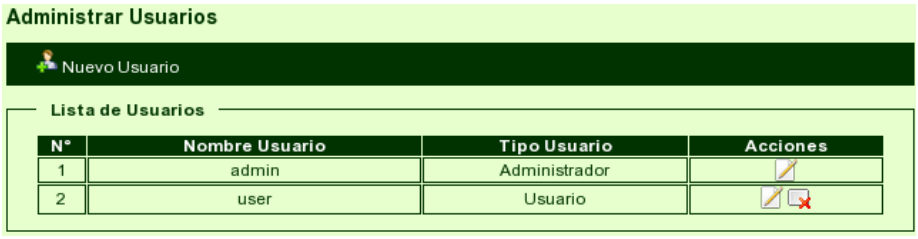


Imagen 3.5: Componentes agrupados

b) Tipos de pantallas

La aplicación hará uso de los siguientes tipos de pantallas:

Pantalla Principal: Es la pantalla principal que la aplicación mostrará luego de haber iniciado sesión. Esta pantalla contará con un menú de accesos rápidos a través del cual se podrá acceder a diversas funcionalidades de la aplicación.

Pantalla Secundaria: Son las pantallas que brindarán diversas opciones como, por ejemplo, la generación de reportes, visualización y administración de registros históricos, administración de usuarios.

Pantalla de Propiedades: Pantalla a través de la cual se llevarán a cabo las configuraciones de las herramientas de seguridad que la aplicación administra.

Cajas de diálogo: Son pequeñas ventanas que el sistema utilizará para brindar información sobre un error o un evento ocurrido; o para solicitar una confirmación al usuario previamente a ejecutar una acción.

c) Diseño estructural de las pantallas

Las pantallas de la aplicación presentarán una estructura conformada por cuatro (4) secciones bien definidas, como puede apreciarse en la siguiente imagen.

Imagen 3.6: Estructura de la página

A continuación, se dará una breve descripción de cada una de las secciones presentadas.

Cabecera de la página: La cabecera de la página muestra el nombre de la aplicación, así como las opciones para cerrar sesión.

Pie de página: El pie de página muestra información que permita saber la fecha de creación de la aplicación.

Menú: El menú es la herramienta dentro de la aplicación que permitirá a los usuarios navegar por la misma.

Sección de contenidos: Es dentro de esta sección en donde el usuario interactuará con las herramientas y opciones que la aplicación le provea.

Capítulo 4: Construcción

En este capítulo se definirá el lenguaje utilizado para llevar a cabo la implementación de la solución; así como los estándares definidos y las pruebas elaboradas con el objetivo de asegurar la calidad y adecuado funcionamiento de la aplicación.

4.1. Construcción

En este punto se detallarán las decisiones tomadas para la una adecuada construcción de la solución.

a) Entorno en el que se ejecutará la aplicación

La aplicación a desarrollar se ejecutará sobre un servidor GNU/Linux. Dicho sistema ha sido elegido por contar con una amplia gama de aplicaciones de tipo proxy, cortafuegos y otros tipos de herramientas que permitan asegurar una red local.

Adicionalmente, cabe considerar el bajo coste de este sistema operativo en cuanto tema de licencias, pudiendo ser adoptado fácilmente por cualquier organización sin importar el tamaño de la misma.

b) Lenguaje de programación y la IDE a utilizar para el desarrollo del proyecto

El lenguaje de programación que se utilizará durante la construcción del proyecto será el lenguaje PHP; asimismo, el entorno de desarrollo integrado elegido es Quanta Plus.

La sustentación tanto del uso de este lenguaje, como de la herramienta, se hallan en el punto 2.3 del presente documento.

c) Librerías a utilizar

Para la elaboración de la solución se harán uso de una serie de librerías elaboradas por terceras personas.

Librerías PHP

La tabla 4.1 resume las librerías PHP que se utilizarán para la implementación de la solución.

Librería	Descripción	Funcionalidad a la que apoyan
php - session	Librería que permite a las aplicaciones PHP administrar sesiones.	Esta librería apoya al manejo de la seguridad y a la navegabilidad dentro de la aplicación. Permite a los usuarios almacenar información dentro de sus sesiones, la cual puede ser utilizada para verificar si efectivamente el usuario se ha logueado o si cuanta con los permisos suficientes para realizar alguna operación.
php - simplexml	Librería que permite leer y	Esta librería apoya a la lectura y

Librería	Descripción	Funcionalidad a la que apoyan
	escribir archivos xml.	almacenamiento de las configuraciones y reglas que se generen a través de la aplicación.
php - hash	Librería que provee una serie de funciones hash que pueden ser utilizadas en aplicaciones PHP.	Esta librería apoya al cifrado de información desde el código php.
php - pdo	Libraría que permite establecer una conexión con una base de datos.	Esta librería apoya a la persistencia de los objetos en la base de datos Sqlite 3 que la aplicación utilizará.

Tabla 4.1: Librerías PHP que la aplicación utilizará

Librerías Javascript

La tabla 4.2 muestra el listado de librerías Javascript que serán utilizadas para la implementación de la solución propuesta en el presente documento.

Librería	Descripción	Funcionalidad a la que apoyan
jssha	Librería javascript que provee una serie de algoritmos SHA para el cifrado de información.	Esta librería apoya al inicio de sesiones, ya que cifra al contraseña del usuario antes de que ésta sea enviada por al red.
SDMenu	Librerías que permiten trabajar con un menú desplegable escrito en javascript.	Esta librería ayuda a la navegabilidad de la aplicación.
domtab	Librería javascript que brinda la funcionalidad de pestañas.	Esta librería ayuda a la navegabilidad de la aplicación.

Tabla 4.2: Librerías Javascript que la aplicación utilizará

d) Estándares de programación

Con el objetivo de poder establecer un adecuado orden al momento de codificar la aplicación y a la vez garantizar que nuestro código sea legible para cualquier otra persona que desee hacer uso del mismo; es necesario definir una serie de estándares para la programación de la aplicación.

La lista de estándares definidos, puede ser revisada en el Anexo.

4.2. Pruebas

Las pruebas son un proceso importante dentro del desarrollo de una solución, ya que permiten detectar errores para su corrección.

Las pruebas a desarrollar serán ejecutadas a lo largo de todo el proceso de implementación de cada uno de los módulos. Éstas deberán de ser llevadas a cabo por el equipo desarrollador previamente a declarar como finalizada una nueva funcionalidad que esté siendo elaborada para la solución.

a) Tipos de prueba

Se llevarán a cabo los siguientes tipos de prueba:

Pruebas de integración: También conocidas como pruebas de Caso de Uso, Son las pruebas que buscan validar el cumplimiento de los flujos presentes en cada caso de uso.

Caso de Prueba	Caso de Uso Asociado	Objetivo de la Prueba
PI001	CUC01	Verificar se puedan agregar, modificar y eliminar usuarios del sistema.
PI002	CUC02	Verificar que el sistema permita iniciar y detener servicios.
PI003	CUC03	Verificar que el sistema permita listar módulos.
PI004	CUC04	Verificar que el sistema permita mantener un listado de zonas y nodos dentro de la red.
PI005	CUC05	Verificar que se pueda iniciar sesión utilizando únicamente la combinación adecuada de usuario y contraseña.
PI006	CUC06	Verificar la generación de los reportes de sesión.
PI007	CUC07	Verificar que únicamente se pueda

Caso de Prueba	Caso de Uso Asociado	Objetivo de la Prueba
		iniciar sesión desde equipos cuya IP ha sido autorizada.
PI008	CUC08	Verificar que se pueda habilitar / deshabilitar el reenvío de paquetes dentro de la red.
PI009	CUI01	Verificar que el sistema permita efectuar configuraciones dentro del módulo Iptables.
PI010	CUI02	Verificar el sistema permita registrar y eliminar reglas del Iptables.
PI011	CUI03	Verificar que la aplicación permita mantener el histórico de reglas y configuraciones del módulo Iptables.
PI012	CUI04	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Iptables.
PI013	CUQ01	Verificar que el sistema permita efectuar configuraciones dentro del módulo Squid.
PI014	CUQ02	Verificar el sistema permita registrar, modificar y eliminar reglas del Squid.
PI015	CUQ03	La aplicación permitirá mantener el histórico de reglas y configuraciones del módulo Squid.
PI016	CUQ04	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Squid.
PI017	CUS01	Verificar que el sistema permita efectuar configuraciones dentro del módulo Snort.
PI018	CUS02	Verificar que el sistema permita registrar, modificar y eliminar reglas del Snort.
PI019	CUS03	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Snort.

Tabla 4.3: Pruebas de integración

Pruebas de sistema: Son pruebas que buscan evaluar el desempeño funcional y tecnológico del sistema. Las pruebas de sistema buscarán

analizar aspectos de desempeño, uso de recursos, seguridad y estado del sistema tras realizarse la instalación y desinstalación del mismo.

Caso de Prueba	Aspecto a Evaluar	Objetivo de la Prueba
PS001	Prueba Funcional	Verificar que las llamadas al sistema por parte de la aplicación, sean ejecutadas.
PS002	Prueba de Desempeño	Verificar que la aplicación presente un bajo consumo de recursos por parte de memoria y procesamiento.
PS003	Prueba de Desempeño	Verificar que la página demore en ser visualizada, un tiempo menor a 5 segundos.
PS004	Prueba de Instalación	Verificar la instalación exitosa de la aplicación.

Tabla 4.4: Pruebas de sistema

Para poder revisar el catálogo de pruebas detallado, revisar el documento Anexo.

b) Técnica utilizada

Para la elaboración de las pruebas se hará uso de las siguientes técnicas:

- **Prueba de caja negra:** Consiste en definir los valores para cada uno de los campos de ingreso de datos del sistema, con el objetivo de determinar si la salida obtenida es equivalente al resultado esperado.
- **Prueba de valor extremo:** Consiste en utilizar valores tope, máximos y mínimos, como parámetros de entrada del sistema, con el objetivo de determinar su adecuado funcionamiento para dichos casos.

c) Resultados de las pruebas

Tras efectuarse cada una de las pruebas listadas, se obtuvieron los siguientes resultados.

Resultados de las pruebas de integración

Caso de Prueba	Resultado	Observaciones
PI001	Éxito	-
PI002	Éxito	-
PI003	Éxito	-
PI004	Éxito	-
PI005	Éxito	-
PI006	Éxito	-
PI007	Éxito	-
PI008	Éxito	-
PI009	Éxito	-
PI010	Éxito	-
PI011	Éxito	-
PI012	Éxito	-
PI013	Éxito	-
PI014	Éxito	-
PI015	Éxito	-
PI016	Éxito	-
PI017	Éxito	-
PI018	Éxito	-
PI019	Éxito	-

Tabla 4.5: Resultado de las pruebas de integración

Resultados de las pruebas de sistema

Caso de Prueba	Resultado	Observaciones
PS001	Éxito	Para que las llamadas a procesos de administración pudieran ser efectuados, fue necesario editar el archivo sudoers.
PS002	Éxito	Las pruebas fueron realizadas limitando el consumo de memoria a 32 MB para la ejecución de los scripts PHP.
PS003	Éxito	La prueba fue realizada deshabilitando servicios no críticos para el sistema.
PS004	Éxito	-

Tabla 4.6: Resultado de las pruebas de sistema

Capítulo 5: Observaciones, Conclusiones y Recomendaciones

En este capítulo se darán a conocer las observaciones, conclusiones y recomendaciones que han surgido a consecuencia de la implantación de la presente solución.

5.1. Observaciones

La aplicación implantada, ha presentado un adecuado funcionamiento sobre redes cableadas que trabajan sobre protocolo IPv4; sin embargo, las herramientas que son administradas por esta solución aún carecen de soporte para el protocolo IPv6 y redes inalámbricas.

Para poder ejecutar llamadas a procesos y funciones administrativas (a nivel del Sistema Operativo) ha sido necesario editar el archivo de configuración sudoers. La configuración de este archivo no puede ser automatizada ya que contiene información altamente sensible.

5.2. Conclusiones

Luego de llevar a cabo la implantación y pruebas de la solución propuesta, se concluye lo siguiente:

- Mientras más sencilla y fácil de utilizar sea una aplicación para los usuarios, los riesgos de llevar a cabo una inadecuada configuración son menores; asimismo, el tiempo invertido en llevar a cabo las configuraciones es menor, lo cual permite asignar dicho personal a tareas críticas.
- El riesgo que muchos usuarios carezcan de conocimientos en cuanto a reglas de seguridad, puede ser aminorado haciendo uso de una solución que le ayude a establecer reglas y configuraciones iniciales claras y fáciles de comprender.
- A pesar de las soluciones que permiten configurar y asegurar las redes y sistemas; siempre existirá un nivel de riesgo a ataques, mientras los usuarios no conciencien sobre los riesgos a los que se hallan expuestos.

5.3. Recomendaciones

La arquitectura utilizada para la implementación de la aplicación permite a los usuarios personalizar, agregar y remover funcionalidades, es por ello que pueden darse varios cambios y mejoras en torno al módulo central de la solución.

Algunas recomendaciones en cuanto dichas mejoras serían las siguientes:

- Agregar soporte para el protocolo IPv6, en futuras versiones de las herramientas Squid y Snort.
- Agregar soporte para conexiones a través de dispositivos inalámbricos.
- Flexibilizar más la generación de reglas para las diversas herramientas. En el presente proyecto se han considerado los elementos necesarios para elaborar configuraciones básicas que aseguren mejor la red frente a los ataques; sin embargo, es posible incluir nuevas opciones como por ejemplo, control por horas y días.

- Agregar a la aplicación un soporte multilenguaje.

Algunos cambios que podrían hacerse sobre la misma aplicación, o sobre la idea en la que la aplicación está basada:

- Diseñar e Implementar un módulo que permita gestionar y administrar la configuración del servidor que permita administrar los protocolos DNS y DHCP en una red IPv6.

Bibliografía

Definición del Problema

- [WWW0001] Computer Security Institute. http://www.gocsi.com/
- [CIS2007] CISCO. CISCO Annual security Report 2007. 2007
- [WWW0002] SANS Institute. http://www.sans.org/top20/?portal=7864dc3b7f1cd9b3202c2494164f574c

Marco Conceptual

- [HUN2002] HUNT, Craig. Linux Network Servers. SYBEX. 2002. ISBN: 0-7821-4123-4
- [COL2005] COLE, Erick. Network Security Bible. Wiley Publishing Inc. 2005. ISBN: 0-7645-7397-7
- [NEG2004] NEGUS, Christopher, WEEKS, Thomas. Linux Troubleshooting Bible. Wiley Publishing Inc. 2004. ISBN: 076456997X
- [WWW0003] Netfilter. http://netfilter.org
- [WWW0004] Snort. http://www.snort.org
- [WWW0005] Squid. http://www.squid-cache.org/

Plan de Proyecto

- [PMB2004] PMBOK 2004.
- [SOF2008] SOFTHOUSE Consulting. Scrum in five minutes.

Crystal Clear

- [COC2004] COCKBURN, Alistair. Crystal Clear A Human-Powered Methodology for Small Teams. Addison Wesley Professional. 2004. ISBN: 0-201-69947-8

Estado del Arte

- [WWW0006] Webmin. http://www.webmin.com
- [WWW0007] Firestarter. http://www.fs-security.com/
- [WWW0008] Guardog. http://www.simonzone.com/software/guarddog/
- [WWW0009] BASE. http://base.secureideas.net/about.php

Configuración del Iptables

- [EYC2002] EYCHENNE, Herve. Linux Man Page: Iptables. 2002
- [DAW2005] BAUTTS, Tony, DAWSON, Terry, PURDY, Gregor. Linux Network Administrators Guide. 2005. ISBN: 0-596-00548-2
- [BRO2001] BROCKMEIER, Joe, LEBLANC, Dee-Ann, McCARTY, Ron. Linux Routing. New Riders Publishing. 2001. ISBN: 1-57870-267-4
- [BAR2003] BARRETT, Daniel, BYRNES, Robert, SILVERMAN, Richard. Linux Security Cookbook. O'Reilly. 2003. ISBN: 0-596-00391-9
- [SHI2004] SHINN, Michael, SHINN, Scott. Troubleshooting Linux Firewalls. Prentice Hall PTR. 2004. ISBN: 0-321-22723-9

Configuración del Squid

- [WES2004] WESSELS, Duane. Squid: The definitive Guide. O' Reilly. 2004.
  ISBN:0-596-00162-2

Configuración del Snort

- [ROE2006] ROESCH, Martin. Linux Man Page : Snort. 2006
- [SNO2006] The SNORT PROJECT. Snort Users Manual. Sourcefire. 2006

Arquitectura

- [FOW2002] FOWLER, Martin. Patterns of Enterprise Application Architecture. Addison-Wesley Professional. 2002. ISBN: 0321127420

Construcción

- [WWW0010] TIOBE. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
- [WWW0011] PHP.NET. http://www.php.net
- [ALS2005] ALSHANETSKY, Ilia. PHP|Architect's Guide to PHP Security. Nanobooks. 2005. ISBN: 0-9738621-0-6
- [SHI2005] SHIFLETT, Chris. Essential PHP Security. O'Reilly. 2005. ISBN: 0-596-00656-X
- [OWE2006] OWENS, Michael. The Definitive Guide to Sqlite. Apress. 2006. ISBN: 1-59059-673-0

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN PARA LA ADMINISTRACIÓN DE LAS HERRAMIENTAS DE SEGURIDAD EN UNA RED LOCAL

Anexos

Dennis Stephen Cohn Muroy

ASESOR: Ingeniero Corrado Daly Scaletti

Lima, octubre del 2008

INDICE

Indice de Imágenes

Índice de Tablas

1. Objetivo del Proyecto

El objetivo principal del presente proyecto de fin de carrera, es el de realizar el análisis, diseño e implementación de una aplicación para facilitar la administración de las herramientas de seguridad para una red local.

Siendo los objetivos específicos, los listados a continuación:

- Identificar los protocolos y aplicaciones más utilizadas por los usuarios para poder contar con una serie de reglas predefinidas.
- Simplificar el uso de las herramientas utilizadas para asegurar la red local.
- Disminuir los costos de entrenamiento de los usuarios que administrarán las herramientas de seguridad de la red local.
- Implementar un registro histórico de reglas y configuraciones que permita restablecer la aplicación a una configuración previa.

Las prioridades en la elaboración del proyecto son:

Sacrificar tareas por:

- Cumplir con la fecha de entrega.
- Asegurar la calidad y seguridad de la aplicación.

Mantener:

- Posibilidad de crecer en una herramienta más extensa.
- Consumo bajo de recursos.
- Usabilidad.
- Facilidad de aprendizaje.

Sacrificar estas tareas por otras:

- Contenido de imágenes.
- Calidad de las imágenes.

2. Catálogo de Actores

El presente acápite da a conocer el conjunto de actores, quienes interactuarán con el sistema y las tareas que cada uno de ellos desempeñará haciendo uso del sistema.

En la siguiente imagen se puede apreciar a los principales actores que interactuarán con la aplicación; así como la relación existente entre ellos.

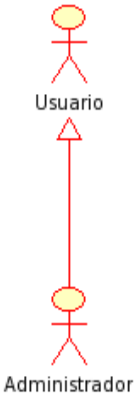


Imagen 2.1: Actores que interactúan con el sistema

Usuario:

El Usuario representa a un usuario – administrador del sistema; es aquella persona que apoya la tarea de administración; pero cuenta con permisos reducidos o restringidos. Sus principales tareas se centran en la configuración de reglas para las diversas herramientas de seguridad, asimismo puede generar los reportes que resumen la bitácora registrada por cada herramienta.

Administrador:

El administrador del sistema es único. Cuenta con todos los permisos y privilegios para efectuar cambios dentro de la configuración de las herramientas que el sistema permite administrar.

La siguiente tabla resume las principales tareas que ejecutarán los actores en la aplicación.

Actor	Tarea
Usuario	Iniciar sesión
	Mantener reglas del Snort
	Mantener reglas del Iptables
	Mantener reglas del Squid
	Actualizar contraseña
	Visualizar características del servidor
	Generar reportes de la bitácora del Snort
	Generar reportes de la bitácora del Iptables
	Generar reportes de la bitácora del Squid
Administrador	Mantener usuarios
	Mantener servicios
	Mantener módulos
	Mantener registro histórico
	Generar reporte de sesiones
	Mantener subredes y nodos
	Configurar Snort
	Configurar Iptables
	Configurar Squid

Tabla 2.1: Lista de tareas efectuadas por los actores

3. Catálogo de Requerimientos

Este acápite contiene un listado de requerimientos – de acuerdo a la documentación de la metodología que se está utilizando, este punto podría ser llamado ”Lista de Deseos”.

El presente punto busca brindar una sección en donde los requerimientos puedan ser almacenados para así evitar que sean olvidados. Asimismo, provee un contexto sobre el cual se pueden tomar decisiones.

A continuación se listará el listado de requerimientos, tanto funcionales como no funcionales, para el presente proyecto.

3.1. Requerimientos Funcionales

La lista de requerimientos funcionales se halla dividida en cuatro (4) grupos cada uno de los cuales representa cada módulo en los que el proyecto se hallará subdividido.

Dentro de la información que muestra la cabecera de cada tabla de requerimientos, encontramos los siguientes campos: Número de requerimiento, la descripción del requerimiento, la prioridad del requerimiento – que pude tomar valores del uno (menor prioridad) al cinco (mayor prioridad) – y el tipo de requerimiento – que puede ser Exigible (E) o Deseable (D).

Módulo Central	
Código	FC001
Requerimiento	La aplicación permitirá administrar el listado de los módulos del sistema.
Actor	Administrador, Usuario
Valor aportado	Permite a los administradores de la red, conocer cuáles son los módulos que se hallan instalados actualmente y si se hallan configurados de forma adecuada.

Prioridad	4
Tipo	E

Código	FC002
Requerimiento	La aplicación permitirá asignar un alias a las diferentes interfaces detectadas en el equipo.
Actor	Administrador
Valor aportado	De esta forma es posible identificar fácilmente la utilidad de cada interfaz.
Prioridad	4
Tipo	E

Código	FC003
Requerimiento	La aplicación permitirá activar y desactivar la opción de reenvío de paquetes.
Actor	Administrador
Valor aportado	Permite habilitar el ruteo de datos entre las redes que el servidor (en donde está instalada la aplicación) interconecta.
Prioridad	3
Tipo	E

Código	FC004
Requerimiento	La aplicación permitirá exportar e importar las configuraciones y reglas establecidas por cada módulo del sistema.
Actor	Administrador, Usuario
Valor aportado	Ahorro de tiempo en caso se desee configurar de seguridad de la red desde cualquier nodo de la red.
Prioridad	4
Tipo	E

Código	FC005
Requerimiento	La aplicación permitirá mantener la lista de usuarios que podrán iniciar sesión remota en el sistema.
Actor	Administrador
Valor aportado	Seguridad, permitiendo que únicamente los usuarios autorizados hagan uso de la aplicación.
Prioridad	3
Tipo	E

Código	FC006
Requerimiento	La aplicación permitirá administrar una lista de direcciones IP hábiles que podrán iniciar sesión remota en el sistema.

Actor	Administrador
Valor aportado	Seguridad, se disminuye el riesgo por ataques que se centren en romper contraseñas, ya que los intentos de acceso se filtran por la dirección IP de los equipos.
Prioridad	4
Tipo	E

Código	FC007
Requerimiento	La aplicación permitirá manejar una bitácora que permita auditar las sesiones iniciadas, almacenando información como usuario, IP de origen, sistema operativo y hora.
Actor	Aplicación
Valor aportado	Seguridad, de esta forma es posible determinas si es que se está llevando a cabo un ataque para obtener los usuarios y contraseñas.
Prioridad	4
Tipo	E

Código	FC008
Requerimiento	La aplicación permitirá iniciar y detener los servicios relacionados con cada uno de los módulos activados.
Actor	Administrador
Valor aportado	Permite a los administradores de la red, habilitar únicamente los servicios que desee estén funcionales.
Prioridad	3
Tipo	E

Código	FC009
Requerimiento	La aplicación permitirá mantener la lista de los nodos y las zonas existentes en la red
Actor	Administrador
Valor aportado	Permite a los administradores registrar los nodos de red que están conectados a cada interfaz.
Prioridad	4
Tipo	E

Módulo Iptables	
Código	FI001
Requerimiento	La aplicación permitirá configurar las opciones básicas del Iptables.
Actor	Administrador
Valor aportado	Permite al administrador llevar a cabo configuraciones del módulo

	Iptables.
Prioridad	5
Tipo	E

Código	FI002
Requerimiento	La aplicación podrá trabajar con una serie de reglas predeterminadas para determinados programas.
Actor	Administrador, Usuario
Valor aportado	El administrador del sistema ahorrará tiempo en la generación de reglas comunes, pudiéndose centrar en aquellas reglas que demanden mayor exactitud.
Prioridad	5
Tipo	E

Código	FI003
Requerimiento	La aplicación permitirá administrar la tabla nat y la tabla filter.
Actor	Aplicación
Valor aportado	El módulo brindará soporte a las funciones de filtrado de paquetes y de nateo.
Prioridad	5
Tipo	E

Código	FI004
Requerimiento	La aplicación permitirá mantener las reglas de Iptables.
Actor	Administrador, Usuario
Valor aportado	Los administradores de la red, podrán registrar las reglas que la herramienta ha de ejecutar.
Prioridad	5
Tipo	E

Código	FI005
Requerimiento	La aplicación permitirá establecer políticas para cada cadena registrada en Iptables.
Actor	Administrador, Usuario
Valor aportado	Los administradores de la red podrán registrar políticas, reglas a ejecutarse en caso la aplicación no halle coincidencia alguna con ninguna de las reglas registradas.
Prioridad	4
Tipo	E

Código	FI006
Requerimiento	Las reglas establecidas por la aplicación podrán ser aplicables a

	un rango de direcciones IP.
Actor	Administrador, Usuario
Valor aportado	Las reglas establecidas podrán ser aplicadas sobre un rango de direcciones IP.
Prioridad	2
Tipo	E

Código	FI007
Requerimiento	La aplicación podrá registrar reglas de filtrado por MAC
Actor	Administrador, Usuario
Valor aportado	Las reglas establecidas podrán ser aplicadas sobre direcciones MAC.
Prioridad	2
Tipo	E

Código	FI008
Requerimiento	La aplicación permitirá administrar el registro histórico de reglas y configuraciones generadas por el módulo Iptables
Actor	Administrador, Usuario
Valor aportado	Brinda un ahorro de tiempo en caso se necesite retomar una configuración que ha sido usado con anterioridad.
Prioridad	5
Tipo	E

Código	FI009
Requerimiento	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Iptables.
Actor	Administrador, Usuario
Valor aportado	Los administradores podrán generar reportes basados en la bitácora registrada por el sistema, para así determinar los posibles intentos por vulnerabilizar la red.
Prioridad	3
Tipo	E

Módulo Squid	
Código	FQ001
Requerimiento	La aplicación permitirá configurar las opciones básicas del Squid.
Actor	Administrador
Valor aportado	Permite al administrador llevar a cabo configuraciones del módulo Squid.
Prioridad	5

Tipo	E

Código	FQ002
Requerimiento	La aplicación podrá trabajar con una serie de reglas preestablecidas para determinados programas.
Actor	Administrador, Usuario
Valor aportado	El administrador del sistema ahorrará tiempo en la generación de reglas comunes, pudiéndose centrar en aquellas reglas que demanden mayor exactitud.
Prioridad	5
Tipo	E

Código	FQ003
Requerimiento	La aplicación permitirá habilitar y configurar las funcionalidades básicas de la caché y del proxy.
Actor	Administrador
Valor aportado	Permite al administrador configurar los parámetros que regulan el comportamiento de las opciones de proxy y cache del Squid.
Prioridad	4
Tipo	E

Código	FQ004
Requerimiento	La aplicación permitirá registrar una lista de control de acceso.
Actor	Administrador, Usuario
Valor aportado	La aplicación permitirá registrar la lista de direcciones IP que serán filtradas por la herramienta.
Prioridad	5
Tipo	E

Código	FQ005
Requerimiento	La aplicación permitirá registrar grupos de control de acceso.
Actor	Administrador, Usuario
Valor aportado	La aplicación permitirá agrupar la lista de direcciones IP que serán filtradas por la herramienta.
Prioridad	5
Tipo	E

Código	FQ006
Requerimiento	La aplicación permitirá administrar reglas de control de acceso a través de las cuales se indicarán las listas y grupos de control a quienes se les permitirá acceder o se les negará el acceso a determinados recursos.

Actor	Administrador, Usuario
Valor aportado	La aplicación permitirá asignar una serie de reglas de filtrado sobre los grupos y listas de control de acceso registrados.
Prioridad	5
Tipo	E

Código	FQ007
Requerimiento	La aplicación permitirá administrar el registro histórico de reglas y configuraciones generadas por el módulo Squid.
Actor	Administrador, Usuario
Valor aportado	Brinda un ahorro de tiempo en caso se necesite retomar una configuración que ha sido usado con anterioridad.
Prioridad	4
Tipo	E

Código	FQ008
Requerimiento	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Squid.
Actor	Administrador, Usuario
Valor aportado	Los administradores podrán generar reportes basados en la bitácora registrada por el sistema, para así determinar los posibles intentos por vulnerabilizar la red.
Prioridad	3
Tipo	E

Módulo Snort	
Código	FS001
Requerimiento	La aplicación permitirá configurar las opciones básicas del Snort para que trabaje en modo NIDS
Actor	Administrador
Valor aportado	Permite al administrador llevar a cabo configuraciones del módulo Snort para que así desempeñe las funciones de un Sistema de Detección de Intrusos basado en red.
Prioridad	5
Tipo	E

Código	FS002
Requerimiento	La aplicación permite definir la lista de preprocesadores que Snort utilizará.
Actor	Administrador
Valor aportado	El administrador podrá definir el preprocesador que la aplicación deberá de utilizar.

Prioridad	3
Tipo	E

Código	FS003
Requerimiento	El módulo trabajará en forma predeterminada con las reglas proporcionadas por la comunidad de usuario de Snort, debidamente validadas por *Sourcefire*, empresa a cargo de la implementación y distribución de Snort.
Actor	Administrador, Usuario
Valor aportado	El administrador del sistema ahorrará tiempo en la generación de reglas comunes, pudiéndose centrar en aquellas reglas que demanden mayor exactitud.
Prioridad	5
Tipo	E

Código	FS004
Requerimiento	La aplicación generará reportes basándose en las bitácoras registradas por el módulo Snort.
Actor	Administrador, Usuario
Valor aportado	Los administradores podrán generar reportes basados en la bitácora registrada por el sistema, para así determinar los posibles intentos por vulnerabilizar la red.
Prioridad	3
Tipo	E

3.2. Requerimientos no funcionales

Requisitos no funcionales	
Código	FN001
Requerimiento	La aplicación permitirá acceso remoto vía web.
Valor aportado	Permite contar con una interfaz gráfica para llevar a cabo las configuraciones de forma remota.
Tipo	E

Código	FN002
Requerimiento	La versión en desarrollo de la aplicación soportará interfaces ethernet.
Valor aportado	Las herramientas que se administrarán presentan un adecuado soporte para trabajar con interfaces ethernet.
Tipo	E

Código	FN003
Requerimiento	La aplicación hará uso del protocolo IPv4.
Valor aportado	En la actualidad, a nivel nacional, es el protocolo que se está utilizando para establecer una comunicación entre dos máquinas.
Tipo	E

Código	FN004
Requerimiento	La aplicación presentará una interfaz que sea fácil de utilizar.
Valor aportado	Rápida configuración de las herramientas, disminuyendo los riesgos producidos por configuraciones ineficientes de las herramientas.
Tipo	E

Código	FN005
Requerimiento	La aplicación estará escrita en el lenguaje PHP 5.
Valor aportado	Lenguaje bastante ligero y que puede ser ejecutado sobre Sistemas Operativos GNU/Linux, sobre el que se ejecutan las herramientas a configurar.
Tipo	E

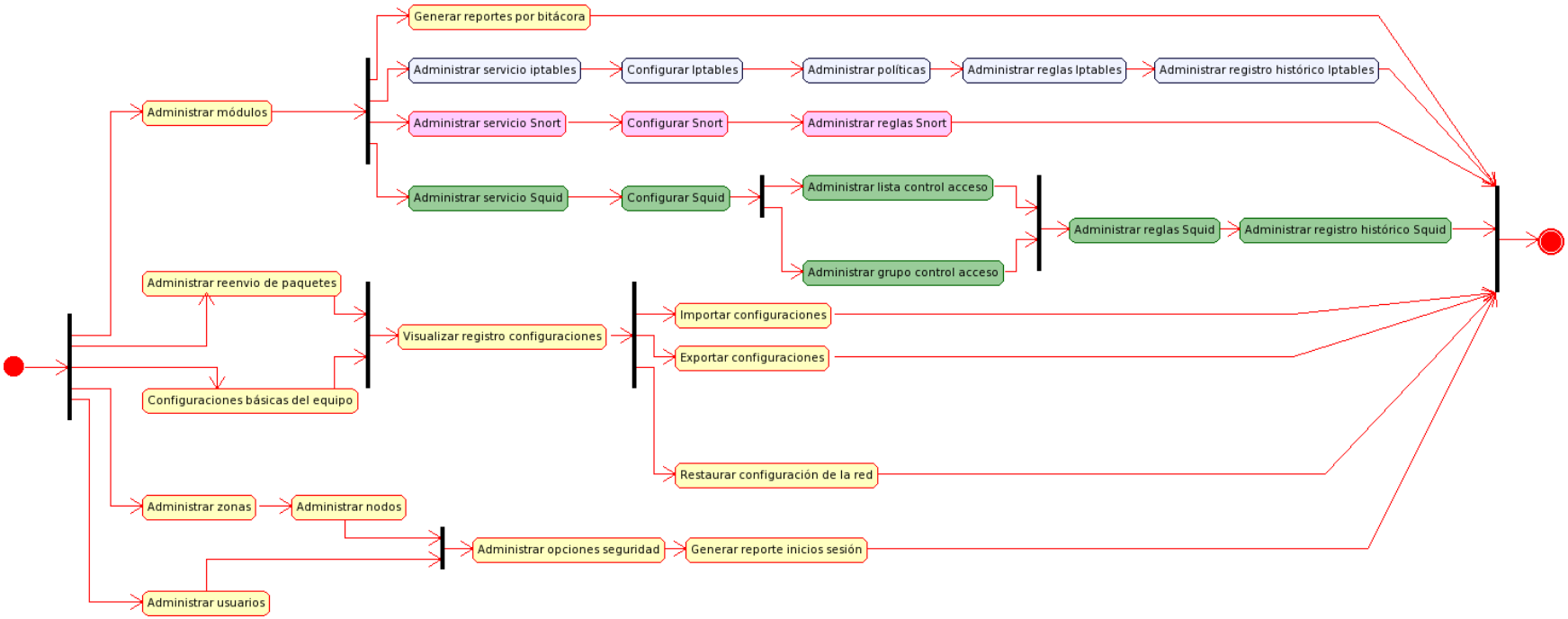
Código	FN006
Requerimiento	El instalador de la aplicación estará escrito en el lenguaje Perl.
Valor aportado	Lenguaje script bastante popular, ligero y robusto.
Tipo	E

Código	FN007
Requerimiento	La aplicación se ejecutará sobre el servidor web Apache versión 2.
Valor aportado	La versión actual del servidor web, comúnmente utilizado para trabajar con aplicaciones php.
Tipo	E

Código	FN008
Requerimiento	La aplicación se ejecutará prioritariamente en sistemas operativos GNU/Linux basados en RedHat.
Valor aportado	Las distribuciones basadas en Redhat tienen bastante tiempo en el mercado y han demostrado ser bastante robustas y seguras.
Tipo	E

4. Mapa del Proyecto

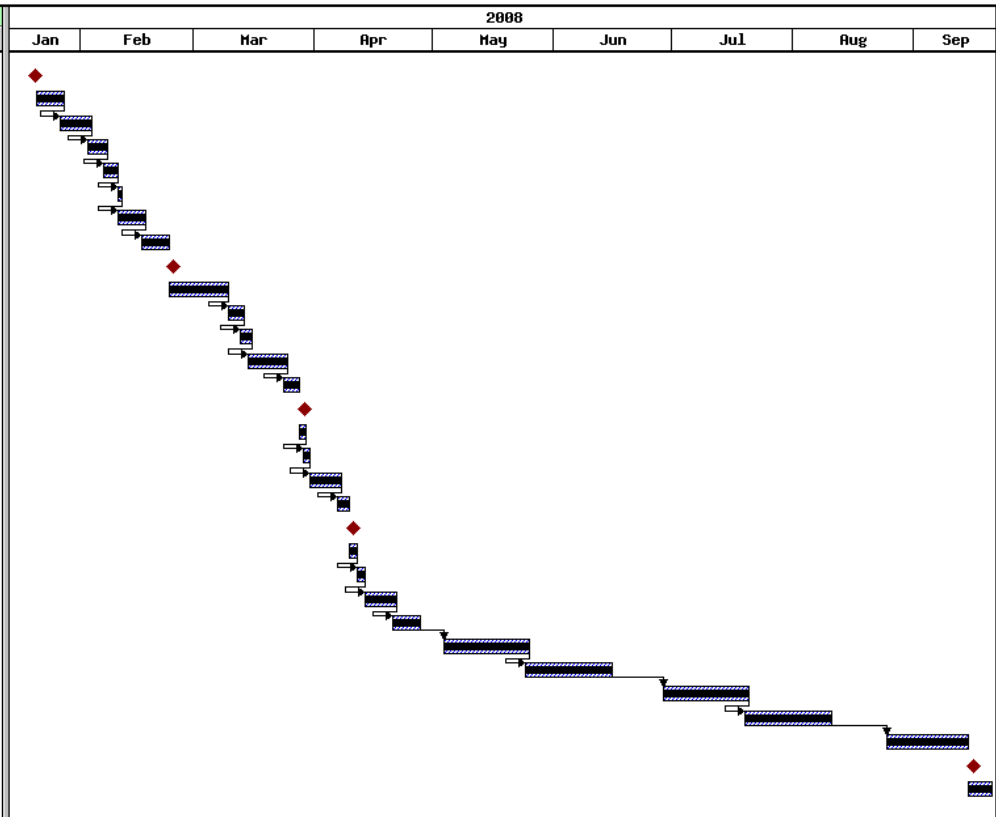
El mapa del proyecto es un diagrama en donde se da a conocer las tareas que deben de desarrollarse; así como el orden de las mismas. Para así definir la secuencia en que las tareas serán resueltas.



5. Diagrama de Gant

Proyecto de Tesis

Task name	Dur.	Start	Finish
Capítulo1: Generalidades		20/01/2008	20/01/2008
Definición del Problema	35 h	21/01/2008	27/01/2008
Marco Conceptual	31 h	27/01/2008	03/02/2008
Plan de Proyecto	19 h	03/02/2008	07/02/2008
Estado del Arte	19 h	07/02/2008	10/02/2008
Sustentación de la Solución	5 h	11/02/2008	11/02/2008
Revisión Capítulo 1	42 h	11/02/2008	17/02/2008
Corrección Capítulo 1	41 h	17/02/2008	23/02/2008
Capítulo 2: Análisis		24/02/2008	24/02/2008
Definición de la metodología	61 h	24/02/2008	09/03/2008
Identifiación de requerimientos	20 h	10/03/2008	13/03/2008
Análisis de la solución	11 h	13/03/2008	15/03/2008
Revisión Capítulo 2	40 h	15/03/2008	24/03/2008
Corrección Capítulo 2	20 h	24/03/2008	27/03/2008
Capítulo 3: Diseño		28/03/2008	28/03/2008
Arquitectura de la solución	9 h	28/03/2008	29/03/2008
Elaboración de pantallas	5 h	29/03/2008	30/03/2008
Revisión Capítulo 3	35 h	31/03/2008	07/04/2008
Corrección Capítulo 3	15 h	07/04/2008	09/04/2008
Capítulo 4: Construcción		10/04/2008	10/04/2008
Definición de la Construcción	9 h	10/04/2008	11/04/2008
Definición de las pruebas	4 h	12/04/2008	13/04/2008
Revisión Capítulo 4	35 h	14/04/2008	21/04/2008
Corrección Capítulo 4	30 h	21/04/2008	27/04/2008
Sprint 1	91 h	04/05/2008	25/05/2008
Sprint 2	91 h	25/05/2008	15/06/2008
Sprint 3	91 h	29/06/2008	20/07/2008
Sprint 4	91 h	20/07/2008	10/08/2008
Sprint 5	91 h	25/08/2008	14/09/2008
Capítulo 5: Conclusiones		15/09/2008	15/09/2008
Conclusiones y Sugerencias	26 h	15/09/2008	20/09/2008



6. Plan de Lanzamiento

El plan de lanzamiento resume las principales fechas de entrega de cada uno de las tareas definidas.

Tarea	Fecha Inicio	Fecha Fin
Sprint 1		
Administrar usuarios	04/05/2008	17/05/2008
Administrar configuraciones	17/05/2008	24/05/2008
Visualizar configuraciones	17/05/2008	24/05/2008
Generar reportes de sesiones	24/05/2008	25/05/2008
Sprint 2		
Administrar módulos	25/05/2008	01/05/2008
Administrar servicios	01/05/2008	08/05/2008
Administrar zonas	07/05/2008	15/06/2008
Sprint 3		
Administrar configuración Iptables	29/06/2008	06/07/2008
Administrar reglas Iptables	06/07/2008	12/07/2008
Administrar registro histórico de Iptables	12/07/2008	18/07/2008
Generar reporte Iptables	18/07/2008	20/07/2008
Sprint 4		
Administrar configuración Squid	20/07/2008	25/07/2008
Administrar reglas Squid	25/07/2008	01/08/2008
Administrar registro histórico de Squid	01/08/2008	08/08/2008
Generar reporte Squid	08/08/2008	10/08/2008
Sprint 5		
Administrar configuración Snort	25/08/2008	02/09/2008
Administrar reglas Snort	02/09/2008	07/09/2008
Generar reporte Snort	07/09/2008	14/09/2008

Tabla 6.1: Plan de Lanzamiento

7. Catálogo de Riesgos

El siguiente cuadro presenta un listado de los principales riesgos a los que el proyecto se puede enfrentar a lo largo de todo su ciclo de vida; así como las acciones a efectuar para mermar el impacto producido en caso el riesgo se llegara a dar.

N°	Riesgo	%	Impacto	Respuesta
1	La planificación no incluye tareas necesarias.	20%	Mediano	Se programara reuniones con los miembros del grupo para dar a conocer las tareas y hallar las faltantes, las cuales se agregarán en el backlog para el siguiente Sprint
3	Un retraso en una tarea produce retrasos en cascada en las tareas dependientes.	20%	Mediano	Las tareas fatantes formarán parte del backlog del siguiente Sprint.
4	Los espacios no están disponibles en el momento necesario.	40%	Bajo	Hacer uso de equipos propios (portátiles)
5	Los espacios están disponibles pero no cuentan con las herramientas o configuraciones necesarias.	60%	Mediano	Hacer uso de recursos remotos para las configuraciones faltantes.
6	La curva de aprendizaje para la herramienta a configurar es más larga de lo esperado.	30%	Alto	Revisión de manuales para ayuda. Consulta con expertos del tema.
7	El dueño del proyecto pide nuevos requisitos.	10%	Bajo	Los nuevos requisitos se agregarán al backlog del siguiente Sprint.
8	Los ciclos de revisión/decisión del dueño del proyecto para los prototipos y funcionalidades son más lentos de lo esperado.	40%	Alto	Pruebas individuales rigurosas para minimizar la cantidad de errores en los productos entregados. Agregar las correcciones solicitadas al backlog del siguiente Sprint.
9	Pérdida de información por daño de equipos o robo	15%	Muy Alto	Mantener copias remotas en un servidor.

Tabla 7.1: Catálogo de Riesgos

8. Especificación de casos de uso

En esta sección se describe a manera general las funcionalidades de la aplicación. Se presentará el modelo de casos de uso (modelo que muestra las funcionalidades del sistema), supuestos y dependencias.

8.1. Módulo Central

Este paquete contiene los casos que corresponden a la seguridad del sistema, la validación de los usuarios y la administración de los permisos de los mismos de manera que se implemente un esquema de privilegios para los usuarios y mantener segura la información.

Los casos de uso incluidos en este paquete son: Validar Usuario, Mantener Usuarios y Mantener Perfiles.

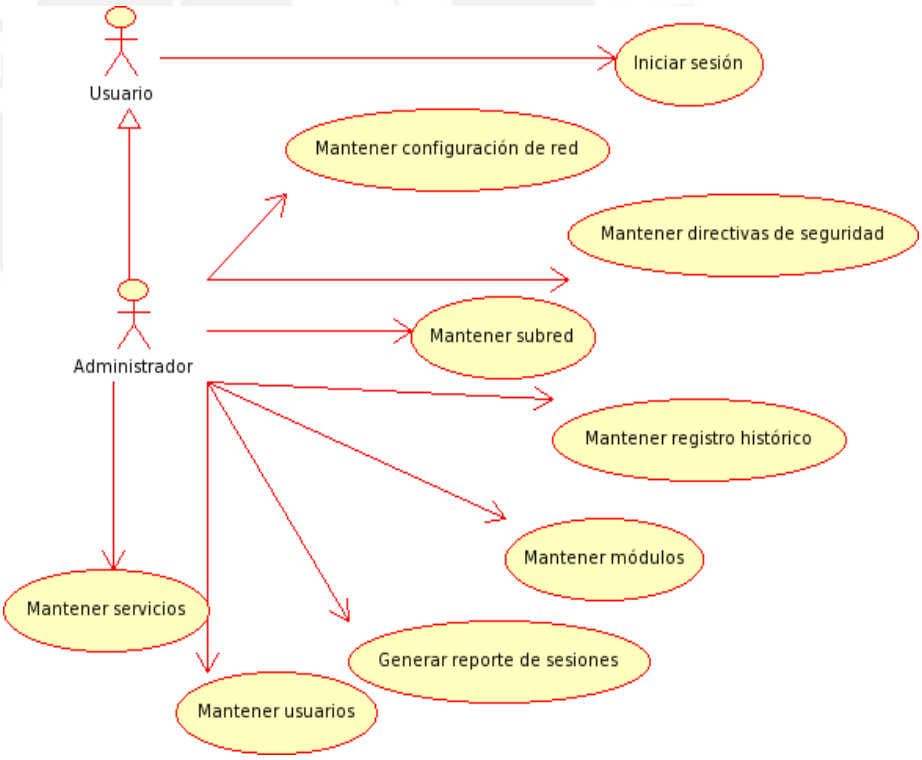


Imagen 8.1: Diagrama de Casos de Uso - Módulo Central

Módulo Central		
Código	**Caso de uso**	**Requerimientos Asociados**
CUC01	Mantener usuarios	FC005
CUC02	Mantener servicios	FC008
CUC03	Mantener módulos	FC001, FC004
CUC04	Mantener subred	FC009
CUC05	Iniciar sesión	FC005, FC006
CUC06	Generar reporte de sesiones	FC007
CUC07	Administrar directivas de seguridad	FC006
CUC08	Mantener configuración de red	FC002, FC003, FC009

Tabla 8.1: Casos de Uso - Módulo Central

a) Iniciar sesión

Caso de Uso: Iniciar Sesión	
Descripción:	Permite a un actor iniciar sesión en la aplicación.
Actores:	Administrador, Usuario
Precondición:	Ninguna.
Flujo Principal: Iniciar Sesión	
1. El actor ingresa en la pantalla de inicio de sesión su nombre de usuario y contraseña. 2. La aplicación verifica que los datos ingresados sean válidos. Si son correctos muestra al usuario la pantalla principal de la aplicación. 3. El caso de uso finaliza.	
Postcondición:	El usuario ha iniciado sesión satisfactoriamente.
Flujo Excepcional: Datos incorrectos	
1. Si el usuario y/o la contraseña ingresados son incorrectos, la aplicación mostrará el mensaje de error: “El usuario y/o la contraseña son incorrectos”	

b) Mantener usuarios

Caso de Uso: Mantener Usuarios	
Descripción:	Permite registrar un nuevo usuario en el sistema. Así mismo permite evitar replicaciones de usuarios, modificar y eliminar a un usuario registrado en el sistema.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Registrar Usuario	
1. El actor selecciona la opción “Nuevo Usuario”.	

Caso de Uso: Mantener Usuarios	
2. La aplicación muestra un formulario para que el actor registre los datos del nuevo usuario. 3. El actor llena los campos obligatorios del formulario (Nombre, Contraseña, Confirmar Contraseña). 4. La aplicación verifica que no halla un Nombre de usuario ya existente y que los cuadros de Contraseña y Confirmar Contraseña sean idénticos y no vacíos. Luego de ello registra los datos en la Base de Datos y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente los datos de un nuevo usuario.
Flujo Alternativo: Modificar Usuario	
1. El actor selecciona la opción “Editar Usuario”. 2. La aplicación muestra un formulario para que el actor edite los datos del usuario. 3. El actor actualiza el valor de los campos. 4. La aplicación verifica que no halla un Nombre de usuario ya existente y que los cuadros de Contraseña y Confirmar Contraseña sean idénticos y no vacíos. Luego de ello actualiza los datos en la Base de Datos y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Flujo Alternativo: Eliminar Usuario	
1. El actor selecciona la opción “Eliminar Usuario”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación del usuario.. 4. El sistema elimina el usuario y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Flujo Excepcional: Datos incorrectos	
1. Si el nombre de usuario ya existe o si los valores registrados en los cuadros de Contraseña y Confirmar Contraseña no coinciden o son vacíos, la aplicación mostrará un mensaje de error al usuario.	

c) Mantener servicios

Caso de Uso: Mantener Servicios	
Descripción:	Permite iniciar y detener los servicios relacionados con cada uno de los módulos detectados.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Actualizar Servicio	
1. El actor selecciona la opción “Administrar Servicios”.	

Caso de Uso: Mantener Servicios	
2. La aplicación muestra una tabla en donde están listados los servicios relacionados a cada módulo de la aplicación, así como el estado de cada servicio (iniciado / detenido). 3. El actor selecciona la opción Iniciar / Detener servicio del servicio cuyo estado desea modificar. 4. La aplicación inicia / detiene el servicio, mostrando el mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha modificado satisfactoriamente el estado del servicio.

d) Mantener módulos

Caso de Uso: Mantener Módulos	
Descripción:	Permite administrar el listado de los módulos del sistema, habilitándolos o deshabilitándolos.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Actualizar Módulo	
1. El actor selecciona la opción “Administrar Módulo”. 2. La aplicación muestra una tabla en donde están listados los módulos detectados por la aplicación. 3. El caso de uso finaliza.	
Postcondición:	Se ha mostrado satisfactoriamente el estado de cada módulo.

e) Mantener subred

Caso de Uso: Mantener subred	
Descripción:	Permite agregar un nuevo nodo a una interfaz detectada por el sistema.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Registrar Subred	
1. El actor selecciona la opción “Administrar Zonas”. 2. La aplicación muestra la lista de interfaces detectadas con sus respectivas subredes. 3. El actor selecciona la opción “Nueva Subred”. 4. La aplicación muestra un formulario para registrar la nueva subred. 5. El actor registra los datos de la nueva subred (Nombre de la Subred, IP, Máscara, Máscara Corta, Interfaz). 6. La aplicación verifica los datos ingresados. Si son correctos guarda los datos de la subred y muestra un mensaje de éxito y una tabla a través de	

Caso de Uso: Mantener subred	
la cual el usuario puede registrar los nodos de la red. 7. El usuario ingresa la dirección IP del nodo y el nombre del mismo; luego pulsa el botón agregar. 8. La aplicación registra los valores del nuevo nodo. 9. Se repite el punto 7 y 8 hasta que el usuario pulse sobre el botón “Salir”. 10. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente los datos de un nuevo nodo.
Flujo Alternativo: Modificar subred	
1. El actor selecciona la opción “Editar Subred”. 2. La aplicación muestra un formulario para que el actor edite los datos de la subred. 3. El actor actualiza el valor de los campos (Nombre de la Subred, IP, Máscara, Máscara Corta, Interfaz). 4. El sistema verifica los datos ingresados. Si son correctos actualiza los datos de la subred y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Flujo Alternativo: Eliminar Nodo	
1. El actor selecciona la opción “Eliminar Subred”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación del nodo. 4. El sistema elimina el nodo y muestra un mensaje de éxito. 5. El caso de uso finaliza.	

f) Generar reporte de sesiones

Caso de Uso: Generar Reporte de Sesiones	
Descripción:	Generar un reporte basado en la bitácora de la aplicación que permita auditar las sesiones iniciadas, almacenando información como usuario, IP de origen, sistema operativo y hora.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Generar Reporte de Sesiones	
1. El actor selecciona la opción “Generar Reporte de Sesiones”. 2. La aplicación muestra un formulario que permite filtrar por rango de fechas y nombre de Usuario. 3. El actor selecciona las opciones de filtrado que desee y pulsa la opción “Ver Reporte”. 4. La aplicación genera el reporte solicitado. 5. El caso de uso finaliza.	
Postcondición:	Se ha generado satisfactoriamente un reporte de sesiones.

g) Administrar directivas de seguridad

Caso de Uso: Administrar directivas de seguridad	
Descripción:	La aplicación permitirá administrar una lista de direcciones IP hábiles que podrán iniciar sesión remota en el sistema.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Registrar IP Válida	
1. El actor selecciona la opción “Información del Servidor”. 2. La aplicación muestra la configuración actual del servidor. 3. El actor registra la nueva IP válida dentro de la lista de IPs Válidas y pulsa la opción Guardar. 4. La aplicación guarda los datos del nodo y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente una nueva dirección IP válida para iniciar sesión en la aplicación.
Flujo Alternativo: Eliminar IP Válida	
1. El actor selecciona la opción “Eliminar IP”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación de registro. 4. El sistema elimina la dirección IP de la lista y muestra un mensaje de éxito. 5. El caso de uso finaliza.	

h) Mantener configuración de red

Caso de Uso: Mantener configuración de red	
Descripción:	La aplicación permitirá activar y desactivar la opción de reenvío de paquetes; así como mantener actualizados los valores de la red.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión
Flujo Principal: Recargar valores de la red	
1. El actor selecciona la opción “Recargar”. 2. La aplicación busca los valores de la configuración de la red y actualiza los valores dentro de la base de datos. 3. El caso de uso finaliza.	
Postcondición:	Se ha actualizado satisfactoriamente los valores de red asignados al servidor.
Flujo Alternativo: Habilitar el reenvío de paquetes	
1. El actor selecciona la opción “Habilitar Reenvío de Paquetes”. 2. La aplicación activa el reenvío de paquetes. 3. El caso de uso finaliza.	

Caso de Uso: Mantener configuración de red
Flujo Alternativo: Deshabilitar el reenvío de paquetes
1. El actor selecciona la opción “Deshabilitar Reenvío de Paquetes”. 2. La aplicación desactiva el reenvío de paquetes. 3. El caso de uso finaliza.

8.2. Módulo Iptables

Este paquete contiene los casos que corresponden a la administración de la herramienta Iptables.

Los casos de uso incluidos en este paquete son: Configurar Iptables, Mantener Reglas Iptables, Mantener Registro Histórico, Generar Reporte Iptables.

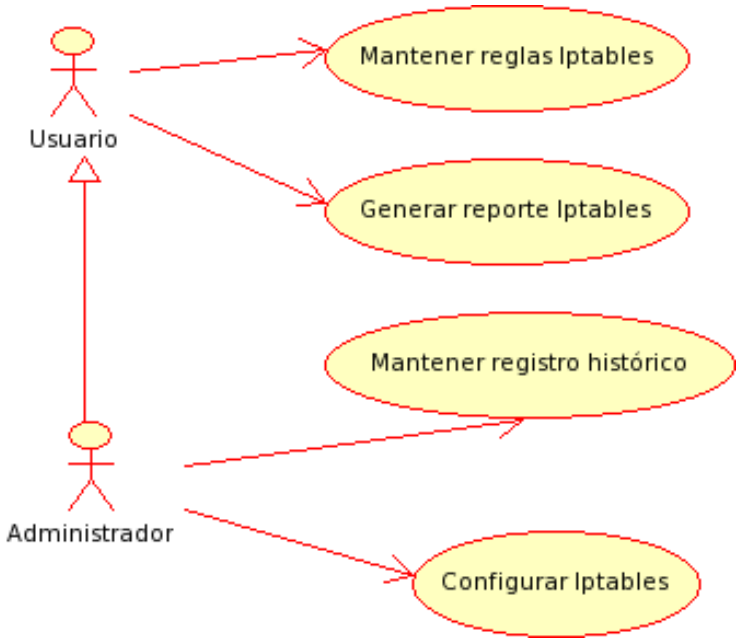


Imagen 8.2: Diagrama de Casos de Uso - Módulo Iptables

Módulo Iptables		
Código	**Caso de uso**	**Requerimientos Asociados**
CUI01	Configurar Iptables	FI001
CUI02	Mantener reglas Iptables	FI002, FI003, FI004, FI005, FI006, FI007
CUI03	Mantener registro histórico	FI008

Módulo Iptables		
CUI04	Generar reporte Iptables	FI009

Tabla 8.2: Casos de Uso - Módulo Iptables

a) Configurar Iptables

Caso de Uso: Configurar Iptables	
Descripción:	Permite registrar las configuraciones generales del módulo Iptables.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Iptables debe de estar habilitado.
Flujo Principal: Registrar Configuración	
1. El actor selecciona la opción “Configuración” del módulo Iptables. 2. La aplicación muestra un formulario para que el actor registre los datos de la configuración. 3. El actor llena los campos obligatorios del formulario (Descripción de la Configuración). 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos del usuario y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente la nueva configuración.
Flujo Alternativo: Importar Configuración	
1. El actor selecciona la opción “Importar configuración”. 2. La aplicación muestra una ventana para que seleccione la configuración que desea importar. 3. El actor selecciona el archivo de configuración que desea importar. 4. El sistema carga los datos de configuración. 5. El caso de uso finaliza.	

b) Mantener reglas Iptables

Caso de Uso: Mantener reglas Iptables	
Descripción:	Permite registrar, modificar y eliminar una nueva regla en el módulo Iptables.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Iptables debe de estar habilitado.
Flujo Principal: Registrar Regla	
1. El actor selecciona la opción “Nueva Regla”.	

Caso de Uso: Mantener reglas Iptables	
2. La aplicación muestra un formulario para que el actor registre los datos de la nueva regla. 3. El actor llena los campos obligatorios del formulario. 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos de la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente los datos de una nueva regla.
Flujo Alternativo: Modificar Regla	
1. El actor selecciona la opción “Modificar Regla”. 2. La aplicación muestra un formulario para que el actor edite los datos de la regla. 3. El actor actualiza el valor de los campos. 4. El sistema verifica los datos ingresados. Si son correctos actualiza los datos de la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Flujo Alternativo: Eliminar Regla	
1. El actor selecciona la opción “Eliminar Regla”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación de la regla. 4. El sistema elimina la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	

c) Mantener registro histórico

Caso de Uso: Mantener registro histórico	
Descripción:	La aplicación permitirá exportar las configuraciones y reglas establecidas dentro del módulo Iptables; así como eliminar configuraciones que ya no se consideren útiles.
Actores:	Administrador
Precondición:	El módulo Iptables debe de hallarse habilitado en el sistema.
Flujo Principal: Exportar registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos de cada módulo Iptables. 3. El actor selecciona las opción que “Exportar” del registro que desee exportar. 4. La aplicación guardar los valores del registro dentro de un archivo. 5. El caso de uso finaliza.	
Postcondición:	Se ha exportado los valores del registro satisfactoriamente.
Flujo Alternativo: Eliminar registro histórico	
1. El actor selecciona la opción “Registro histórico”.	

Caso de Uso: Mantener registro histórico
2. La aplicación muestra una tabla con los registros históricos del módulo Iptables. 3. El actor selecciona las opción que “Eliminar” del registro que desee eliminar. 4. La aplicación muestra un mensaje de confirmación. 5. El actor acepta la eliminación del registro. 6. La aplicación elimina el registro. 7. El caso de uso finaliza.
Flujo Alternativo: Restaurar del registro histórico
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos del módulo Iptables. 3. El actor selecciona las opción que “Restaurar” del registro que desee restaurar. 4. La aplicación restaura las reglas o configuraciones relacionadas con dicho registro. 5. El caso de uso finaliza.

d) Generar reporte Iptables

Caso de Uso: Generar Reporte Iptables	
Descripción:	Generar un reporte basado en la bitácora de la herramienta en un formato de fácil lectura para el usuario.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Iptables debe de estar habilitado.
Flujo Principal: Generar Reporte Iptables	
1. El actor selecciona la opción “Generar Reporte Iptables”. 2. La aplicación muestra un formulario que permite filtrar por rango de fechas o por un rango de cantidades. 3. El actor selecciona las opciones de filtrado que desee y pulsa la opción “Ver Reporte”. 4. La aplicación genera el reporte solicitado. 5. El caso de uso finaliza.	
Postcondición:	Se ha generado satisfactoriamente un reporte de la herramienta Iptables.

8.3. Módulo Squid

Este paquete contiene los casos que corresponden a la administración de la herramienta Squid.

Los casos de uso incluidos en este paquete son: Configurar Squid, Mantener Reglas Squid, Mantener Registro Histórico, Generar Reporte Squid.

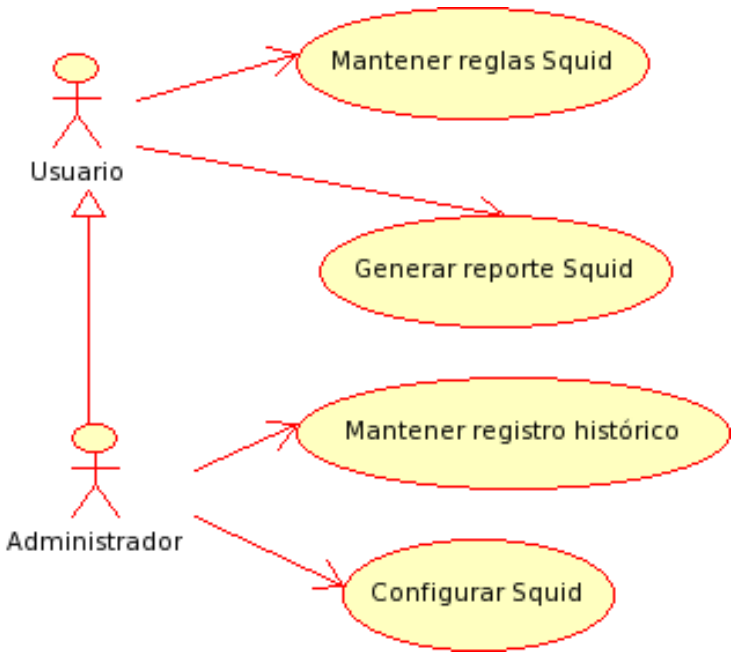


Imagen 8.3: Diagrama de Casos de Uso - Módulo Squid

Módulo Squid		
Código	**Caso de uso**	**Requerimientos Asociados**
CUQ01	Configurar Squid	FQ001, FQ003
CUQ02	Mantener reglas Squid	FQ002, FQ004, FQ005, FQ006
CUQ03	Mantener registro histórico	FQ007
CUQ04	Generar reporte Squid	FQ008

Tabla 8.3: Casos de Uso - Módulo Squid

a) Configurar Squid

Caso de Uso: Configurar Squid	
Descripción:	Permite registrar las configuraciones generales del módulo Squid.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Squid debe de estar habilitado.
Flujo Principal: Registrar Configuración	
1. El actor selecciona la opción “Configuración” del módulo Squid. 2. La aplicación muestra un formulario para que el actor registre los datos de la configuración. 3. El actor llena los campos obligatorios del formulario. 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos del usuario y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente la nueva configuración.
Flujo Alternativo: Importar Configuración	
1. El actor selecciona la opción “Importar configuración”. 2. La aplicación muestra una ventana para que seleccione la configuración que desea importar. 3. El actor selecciona el archivo de configuración que desea importar. 4. El sistema carga los datos de configuración. 5. El caso de uso finaliza.	

b) Mantener reglas Squid

Caso de Uso: Mantener reglas Squid	
Descripción:	Permite registrar, modificar y eliminar una nueva regla en el módulo Squid.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Squid debe de estar habilitado.
Flujo Principal: Registrar Regla	
1. El actor selecciona la opción “Nueva Regla”. 2. La aplicación muestra un formulario para que el actor registre los datos de la nueva regla. 3. El actor llena los campos obligatorios del formulario. 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos de la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente los datos de una nueva regla.
Flujo Alternativo: Modificar Regla	

Caso de Uso: Mantener reglas Squid
1. El actor selecciona la opción “Modificar Regla”. 2. La aplicación muestra un formulario para que el actor edite los datos de la regla. 3. El actor actualiza el valor de los campos. 4. El sistema verifica los datos ingresados. Si son correctos actualiza los datos de la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.
Flujo Alternativo: Eliminar Regla
1. El actor selecciona la opción “Eliminar Regla”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación de la regla. 4. El sistema elimina la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.

c) Mantener registro histórico

Caso de Uso: Mantener registro histórico	
Descripción:	La aplicación permitirá exportar las configuraciones y reglas establecidas dentro del módulo Squid; así como eliminar configuraciones que ya no se consideren útiles.
Actores:	Administrador
Precondición:	El módulo Squid debe de hallarse habilitado en el sistema.
Flujo Principal: Exportar registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos de cada módulo Squid. 3. El actor selecciona las opción que “Exportar” del registro que desee exportar. 4. La aplicación guardar los valores del registro dentro de un archivo. 5. El caso de uso finaliza.	
Postcondición:	Se ha exportado los valores del registro satisfactoriamente.
Flujo Alternativo: Eliminar registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos del módulo Squid. 3. El actor selecciona las opción que “Eliminar” del registro que desee eliminar. 4. La aplicación muestra un mensaje de confirmación. 5. El actor acepta la eliminación del registro. 6. La aplicación elimina el registro. 7. El caso de uso finaliza.	

Caso de Uso: Mantener registro histórico
Flujo Alternativo: Restaurar del registro histórico
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos del módulo Squid. 3. El actor selecciona las opción que “Restaurar” del registro que desee restaurar. 4. La aplicación restaura las reglas o configuraciones relacionadas con dicho registro. 5. El caso de uso finaliza.

d) Generar reporte Squid

Caso de Uso: Generar Reporte Squid	
Descripción:	Generar un reporte basado en la bitácora de la herramienta en un formato de fácil lectura para el usuario.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Squid debe de estar habilitado.
Flujo Principal: Generar Reporte Squid	
1. El actor selecciona la opción “Generar Reporte Squid”. 2. La aplicación muestra un formulario que permite filtrar por rango de fechas o por un rango de cantidades. 3. El actor selecciona las opciones de filtrado que desee y pulsa la opción “Ver Reporte”. 4. La aplicación genera el reporte solicitado. 5. El caso de uso finaliza.	
Postcondición:	Se ha generado satisfactoriamente un reporte de la herramienta Squid.

8.4. Módulo Snort

Este paquete contiene los casos que corresponden a la administración de la herramienta Snort.

Los casos de uso incluidos en este paquete son: Configurar Snort, Mantener Reglas Snort, Mantener Registro Histórico, Generar Reporte Snort.

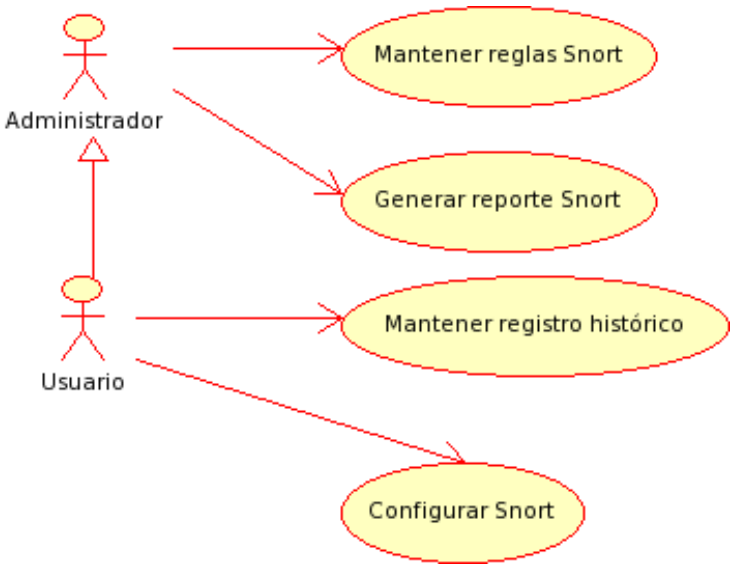


Imagen 8.4: Diagrama de Casos de Uso - Módulo Snort

Módulo Snort		
Código	**Caso de uso**	**Requerimientos Asociados**
CUS01	Configurar Snort	FS001, FS002
CUS02	Mantener reglas Snort	FS003
CUS03	Generar reporte Snort	FS004

Tabla 8.4: Casos de Uso - Módulo Snort

a) Configurar Snort

Caso de Uso: Configurar Snort	
Descripción:	Permite registrar las configuraciones generales del módulo Snort.
Actores:	Administrador
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Snort debe de estar habilitado.
Flujo Principal: Registrar Configuración	
1. El actor selecciona la opción “Configuración” del módulo Snort. 2. La aplicación muestra un formulario para que el actor registre los datos de la configuración. 3. El actor llena los campos obligatorios del formulario. 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos del usuario y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente la nueva configuración.

Caso de Uso: Configurar Snort
Flujo Alternativo: Importar Configuración
1. El actor selecciona la opción “Importar configuración”. 2. La aplicación muestra una ventana para que seleccione la configuración que desea importar. 3. El actor selecciona el archivo de configuración que desea importar. 4. El sistema carga los datos de configuración. 5. El caso de uso finaliza.

b) Mantener reglas Snort

Caso de Uso: Mantener reglas Snort	
Descripción:	Permite registrar, modificar y eliminar una nueva regla en el módulo Snort.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Snort debe de estar habilitado.
Flujo Principal: Registrar Regla	
1. El actor selecciona la opción “Nueva Regla”. 2. La aplicación muestra un formulario para que el actor registre los datos de la nueva regla. 3. El actor llena los campos obligatorios del formulario. 4. La aplicación verifica los datos ingresados. Si son correctos guarda los datos de la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	
Postcondición:	Se ha registrado satisfactoriamente los datos de una nueva regla.
Flujo Alternativo: Eliminar Regla	
1. El actor selecciona la opción “Eliminar Regla”. 2. La aplicación muestra un mensaje de confirmación. 3. El actor acepta la eliminación de la regla. 4. El sistema elimina la regla y muestra un mensaje de éxito. 5. El caso de uso finaliza.	

c) Mantener registro Snort

Caso de Uso: Mantener registro histórico	
Descripción:	La aplicación permitirá exportar las configuraciones y reglas establecidas dentro del módulo Snort; así como eliminar configuraciones que ya no se consideren útiles.
Actores:	Administrador
Precondición:	El módulo Snort debe de hallarse habilitado en el sistema.

Caso de Uso: Mantener registro histórico	
Flujo Principal: Exportar registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos de cada módulo Snort. 3. El actor selecciona las opción que “Exportar” del registro que desee exportar. 4. La aplicación guardar los valores del registro dentro de un archivo. 5. El caso de uso finaliza.	
Postcondición:	Se ha exportado los valores del registro satisfactoriamente.
Flujo Alternativo: Eliminar registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos del módulo Snort. 3. El actor selecciona las opción que “Eliminar” del registro que desee eliminar. 4. La aplicación muestra un mensaje de confirmación. 5. El actor acepta la eliminación del registro. 6. La aplicación elimina el registro. 7. El caso de uso finaliza.	
Flujo Alternativo: Restaurar del registro histórico	
1. El actor selecciona la opción “Registro histórico”. 2. La aplicación muestra una tabla con los registros históricos del módulo Snort. 3. El actor selecciona las opción que “Restaurar” del registro que desee restaurar. 4. La aplicación restaura las reglas o configuraciones relacionadas con dicho registro. 5. El caso de uso finaliza.	

d) Generar reporte Snort

Caso de Uso: Generar Reporte Snort	
Descripción:	Generar un reporte basado en la bitácora de la herramienta en un formato de fácil lectura para el usuario.
Actores:	Administrador, Usuario
Precondición:	Haberse ejecutado el caso de uso Iniciar Sesión El módulo Iptables debe de estar habilitado.
Flujo Principal: Generar Reporte Snort	
1. El actor selecciona la opción “Generar Reporte Snort”. 2. La aplicación muestra un formulario que permite filtrar por rango de fechas o por un rango de cantidades.	

Caso de Uso: Generar Reporte Snort	
3. El actor selecciona las opciones de filtrado que desee y pulsa la opción “Ver Reporte”. 4. La aplicación genera el reporte solicitado. 5. El caso de uso finaliza.	
Postcondición:	Se ha generado satisfactoriamente un reporte de la herramienta Snort.

9. Pantallas

A continuación se presentarán los principales diseños sobre los que se ha basado la elaboración de las pantallas de la aplicación.

9.1. Secciones de la Pantalla

Imagen 9.1: Secciones de la pantalla

Las Pantallas se hallan divididas en cuatro (4) secciones:

Cabecera: Contiene el logo y el nombre de la aplicación, asimismo, muestra el nombre del usuario y la opción de cierre de sesión.
Pie de página: Contiene datos relacionados a la fecha de creación de la aplicación.
Menú: Muestra el menú principal de la aplicación.
Contenidos: Es la zona de la pantalla en donde se mostrarán cada una de las vistas a las que el usuario acceda.

9.2. Estructura del menú principal

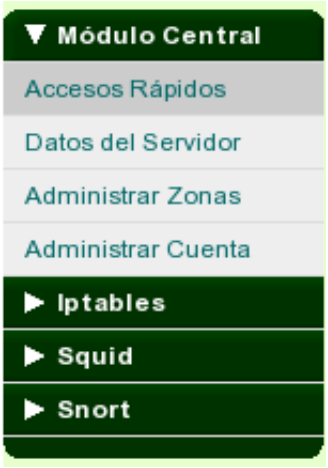


Imagen 9.2: Estructura del Menú Principal

El menú principal está conformado por dos (2) secciones:

Módulo Central: Contiene enlaces a cada una de las pantallas del módulo central de la aplicación.

Módulos Agregados: Contiene los enlaces a cada una de las vistas de los módulos auxiliares.

9.3. Iniciar Sesión

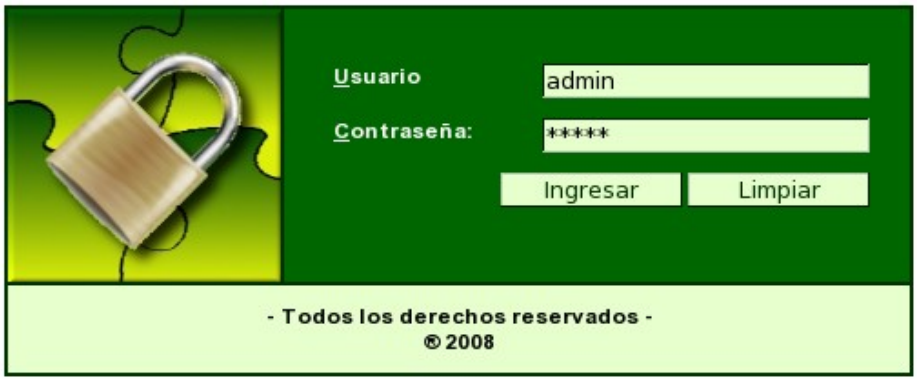


Imagen 9.3: Inicio de Sesión

A través de esta pantalla el usuario iniciará sesión en la aplicación.

9.4. Panel de Accesos Rápidos

Imagen 9.4: Panel de Accesos Rápidos

Esta pantalla contiene una lista de accesos rápidos a las principales funcionalidades del módulo central.

9.5. Administrar Usuario

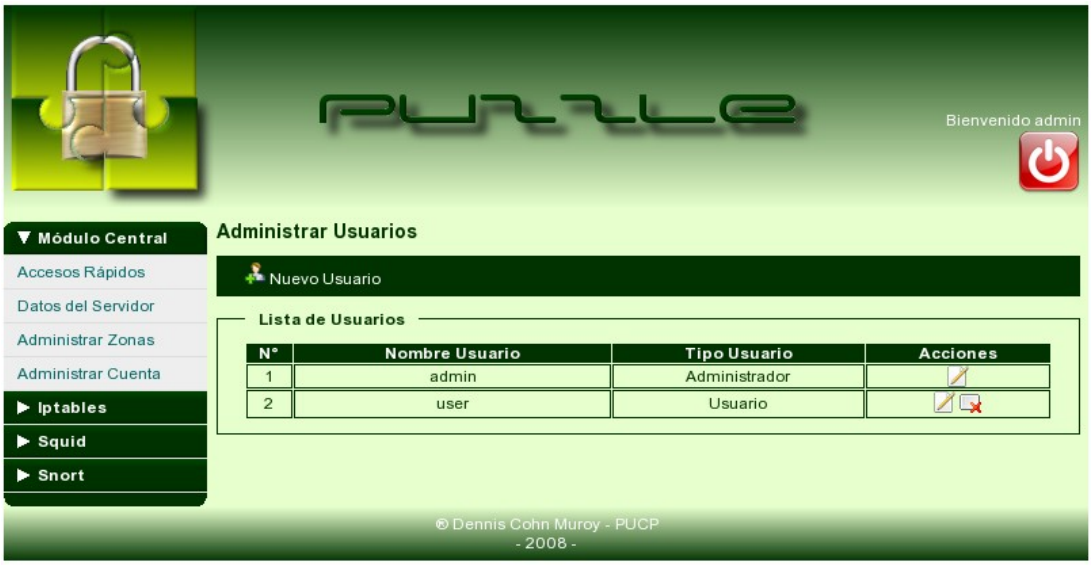


Imagen 9.5: Administrar Usuario

Esta pantalla muestra la lista de usuarios registrados en el sistema; asimismo, brinda la opción de poder eliminar el usuario seleccionado.

9.6. Registrar Usuario

Imagen 9.6: Registrar Usuario

Esta pantalla permite registrar los datos de un nuevo usuario en la

aplicación.

9.7. Información del Servidor

Imagen 9.7: Información del Servidor

Esta pantalla muestra las características de la configuración del servidor (Nombre del Servidor, Puerta de enlace, DNSs, Interfaces detectadas).

Adicionalmente, permite registrar una lista de direcciones IP hábiles para iniciar sesión en el equipo.

9.8. Administrar Servicios

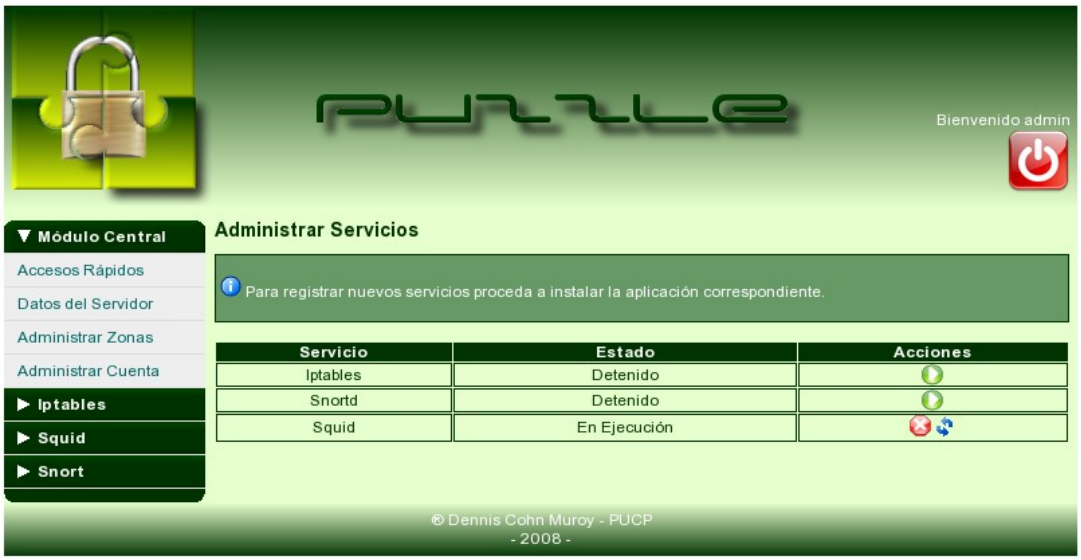


Imagen 9.8: Administrar Servicios

Esta pantalla muestra y administra los servicios presentes en el servidor que son administrados por cada uno de los módulos de la aplicación.

9.9. Registro Histórico de Accesos al Sistema

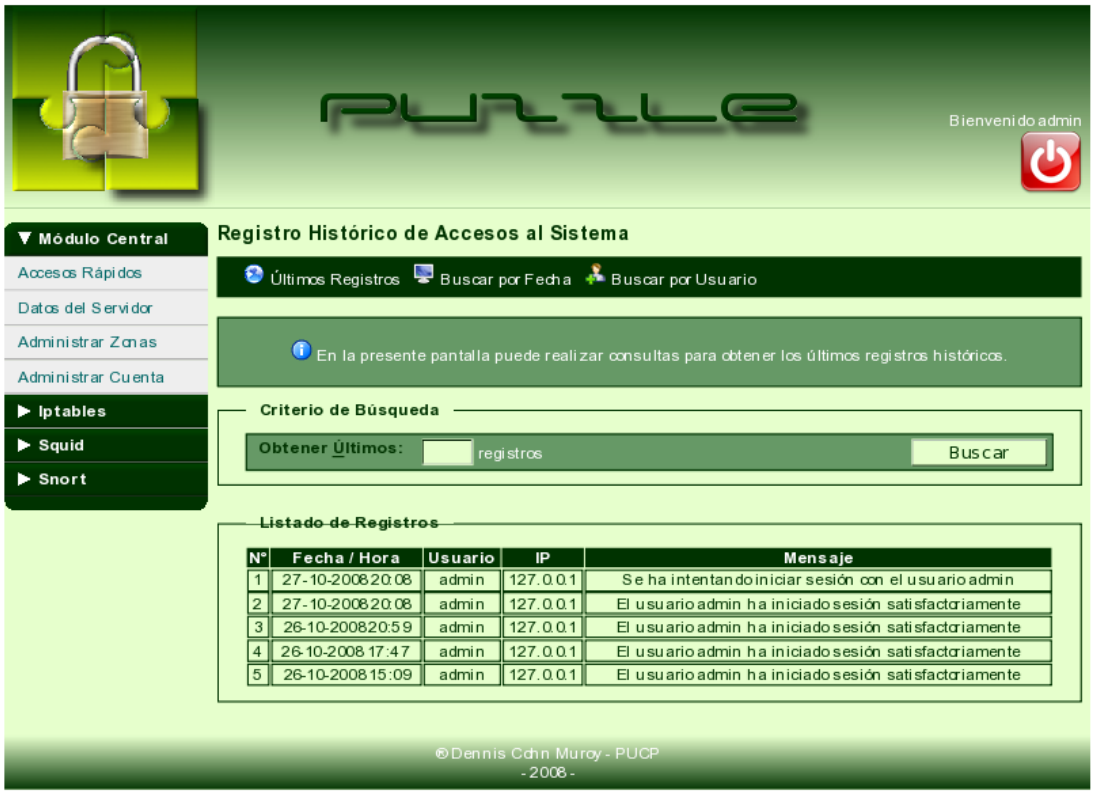


Imagen 9.9: Registro Histórico de Accesos al Sistema

Esta pantalla muestra el registro de todos los intentos de accesos exitosos o infructuosos en el sistema, así como la cuentas de usuario que fueron utilizadas.

10. Arquitectura

La arquitectura a utilizar será Web, con ello el usuario podrá tener acceso remoto a una interfaz gráfica para poder llevar a cabo la configuración de las herramientas de seguridad. Como prinipal requerimiento, el usuario deberá de hacer uso de un navegador de Internet.

10.1. Metas y Restricciones de la arquitectura

La meta principal de la arquitectura del sistema es mostrar los aspectos principales que influirán en la etapa de desarrollo.

Se tomarán en cuenta las siguientes metas y restricciones para el diseño de la arquitectura de la aplicación:

a) Metas

1. La aplicación permitirá, a los administradores de la red, acceder al sistema desde cualquier equipo autorizado vía su dirección IP.
2. Para poder acceder a la aplicación, se requiere de un nombre de usuario válido así como de una contraseña. Además, dependiendo del rango del usuario se deshabilitarán opciones de manejo de la aplicación.
3. Simplificar el uso de las herramientas utilizadas para asegurar la red local.

b) Restricciones del Sistema

1. La aplicación hará uso de archivos Sqlite como Base de Datos, para almacenar las configuraciones llevadas a cabo.
2. Las computadoras que brindarán el servicio cliente del sis-

tema no deberán de presentar potencias menores a las brindadas por una Pentium – 500 MHz con al menos 128 MB de RAM y 10 MB de espacio en el disco, con un Sistema Operativo GNU / Linux – Kernel 2.6.18.

10.2. Descripción de la arquitectura de la solución.

La aplicación se hallará conformada por un módulo central que permitirá, dentro de su lista de funciones, administrar una serie de módulos complementarios cada uno de los cuales aportará nuevas funcionalidades al sistema. Esto trae como beneficio la posibilidad de crear nuevos módulos con el objetivo de agregar nuevas funcionalidades a la herramienta. A lo largo del proyecto de tesis se implementarán tres (3) módulos complementarios, módulo Iptables, módulo Squid y módulo Snort.

Tanto la aplicación como las herramientas se hallarán ubicadas en un servidor GNU/Linux que se halle ejecutando el servicio web Apache, el cual, contará con el módulo PHP.

En la Imagen 3.1 se puede apreciar la arquitectura propuesta para el presente proyecto.

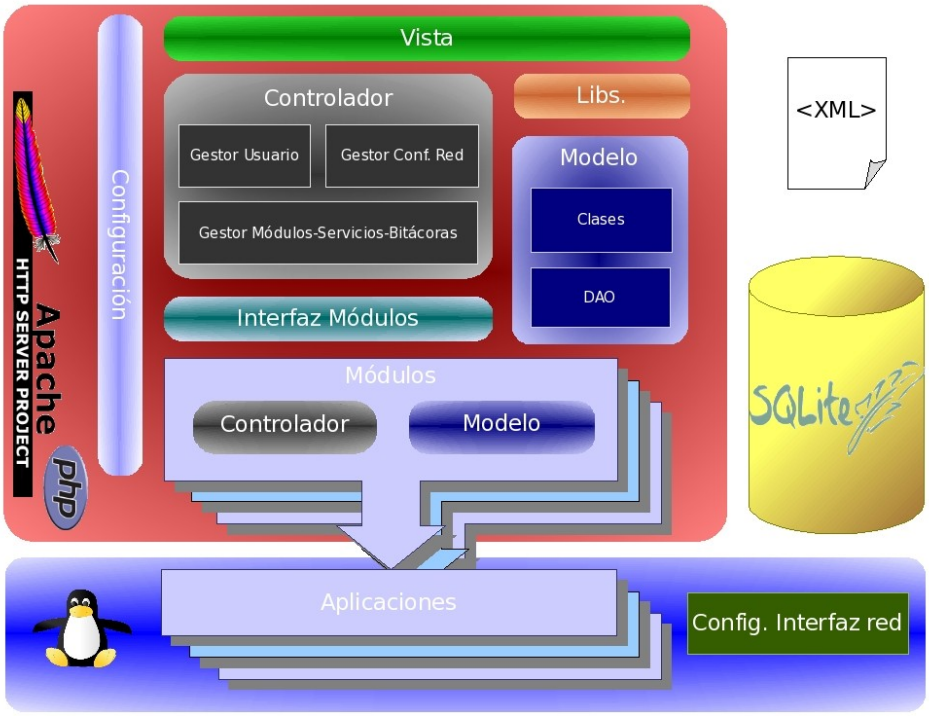


Imagen 10.1: Arquitectura de la solución

La aplicación está conformada por capas, cada una de las cuales aporta una funcionalidad al sistema. La lista de capas se detallan en la tabla siguiente:

Capa	Función que desempeña
Vista	Esta capa se encarga de presentar al usuario una interfaz con la que podrá interactuar, permitiéndole ingresar valores a la aplicación y visualizar las respuestas de la misma.
Controlador	Dentro de esta capa se hallan las clases que permiten manejar la lógica de negocio de todas las funcionalidades involucradas en el sistema. Asimismo será la encargada de controlar la interacción entre el módulo central de la aplicación con cada uno de los módulos complementarios.
Modelo	Esta es la capa que almacena los objetos de negocio de los que hace uso la aplicación a implementar, así como las clases que permiten llevar a cabo la persistencia contra una base de datos.
Módulo	Esta capa contendrá los diversos módulos integrados al sistema, cada uno de los cuales aportará nuevas funcionalidades al mismo.

Tabla 10.1: Capas de la Aplicación

10.3. Patrones a utilizar

Para el desarrollo del presente proyecto se hará uso de los siguientes patrones:

Modelo Vista Controlador (MVC): El patrón MVC está siendo utilizado para separar la capa de presentación (Vista) de los objetos del negocio (Modelo) y la lógica del negocio (Controlador), permitiendo así independizar la implementación de la lógica de la aplicación del diseño de las pantallas.

Data Mapper: Patrón utilizado para manejar la persistencia. Usando este mecanismo, los objetos que pertenecen al modelo desconocen la existencia de una base de datos. Una clase intermedia es la encargada de transferir los datos del modelo a la base de datos y viceversa.

Campo de Identidad: Consiste en almacenar el Id dentro del objeto; ello permite efectuar consultas haciendo uso de los Id, lo cual conlleva a un ahorro de tiempo.

Mapeo de Llave Foránea: Consiste en representar las llaves foráneas como relaciones entre dos objetos.

Solitario: Este patrón es utilizado en el ahorro de recursos; su objetivo es el de evitar instanciar una nueva clase si es que esta ya ha sido instanciada anteriormente.

Fachada: El patrón fachada permite establecer una interfaz a través de la cual se lleve a cabo la comunicación entre dos módulos. Es utilizada en el presente proyecto para garantizar la comunicación entre los módulos complementarios y el módulo central.

Template View: Este patrón permite generar páginas web dinámicas insertando etiquetas (ejemplo: etiquetas PHP) dentro del código HTML.

Front Controller: Este patrón establece la creación de una clase Controlador por cada Vista que se genere. Cada una de estas clases efectuará diferentes procedimientos dependiendo del evento que se accione en su respectiva Vista.

Server Session State: La utilidad de este patrón radica en poder almacenar información del cliente en el servidor, para poder hacer uso de estos valores más adelante.

10.4. Calidad

Para un mejor aprovechamiento de la arquitectura de software se dan los siguientes requerimientos de calidad:

Usabilidad

El sistema permitirá un manejo intuitivo por parte de los usuarios.

Eficiencia

El programa no demorará más de 1 minuto en guardar y aplicar las reglas generadas.

Seguridad

Para poder iniciar sesión en la aplicación será necesario contar con un usuario y una contraseña; asimismo, la dirección IP de nodo desde el cual se está accediendo debe de haber sido registrada en la aplicación.

Confiabilidad

El sistema tendrá en cuenta que la información ingresada en él sea válida, para lo cual mostrará mensajes que expliquen al usuario acerca de los errores que éste pudiera cometer y de aquellos que

pueda cometer el mismo sistema.

Mantenimiento

El sistema será flexible, facilitando su mantenimiento futuro.

Estándares

Se usará un estándar para todas las ventanas e interfaces de la aplicación.

11. Modelo del Dominio

El objetivo del presente documento es describir y mostrar mediante la utilización de diagramas los elementos utilizados en la etapa de análisis del desarrollo de la aplicación.

Para una mejor comprensión de las clases, se las han agrupado por módulo: Módulo Central, Módulo Iptables, Módulo Squid, Módulo Snort.

11.1. Vista Funcional del Módulo Central

Esta vista contiene el diagrama de clases correspondiente al módulo central. Estas clases están relacionadas con la configuración del equipo en el que se halla instalada la aplicación; así como la seguridad de la herramienta.

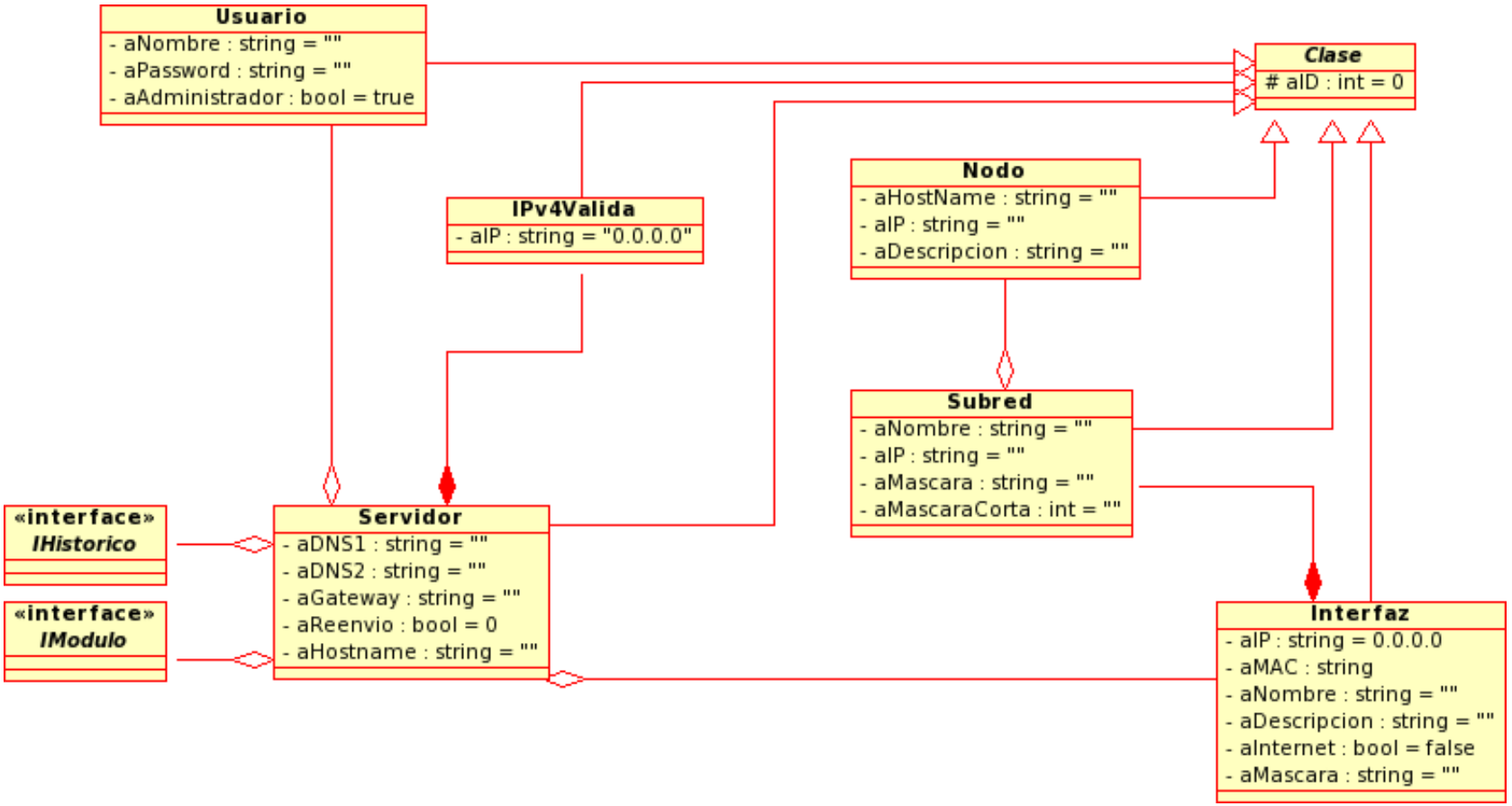


Imagen 11.1: Diagrama del Modelo del Dominio - Módulo Central

Módulo Central		
Código	**Clase**	**Descripción**
CC001	Servidor	Esta clase contiene la información básica sobre las configuraciones de red del servidor en donde se ejecuta la aplicación.
CC002	Usuario	Esta clase contiene información del usuario que iniciado sesión en la aplicación.
CC003	Interfaz	Esta clase representa a una zona dentro de la red local, conectada al servidor a través de una interfaz de red.
CC004	Nodo	Esta clase representa a un nodo dentro de una zona de la red local.
CC005	IPv4Valida	Esta clase representa una IP desde la cual es permitida iniciar sesión en la aplicación.
CC006	RegistroHistorico	Esta clase contiene información de lo intentos de inicio de sesión en la aplicación.
CC007	Subred	Esta clase representa una subred dentro de la red local.

Tabla 11.1: Modelo del Dominio - Módulo Central

11.2. Vista Funcional del Módulo Iptables

Esta vista contiene el diagrama de clases correspondiente al módulo Iptables. Estas clases están relacionadas con la configuración de la configuración de la herramienta Iptables, incluye la configuración de las opciones genrales de la misma, así como el mantenimiento de las reglas registradas.

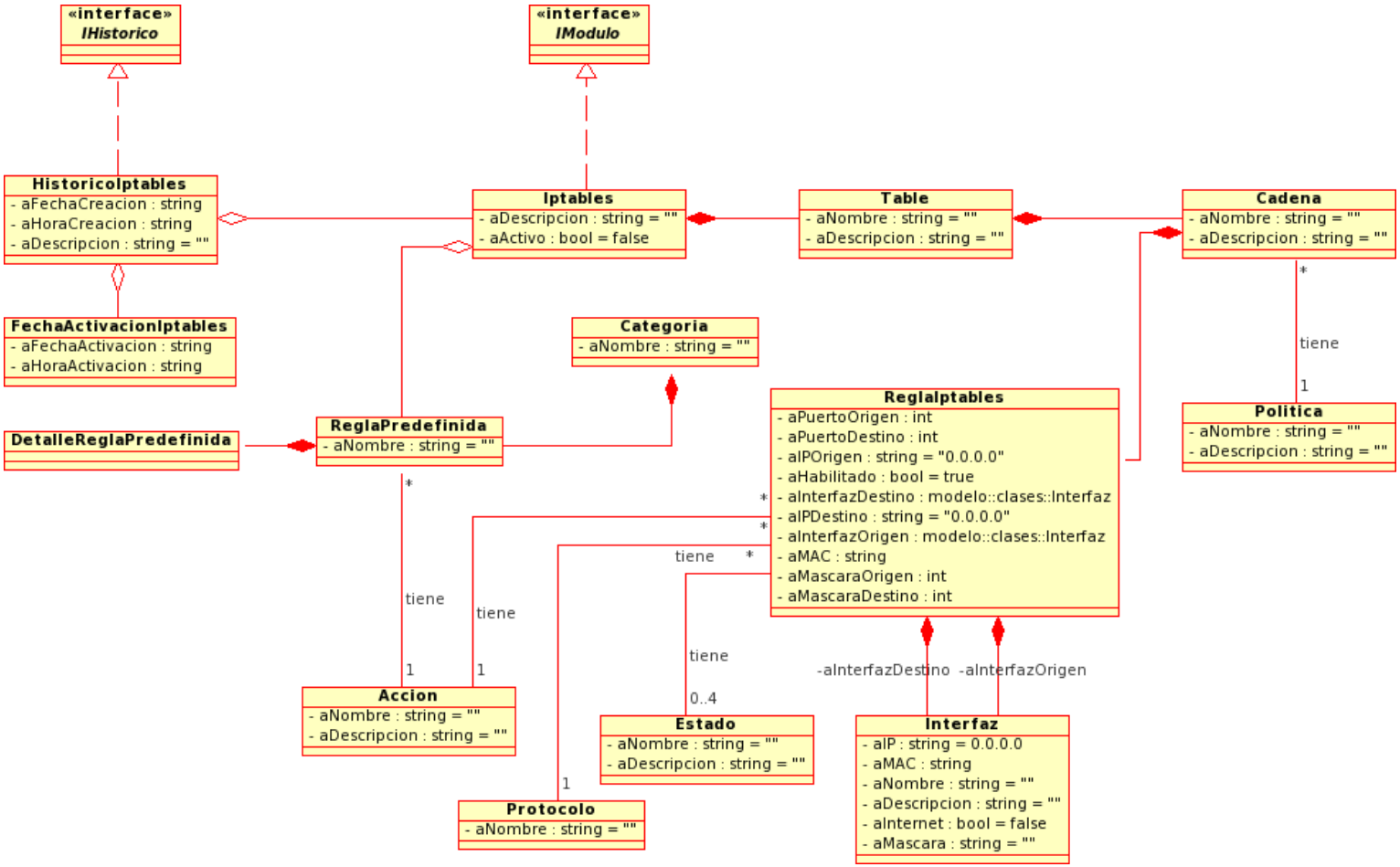


Imagen 11.2: Diagrama del Modelo del Dominio - Módulo Iptables

Módulo Iptables		
Código	**Clase**	**Descripción**
CI001	Iptables	Esta clase representa las configuraciones básicas de la herramienta Iptables.
CI002	ReglaIptables	Esta clase representa cada una de las reglas que pueden ser registradas en la aplicación Iptables.
CI003	Cadena	Esta clase representa una cadena de Iptables.
CI004	Table	Esta clase representa una de las tablas propias de la aplicación Iptables.
CI005	HistoricoIptables	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Iptables.
CI006	Accion	Esta clase representa las acciones a ejecutar sobre cada regla que se defina.
CI007	Categoria	Esta clase representa las categorías en las que se clasifican las reglas predefinidas.
CI008	DetalleReglaPredefinida	Esta clase representa la regla predefinida tal cual es entendida por la aplicación Iptables.
CI009	Estado	Esta clase representa los estados de las conexiones.
CI010	FechaActivacionIptables	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro del registro histórico de configuraciones de Iptables.
CI011	Politica	Esta clase representa las políticas predefinidas de cada cadena del Iptables.
CI012	Protocolo	Esta clase representa cada uno de los protocolos soportados por la aplicación.
CI013	ReglaPredefinida	Esta clase representa la regla predefinida que el administrador de la red quiere utilizar.

Tabla 11.2: Modelo del Dominio - Módulo Iptables

11.3. Vista Funcional del Módulo Squid

Esta vista contiene el diagrama de clases correspondiente al módulo Squid. Estas clases están relacionadas con la configuración de la configuración de la herramienta Squid, incluye la configuración de las funciones de proxy y cache, así como el mantenimiento de las reglas registradas.

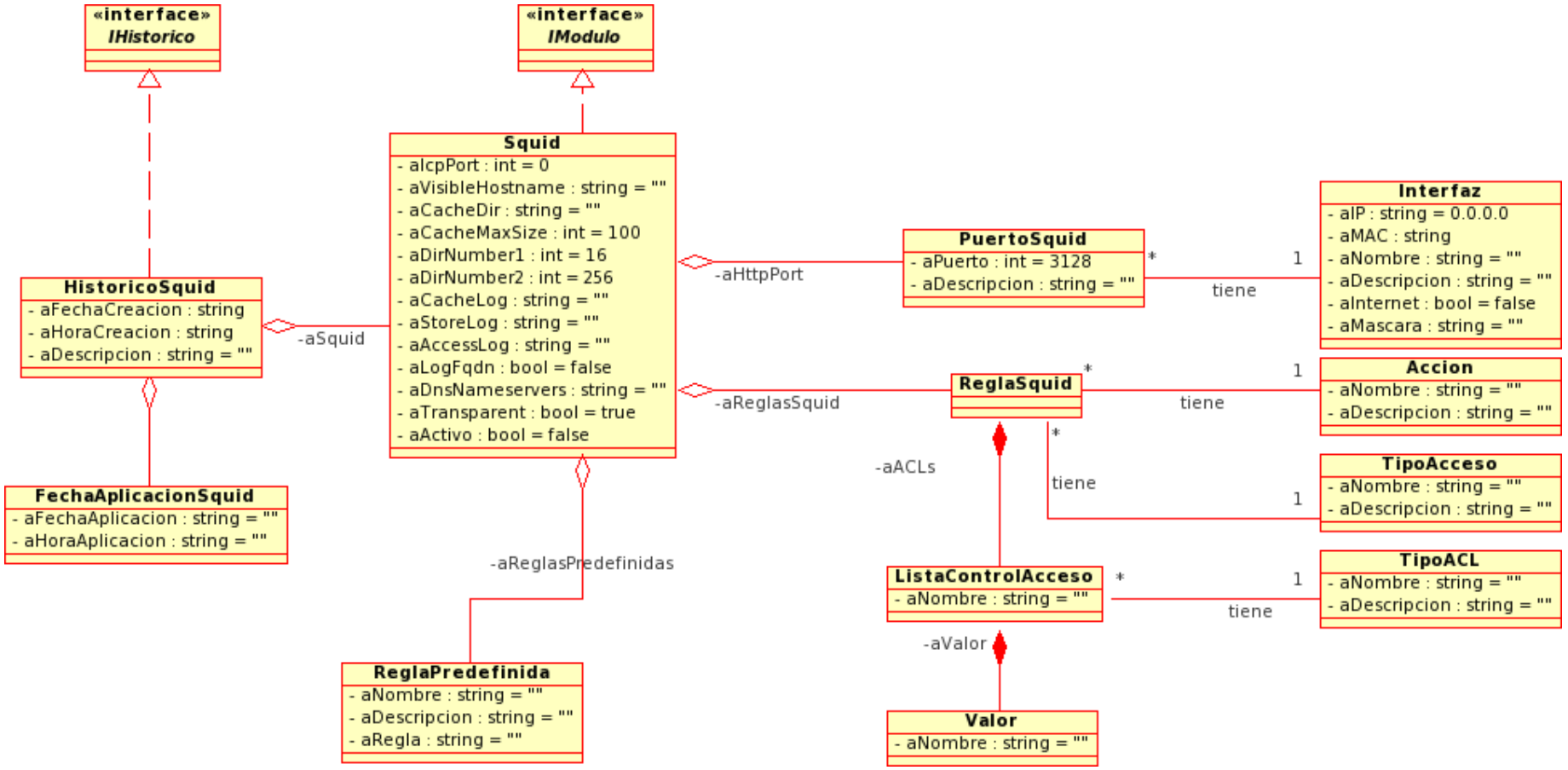


Imagen 11.3: Diagrama del Modelo del Dominio - Módulo Squid

Módulo Squid		
Código	**Clase**	**Descripción**
CQ001	Squid	Esta clase representa las configuraciones básicas de la herramienta Squid.
CQ002	ReglaPredefinida	Esta clase representa una regla predefinida que puede ser utilizada lista para las configuraciones de las opciones de proxy de la herramienta Squid.
CQ003	ListaControlAcceso	Esta clase agrupa un conjunto de reglas Squid, tanto para la cache como para el proxy.
CQ004	ReglaSquid	Esta clase representa cada una de las reglas que pueden ser registradas en el Squid.
CQ005	HistoricoSquid	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Squid.
CQ006	Accion	Esta clase representa la acción a ejecutar sobre una regla registrada.
CQ007	FechaActivacionSquid	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro del registro histórico de configuraciones del Squid.
CQ008	PuertoSquid	Clase que representa el puerto sobre el que se ejecuta el Squid.
CQ009	TipoACL	Representa los tipos de Lista de Control de Acceso.
CQ010	TipoAcceso	Clase que representa el tipo de acceso (web o caché)
CQ011	Valor	Listado de valores sobre los que se ejecutan las reglas registradas.

Tabla 11.3: Modelo del Dominio - Módulo Squid

11.4. Vista Funcional del Módulo Snort

Esta vista contiene el diagrama de clases correspondiente al módulo Snort. Estas clases están relacionadas con la configuración de la herramienta Snort, así como el mantenimiento de las reglas predefinidas.

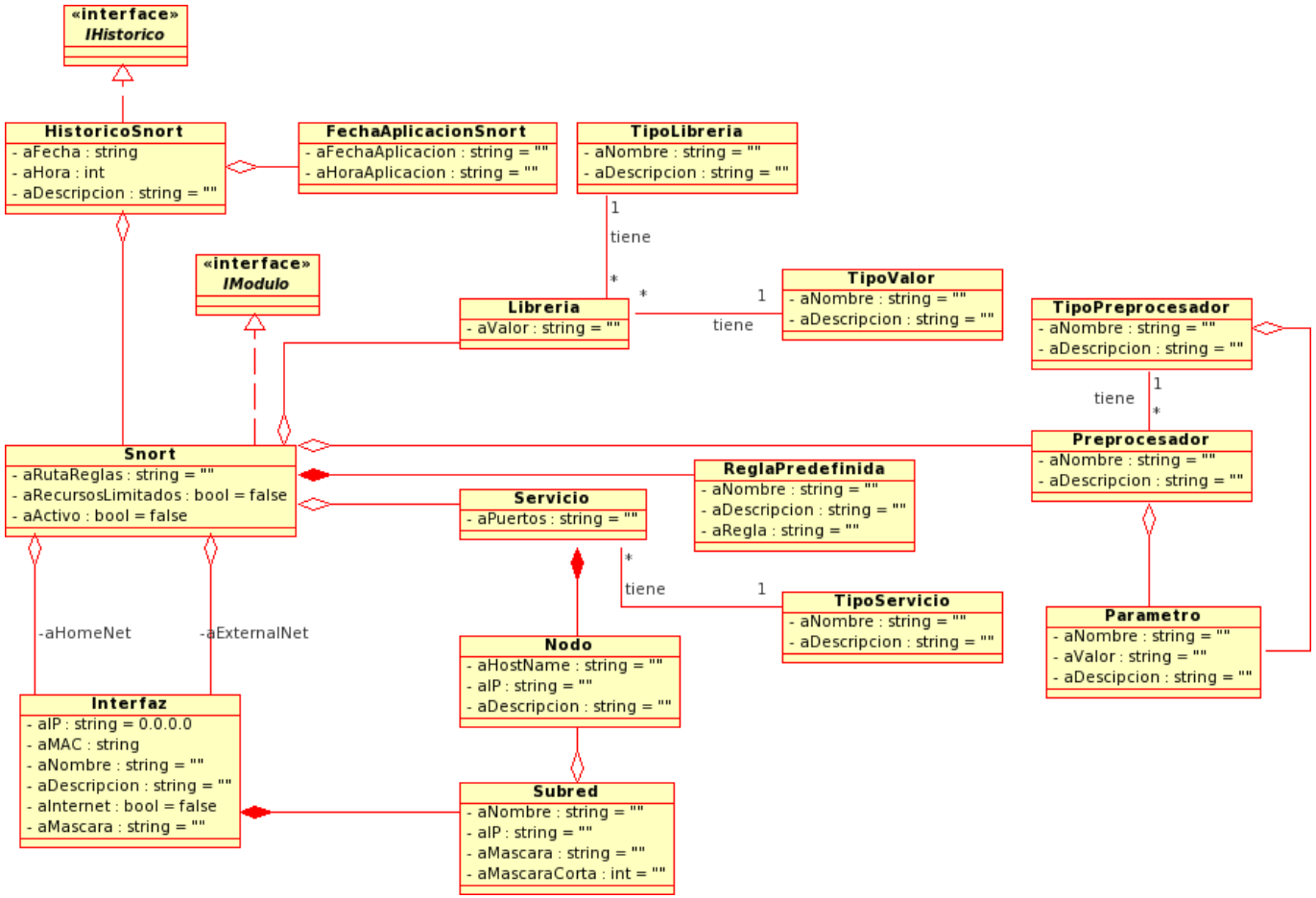


Imagen 11.4: Diagrama del Modelo del Dominio - Módulo Snort

Módulo Snort		
Código	**Clase**	**Descripción**
CS001	Snort	Esta clase representa las configuraciones básicas de la herramienta Snort.
CS002	Servicio	Esta clase representa los servicios que la herramienta Snort se halla manejando; así como los puertos a los que se hallan relacionados.
CS003	Preprocesador	Esta clase representa los preprocesadores que el Snort está utilizando.
CS004	ReglaPredefinida	Esta clase representa cada una de las reglas que pueden ser registradas en la herramienta Snort.
CS005	HistoricoSnort	Esta clase representa cada una de las entradas dentro del registro histórico de configuraciones de Snort.
CS006	FechaAplicacionSnort	Esta clase representa cada una de las fechas en las cuales se utilizó alguna de las entradas dentro del registro histórico de configuraciones del Snort.
CS007	Libreria	Esta clase representa las librerías que el Snort está utilizando.
CS008	Parametro	Esta clase representa los parámetros que el tipo de preprocesador puede recibir.
CS009	TipoLibreria	Esta clase representa los tipos de librerías disponibles.
CS010	TipoPreprocesador	Esta clase representa los tipos de preprocesadores disponibles.
CS011	TipoServicio	Esta clase representa los tipos de servicios disponibles.
CS012	TipoValor	Esta clase representa el tipo de fichero que el Snort utiliza como librería.

Tabla 11.4: Modelo del Dominio - Módulo Snort

12. Estándares de Programación

El objetivo del presente acápite es establecer un conjunto de estándares que serán utilizados para la codificación de las aplicaciones de software.

12.1. Clases

Los nombres de las clases deben de empezar con una letra en mayúscula seguida por caracteres en minúscula, sin dejar espacios en blanco.

En caso un nombre de clase esté formado por más de una palabra, únicamente el carácter inicial de cada palabra estarán en mayúscula, el resto de caracteres estarán en minúscula.

Los nombres de las clases deben de comenzar con uno de los prefijos presentados en la tabla siguiente:

Prefijo	Tipo de Clase	Ejemplo
Controlador	Objeto Controlador de la Vista	ControladorUsuario
Vista	Vista de la aplicación	VistaUsuario
DAO	Objeto de Acceso de Datos	DAOUsuario
-	Objetos de Negocio	Usuario

12.2. Atributos y Variables

Los nombres de las variables y atributos deben de comenzar con un prefijo identificador seguido por el nombre de la variable cuyo primer carácter debe de estar en mayúscula.

Las variables deben de ser definidas únicamente al inicio de cada bloque de código. Asimismo los atributos deben de ser declarados al inicio de la clase (antes de definir cualquier método) y siempre que se les haga referencia dentro de la clase, usar la palabra reservada “this”.

Prefijo Identificador	Alcance	Ejemplo
p	Parámetro	$pUsuario
a	Atributo	$aUsuario
l	Variable local	$lUsuario

12.3. Métodos

Los nombres de los métodos deben de estar en minúsculas. En caso un método esté conformado por más de una palabra, el carácter inicial de la siguiente palabra debe de estar en mayúsculas.

Los nombres de los métodos deben de describir la funcionalidad que realizan.

Ejemplo:
agregarNombre();
iniciarServicio();

12.4. Declaración de Objetos

El nombre de los objetos debe de ser el mismo que el de las clases; sin embargo, el primer carácter del objeto debe de ser una letra minúscula.

Si se hace uso de más de un objeto del mismo tipo, se le añadirá una palabra que comience con una letra mayúscula la cual brinde una mejor descripción del uso del objeto.

Ejemplo:
$lUsuario
$lUsuarioNuevo

12.5. Colecciones de Objetos

Los nombres de las colecciones de objetos deben de ser el nombre

de la clase en plural.

Ejemplo:

$lUsuarios

12.6. Componentes

Los nombres de los componentes de la ventana estarán conformados por un prefijo que indicará el tipo de componente seguido por el nombre del componente.

Prefijo	Tipo de componente
btn	Image button, button, submit button, reset button
txt	Text field, test area
chk	Checkbox
rdb	Radiobutton
cmb	Combobox
lst	SelectList

12.7. Interfaces

Los nombres de las interfaces deben de empezar con el carácter "I"

Ejemplo:

IModulo

IHistorial

12.8. Constantes

Los nombres de las constantes estarán en mayúsculas y si el nombre presenta espacios, estos estarán indicados con el carácter "_".

Ejemplo:

IGV

BASE_DATOS

12.9. Codificación

Descripción del archivo

Descripción de la clase

Descripción del método

Código indentado

```php
<?php
/**
 * @package /home/dennis/uml-generated-code/modelo/clases/Usuario.php
 * @class modelo/clases/Usuario.php
 * @author dennis
 * @version 1.0
 */

/**
 * class Usuario
 * Esta clase contiene información del usuario que iniciado sesión en la
 * aplicación.
 */
class Usuario
{

    /*** Attributes: ***/

    /**
     * @access private
     */
    private $aID = 0;

    /**
     * Nombre del usuario
     * @access private
     */
    private $aNombre = "";

    /**
     * Contraseña del usuario
     * @access private
     */
    private $aPassword = "";

    /**
     * Método constructor de la clase Usuario
     *
     * @param string pNombre Nombre del usuario
     * @param string pPassword Contraseña del usuario
     * @return
     * @access public
     */
    public function __construct( $pNombre = "",  $pPassword = "" ) {
        $this->aNombre = $pNombre;
        $this->aPassword = $pPassword;
    } // end of member function __construct

} // end of Usuario
?>
```

13. Catálogo de Pruebas

Las pruebas son un proceso importante dentro del desarrollo de una solución, ya que permiten detectar errores para su corrección.

Las pruebas a desarrollar serán ejecutadas a lo largo de todo el proceso de implementación de cada uno de los módulos.

13.1. Tipos de prueba

Se llevarán a cabo los siguientes tipos de prueba:

Pruebas de integración:

Caso de Prueba	PI001 – 1
Caso de Uso Asociado	CUC01
Objetivo de la Prueba	Verificar se puedan agregar usuarios al sistema.
Pasos	1. Seleccionar la opción “Nuevo Usuario” 2. Llenar los campos obligatorios con información válida (Nombre, Contraseña, Confirmar Contraseña). 3. Pulsar Aceptar.
Resultado Esperado	Se halla agregado un nuevo usuario al sistema.
Resultado	Éxito

Caso de Prueba	PI001 – 2
Caso de Uso Asociado	CUC01
Objetivo de la Prueba	Verificar se puedan modificar usuarios del sistema.
Pasos	1. Seleccionar la opción “Editar Usuario” 2. Modificar los valores del usuario (Nombre, Contraseña, Confirmar Contraseña). 3. Pulsar Aceptar
Resultado Esperado	Se han modificado los datos del usuario.
Resultado	Éxito

Caso de Prueba	PI001 – 3
Caso de Uso Asociado	CUC01
Objetivo de la Prueba	Verificar se puedan eliminar usuarios del sistema.
Pasos	1. Seleccionar la opción “Eliminar Usuario”
Resultado Esperado	Se ha eliminado el usuario del sistema.
Resultado	Éxito

Caso de Prueba	PI002 – 1
Caso de Uso Asociado	CUC02
Objetivo de la Prueba	Verificar que el sistema pueda iniciar servicios.
Pasos	1. Pulsar sobre el ícono “Iniciar Servicio”
Resultado Esperado	El servicio se inicia satisfactoriamente.
Resultado	Éxito

Caso de Prueba	PI002 – 2
Caso de Uso Asociado	CUC02
Objetivo de la Prueba	Verificar que el sistema pueda detener servicios.
Pasos	1. Pulsar sobre el ícono “Detener Servicio”
Resultado Esperado	El servicio se detiene de forma exitosa.
Resultado	Éxito

Caso de Prueba	PI003 – 1
Caso de Uso Asociado	CUC03
Objetivo de la Prueba	Verificar que el sistema pueda listar los módulos existentes.
Pasos	1. Pulsar sobre el ícono “Módulos Registrados”
Resultado Esperado	Se muestra el listado de módulos registrados
Resultado	Éxito

Caso de Prueba	PI004 – 1
Caso de Uso Asociado	CUC04
Objetivo de la Prueba	Verificar que el sistema pueda registrar subredes
Pasos	1. Pulsar sobre el ícono “Nueva Subred” 2. Ingresar los valores: Nombre de la subred, IP, Máscara, Máscara Corta, Interfaz 3. Pulsar sobre el botón “Guardar”
Resultado Esperado	Se almacenan los valores de la subred.
Resultado	Éxito

Caso de Prueba	PI004 – 2
Caso de Uso Asociado	CUC04
Objetivo de la Prueba	Verificar que se pueden agregar nodos a una subred
Pasos	1. Registrar la IP del nodo y su nombre dentro de la red. 2. Pulsar sobre el botón “Agregar Nodo”
Resultado Esperado	Se agrega un nuevo nodo a la subred.
Resultado	Éxito

Caso de Prueba	PI004 – 3
Caso de Uso Asociado	CUC04
Objetivo de la Prueba	Verificar que se pueden eliminar nodos de la subred
Pasos	1. Pulsar sobre el ícono “Eliminar Nodo”
Resultado Esperado	Se elimina el nodo de la subred.
Resultado	Éxito

Caso de Prueba	PI004 – 4
Caso de Uso Asociado	CUC04
Objetivo de la Prueba	Verificar que puedan ser modificados los valores de una subred.
Pasos	1. Pulsar sobre el ícono “Eliminar Subred” 2. Actualizar los valores: Nombre de la subred, IP, Máscara, Máscara Corta, Interfaz 3. Pulsar sobre el botón “Guardar”
Resultado Esperado	Se actualizan los valores de la subred.
Resultado	Éxito

Caso de Prueba	PI004 – 5
Caso de Uso Asociado	CUC02
Objetivo de la Prueba	Verificar que puedan eliminar una subred.
Pasos	1. Pulsar sobre el ícono “Eliminar Subred”
Resultado Esperado	Se elimina el registro de la subred.
Resultado	Éxito

Caso de Prueba	PI005 – 1
Caso de Uso Asociado	CUC05
Objetivo de la Prueba	Verificar que se pueda iniciar sesión utilizando únicamente la combinación adecuada de usuario

	y contraseña.
Pasos	2. Ingresar un nombre de usuario registrado en el sistema. 3. Ingresar la contraseña del usuario.
Resultado Esperado	Se inicie sesión satisfactoriamente.
Resultado	

Caso de Prueba	PI005 – 2
Caso de Uso Asociado	CUC05
Objetivo de la Prueba	Verificar que se pueda iniciar sesión utilizando únicamente la combinación adecuada de usuario y contraseña.
Pasos	1. Ingresar un nombre de usuario registrado en el sistema 2. Ingresar una contraseña no válida
Resultado Esperado	Se muestra un mensaje indicando un error al intentar iniciar sesión.
Resultado	Éxito

Caso de Prueba	PI006 – 1
Caso de Uso Asociado	CUC06
Objetivo de la Prueba	Verificar que se pueda iniciar sesión utilizando únicamente la combinación adecuada de usuario y contraseña.
Pasos	1. Pulsar sobre el ícono “Registro Histórico”
Resultado Esperado	La aplicación muestra un registro con los intentos de inicio de sesión exitosos e infructuosos.
Resultado	Éxito

Caso de Prueba	PI007 – 1
Caso de Uso Asociado	CUC07
Objetivo de la Prueba	Verificar que únicamente se pueda iniciar sesión desde equipos cuya IP ha sido autorizada.
Pasos	1. Iniciar sesión desde un equipo cuya IP ha sido registrada en el sistema como IP Válida.
Resultado Esperado	Inicia Sesión de forma exitosa.
Resultado	Éxito

Caso de Prueba	PI007 – 2

Caso de Uso Asociado	CUC07
Objetivo de la Prueba	Verificar que únicamente se pueda iniciar sesión desde equipos cuya IP ha sido autorizada.
Pasos	1. Iniciar sesión desde un equipo cuya IP no ha sido registrada en el sistema como IP Válida.
Resultado Esperado	Muestra un mensaje de error indicando que no se puede iniciar sesión desde un equipo no registrado.
Resultado	Éxito

Caso de Prueba	PI008 – 1
Caso de Uso Asociado	CUC08
Objetivo de la Prueba	Verificar que se pueda habilitar / deshabilitar el reenvío de paquetes dentro de la red.
Pasos	1. Pulsar sobre el botón habilitar / deshabilitar reenvío de paquetes dentro de la ventana de “Datos del Servidor”.
Resultado Esperado	El reenvío de paquetes debe de ser habilitado / deshabilitado.
Resultado	Éxito

Caso de Prueba	PI009 – 1
Caso de Uso Asociado	CUI01
Objetivo de la Prueba	Verificar que el sistema permita efectuar configuraciones dentro del módulo Iptables.
Pasos	1. Pulsar sobre “Configuración Iptables”. 2. Registrar la Descripción y luego pulsar sobre “Guardar”.
Resultado Esperado	Se guarda la nueva configuración en Base de Datos.
Resultado	Éxito

Caso de Prueba	PI009 – 2
Caso de Uso Asociado	CUI01
Objetivo de la Prueba	Verificar que se pueda importar una configuración existente para el módulo iptables.
Pasos	1. Pulsar sobre “Configuración Iptables”. 2. Pulsar sobre “Importar Configuración”. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se importan los datos del archivo xml y se

	guardan en base de datos.
Resultado	Éxito

Caso de Prueba	PI010 – 1
Caso de Uso Asociado	CUI02
Objetivo de la Prueba	Verificar el sistema permita registrar reglas del Iptables.
Pasos	1. Pulsar sobre “Regla Iptables”. 2. Ingresar los valores que conformarán los parámetros de las reglas. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se registra una nueva regla.
Resultado	Éxito

Caso de Prueba	PI010 – 2
Caso de Uso Asociado	CUI02
Objetivo de la Prueba	Verificar el sistema eliminar reglas del Iptables.
Pasos	1. Pulsar sobre “Eliminar Regla”.
Resultado Esperado	La regla se elimina de forma exitosa.
Resultado	Éxito

Caso de Prueba	PI011 – 1
Caso de Uso Asociado	CUI03
Objetivo de la Prueba	Verificar que la aplicación permita exportar una configuración de Iptables previamente registrada.
Pasos	1. Pulsar sobre “Histórico Iptables”. 2. Pulsar sobre “Exportar Configuración”.
Resultado Esperado	Se exporta la configuración seleccionada a un archivo xml.
Resultado	Éxito

Caso de Prueba	PI011 – 2
Caso de Uso Asociado	CUI03
Objetivo de la Prueba	Verificar que la aplicación permita aplicar una configuración de Iptables previamente registrada.
Pasos	1. Pulsar sobre “Histórico Iptables”. 2. Pulsar sobre “Aplicar Configuración”.
Resultado Esperado	Se aplica la configuración seleccionada.
Resultado	Éxito

Caso de Prueba	PI011 – 3

Caso de Uso Asociado	CUI03
Objetivo de la Prueba	Verificar que la aplicación permita eliminar una configuración previamente registrada en el Histórico Iptables.
Pasos	1. Pulsar sobre “Eliminar Configuración ”.
Resultado Esperado	Se elimina la configuración seleccionada.
Resultado	Éxito

Caso de Prueba	PI012 – 1
Caso de Uso Asociado	CUI04
Objetivo de la Prueba	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Iptables.
Pasos	1. Pulsar sobre “Reporte Iptables”.
Resultado Esperado	Se muestra una lista con los registros de actividades detectados por el Iptables.
Resultado	Éxito

Caso de Prueba	PI013 – 1
Caso de Uso Asociado	CUQ01
Objetivo de la Prueba	Verificar que el sistema permita efectuar configuraciones dentro del módulo Squid.
Pasos	1. Pulsar sobre “Configuración Squid”. 2. Registrar los datos solicitados en pantalla y pulsar “Siguiente”. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se guarda la nueva configuración en Base de Datos.
Resultado	Éxito

Caso de Prueba	PI013 – 2
Caso de Uso Asociado	CUQ01
Objetivo de la Prueba	Verificar que se pueda importar una configuración existente para el módulo Squid.
Pasos	1. Pulsar sobre “Configuración Squid”. 2. Pulsar sobre “Importar Configuración”. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se importan los datos del archivo xml y se guardan en base de datos.
Resultado	Éxito

Caso de Prueba	PI014 – 1

Caso de Uso Asociado	CUQ02
Objetivo de la Prueba	Verificar el sistema permita registrar reglas del Squid.
Pasos	1. Pulsar sobre “Reglas Squid.”. 2. Ingresar los valores que conformarán los parámetros de las reglas. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se registra una nueva regla.
Resultado	Éxito

Caso de Prueba	PI014 – 2
Caso de Uso Asociado	CUQ02
Objetivo de la Prueba	Verificar el sistema eliminar reglas del Squid.
Pasos	1. Pulsar sobre “Eliminar Regla”.
Resultado Esperado	La regla se elimina de forma exitosa.
Resultado	Éxito

Caso de Prueba	PI015 – 1
Caso de Uso Asociado	CUQ03
Objetivo de la Prueba	Verificar que la aplicación permita exportar una configuración de Squid previamente registrada.
Pasos	1. Pulsar sobre “Histórico Squid”. 2. Pulsar sobre “Exportar Configuración”.
Resultado Esperado	Se exporta la configuración seleccionada a un archivo xml.
Resultado	Éxito

Caso de Prueba	PI015 – 2
Caso de Uso Asociado	CUQ03
Objetivo de la Prueba	Verificar que la aplicación permita aplicar una configuración de Squid previamente registrada.
Pasos	1. Pulsar sobre “Histórico Squid”. 2. Pulsar sobre “Aplicar Configuración”.
Resultado Esperado	Se aplica la configuración seleccionada.
Resultado	Éxito

Caso de Prueba	PI015 – 3
Caso de Uso Asociado	CUQ03
Objetivo de la Prueba	Verificar que la aplicación permita eliminar una configuración previamente registrada en el Histórico Squid.

Pasos	1. Pulsar sobre “Eliminar Configuración ”.
Resultado Esperado	Se elimina la configuración seleccionada.
Resultado	Éxito

Caso de Prueba	PI016 – 1
Caso de Uso Asociado	CUQ04
Objetivo de la Prueba	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Squid.
Pasos	1. Pulsar sobre “Reporte Squid”.
Resultado Esperado	Se muestra una lista con los registros de actividades detectados por el Squid.
Resultado	Éxito

Caso de Prueba	PI017 – 1
Caso de Uso Asociado	CUS01
Objetivo de la Prueba	Verificar que el sistema permita efectuar configuraciones dentro del módulo Snort.
Pasos	1. Pulsar sobre “Configuración Snort”. 2. Registrar los datos solicitados en pantalla y pulsar “Siguiente”. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se guarda la nueva configuración en Base de Datos.
Resultado	Éxito

Caso de Prueba	PI017 – 2
Caso de Uso Asociado	CUS01
Objetivo de la Prueba	Verificar que se pueda importar una configuración existente para el módulo Snort.
Pasos	1. Pulsar sobre “Configuración Snort”. 2. Pulsar sobre “Importar Configuración”. 3. Pulsar sobre “Guardar”.
Resultado Esperado	Se importan los datos del archivo xml y se guardan en base de datos.
Resultado	Éxito

Caso de Prueba	PI018 – 1
Caso de Uso Asociado	CUS02
Objetivo de la Prueba	Verificar el sistema permita registrar reglas del Snort.
Pasos	1. Pulsar sobre “Reglas Snort.”.

	2. Seleccionar las reglas que uno desea agregar.
Resultado Esperado	Se registra una nueva regla.
Resultado	Éxito

Caso de Prueba	PI018 – 2
Caso de Uso Asociado	CUS02
Objetivo de la Prueba	Verificar el sistema eliminar reglas del Snort.
Pasos	1. Pulsar sobre “Eliminar Regla”.
Resultado Esperado	La regla se elimina de forma exitosa.
Resultado	Éxito

Caso de Prueba	PI019 – 1
Caso de Uso Asociado	CUS03
Objetivo de la Prueba	Verificar que la aplicación permita generar un reporte basándose en la bitácora de Snort.
Pasos	1. Pulsar sobre “Reporte Snort”.
Resultado Esperado	Se muestra una lista con los registros de actividades detectados por el Snort.
Resultado	Éxito

Pruebas de sistema:

Caso de Prueba	PS001
Aspecto a Evaluar	Prueba Funcional
Objetivo	Verificar que las llamadas al sistema por parte de la aplicación, sean ejecutadas.
Resultado	Éxito
Observaciones	Para que las llamadas a procesos de administración pudieran ser efectuados, fue necesario editar el archivo sudoers.

Caso de Prueba	PS002
Aspecto a Evaluar	Prueba de Desempeño
Objetivo	Verificar que la aplicación presente un bajo consumo de recursos por parte de memoria y procesamiento.
Resultado	Éxito
Observaciones	Las pruebas fueron realizadas limitando el consumo de memoria a 32 MB para la ejecución de los scripts PHP.

Caso de Prueba	PS003
Aspecto a Evaluar	Prueba de Desempeño
Objetivo	Verificar que la página demore en ser visualizada, un tiempo menor a 5 segundos.
Resultado	Éxito
Observaciones	La prueba fue realizada deshabilitando servicios no críticos para el sistema.

Caso de Prueba	PS004
Aspecto a Evaluar	Prueba de Instalación
Objetivo	Verificar la instalación exitosa de la aplicación.
Resultado	Éxito
Observaciones	-

14. Glosario

ARPANET: (Advanced Research Projects Agency Network) La red de computadoras creada por encargo del Departamento de Defensa de los Estados Unidos como medio de comunicación para los diferentes organismos del país.

Computer Security Institute: Entidad que provee información relacionada a temas de seguridad de la información.

Dirección IP: Número que identifica de manera lógica y jerárquica a un nodo dentro de una red

e-business: Compra y venta de productos o de servicios a través de medios electrónicos, tales como el Internet y otras redes de computadoras.

GNU: (GNU is Not Unix) Proyecto iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente libre

Internet: Red de redes. Conjunto descentralizado de redes de comunicación interconectadas que funcionan como una red lógica única, de alcance mundial.

Linux: Kernel de sistema operativo libre creado por Linus Torvalds en 1991. Lanzado bajo la licencia pública general (GPL - General Public License) de GNU y desarrollado gracias a contribuciones provenientes de todo el mundo.

Nodo: Cada uno de los equipos que forman parte de una red.

Paquete: Un paquete de datos es una unidad fundamental de transporte de información dentro de una red

Política de Seguridad: Conjunto de reglas que se establecen dentro de

una organización para asegurar su información.

Protocolo: Conjunto de reglas que especifican el intercambio de datos durante la comunicación entre los nodos que forman parte de una red.

Puerto: Interfaz que permite que un programa se comunique a través de una red.

Red: Conjunto de equipos conectados por medio de cables, señales, ondas o cualquier otro método de transporte de datos, que comparten información , recursos y/o servicios.