UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Título: “Módulo de administración de cortafuego para HMAST”

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: José Carlos Miranda Azcuy

Tutores: Msc. Angel Goñi Oramas
Ing. Nélio Véliz Pedraza

La Habana

Mayo del 2013

"El hombre debe transformarse al mismo tiempo que la producción progresa; no realizaríamos una tarea adecuada si fuéramos tan sólo productores de artículos, de materias primas y no fuéramos al mismo tiempo productores de hombres."

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo a los _____ días del mes de _____ del año 2013

_____ _____

José Carlos Miranda Azcuy Msc. Ángel Goñi Oramas

_____

Ing. Nelio Véliz Pedraza

Agradecimientos

A pesar de todos los obstáculos que me afectaron durante estos cinco años, al fin estoy haciendo mis agradecimientos:

Agradezco de forma general a todas esas personas que hicieron posible que yo me inspirara hacer estas líneas en estos momentos, una de ellas exclusivamente soy yo.

Guiaron mi camino durante todo este tiempo: Mi padre, profesor incondicional, que no solo le agradezco sino que le dedico mi tesis. Mi madre, por apoyarme en todas mis decisiones y darme ese pedacito materno que todos necesitamos. Mi segunda madre: mi tía Nilda por creer y demostrarme que soy su tercer hijo, mi segundo padre: mi tío Luisito, por encaminarme por el camino correcto y alguien muy especial, mi abuelo por creer en mí y darme su fuerte apoyo y dedicación no solo en estos cinco años, sino en estos 23 años de vida.

Raramente se encuentran amigos en estos tiempos, yo los encontré y les agradezco por estos cinco años de amistad verdadera, en las buenas y en las malas. Les agradezco Yannier Peña Escalona, Yosley Padrón Rodríguez, Yordanis Arvelo Talavera.

Y como no se puede dejar de mencionar a mis colegas, agradezco a todos con los que compartí desde mi primer día en la universidad, en mis respectivos grupos, desde el 1102 hasta el 1510, especialmente a Mayelin Almaguer Sánchez y Ramón Agüero que siempre tuvimos algo en común y por esa razón los considero como mis hermanos.

Agradezco a mis compañeros de proyecto (HMAST) especialmente a los hermanos Villazón: Yasiel y Yadiel, a las más pequeña: Adrianet, al más grande y serio: Alexander, a la más seria: Nurisel, a la más fina: Maren y a mi hermano: Julio Cesar. A Todos les agradezco por formar parte de este proceso que da paso a una nueva vida.

Y si de amor se trata, no sabía que alguien como Saydi Marcillan Pacheco pudiera remover mi corazón como lo ha hecho en este tiempo. Siempre ha estado a mi lado ofreciéndome su amor incondicional y no solo eso, sino que le agradezco a su familia que me han acogido como un miembro más de la misma, dándome su cariño y consentimiento, especialmente mis suegros Nancy Pacheco Cárdenas y Alberto Marcillan.

A todos: Muchas Gracias!

Dedicatoria

RESUMEN

En el presente trabajo se investiga y desarrolla un módulo sobre tecnologías libres capaz de administrar el servicio de cortafuego. Para fomentar la investigación se hace un estudio de las principales herramientas teniendo en cuenta la aceptación de los usuarios y sus principales funcionalidades, También se documenta mediante la guía de la metodología de desarrollo ágil SXP, donde se define la arquitectura, herramientas a utilizar, lenguajes de programación etc.

El módulo desarrollado posee como principales funcionalidades, el filtrado de paquetes, traducción de direcciones de red y redirección de puertos. No obstante, aporta la idea de la eliminación de la sobre escritura de fichero, posee administración remota, optimización de los procesos y sobre todo elimina la necesidad de funcionar sobre *hardware* de altas prestaciones y de varios ordenadores para su funcionamiento.

Palabras claves: cortafuego, filtrado, redirección

Índice

Índice de tablas

Índice de ilustraciones

Introducción

En la sociedad actual, el acceso a cualquier clase de información se ve facilitado por diferentes tecnologías de la informática y las comunicaciones, que permiten interactuar local o remotamente con diferentes centros de cómputos. Sin embargo, este acceso a la información puede traer consigo dificultades en cuanto a la seguridad y privacidad de la información. Estas son cada vez más dependientes de sus redes informáticas y un problema que las afecte, puede llegar a comprometer la continuidad de las tareas y operaciones en las mismas.

Cada vez es mayor el número de atacantes que constantemente van adquiriendo habilidades más especializadas que les permiten obtener mayores beneficios. Otros factores a tener en cuenta son los problemas de seguridad provenientes del interior de la organización. Se puede citar el mal uso del *software* de administración de red, la falta del conocimiento adecuado para trabajar con el mismo y el hecho de que el personal interno que puede o no atentar contra la seguridad de la información.

Hoy en día las redes informáticas son muy inseguras, no obstante, se deben desarrollar políticas de seguridad para poder prevenir cualquier acceso no autorizado a la misma. Estas políticas deben ser muy exigentes si dicha organización cuenta con acceso a internet. Una de las medidas que se toman para proteger la red perimetral consiste en no permitir el acceso desde el exterior hacia el interior. Otra posibilidad es permitir ciertas clases de accesos y negar otros, eliminando así la dificultad antes mencionada. Este tipo de seguridad se puede implementar mediante el uso de un Cortafuego, siendo esta su principal funcionalidad.

Los cortafuegos están presentes tanto en sistemas operativos privativos como en sistemas operativos de código abierto. En los sistemas operativos privativos el cortafuego es administrado a través de herramientas visuales que facilitan las actividades de configuración y administración que no suelen ser triviales para un usuario no experto. En los sistemas operativos de código abierto también existen herramientas visuales de administración y configuración que facilitan dicha actividad, solo que en estos sistemas operativos es más complicado dicha actividad.

Teniendo en cuenta el planteamiento anteriormente abordado, se puede citar el ejemplo de la Universidad de las Ciencias Informáticas (UCI) específicamente el departamento de Servicios Integrados de Migración Asesoría y Soporte (SIMAYS) perteneciente al Centro de Software Libre (CESOL). En este centro se está desarrollando la Herramienta de Migración y Administración de Servicios Telemáticos (HMAST). De forma sencilla y rápida esta herramienta es capaz de administrar los servicios telemáticos de cualquier servidor de Windows a un servidor de código abierto. Entiéndase por servicios telemáticos: DNS, DHCP, Correo, etc. Actualmente esta herramienta se encuentra en su primera versión y solo tiene desarrollados dos módulos: uno para administrar los servicios de SSH y otro para administrar los de DHCP. Se hace necesaria la creación de nuevos módulos entre los que se destaca el módulo de administración de

cortafuego, con el objetivo de ser integrado y usado en la herramienta HASMT.

El **problema científico** a resolver es: ¿Cómo realizar la administración y configuración del servicio de cortafuego desde la herramienta HMAST?

El **objeto de estudio** del presente trabajo está enfocado a la administración y configuración del servicio de cortafuego basado en tecnologías de software libre y código abierto. Enmarcándose en el **campo de acción** la administración y configuración de servicios de cortafuego desde la herramienta HMAST.

Por lo que se propone como **objetivo general** desarrollar un módulo de administración del servicio cortafuego para la herramienta HMAST de forma tal que se mejore la administración del mismo en las empresas cubanas arribando a los siguientes **objetivos específicos**:

- Realizar un estudio del estado del arte de las herramientas de cortafuego.
- Realizar el análisis y diseño de la aplicación a desarrollar.
- Realizar la implementación y prueba del *software* requerido.

Con el propósito de dar cumplimiento a los objetivos planteados se hace necesario realizar las siguientes **tareas de investigación**:

- Revisión y análisis de bibliografía especializada de los servicios de cortafuego sobre plataformas GNU/Linux.
- Análisis y diseño del módulo de HMAST para la administración del servicio cortafuego.
- Desarrollo del módulo para la administración de servicios de cortafuego para su integración con HMAST.
- Realización de pruebas y evaluación de resultados al módulo desarrollado.

Esta investigación está sustentada sobre la base de la utilización de diferentes **métodos científicos** para la realización de la misma. Como métodos científicos teóricos se emplearon el **Analítico-Sintético** el cual se utiliza en la investigación de las herramientas para la administración de cortafuegos existentes. Ayuda a separar los mismos con el objetivo de realizar un estudio del funcionamiento y las configuraciones necesarias de estos, facilitando el desarrollo de la aplicación propuesta.

La **idea a defender** plantea: La realización de un módulo que permita administrar el servicio de cortafuego en la herramienta HMAST, facilitará dicho proceso (administración y configuración) a los usuarios de esta herramienta.

Capítulo 1. Fundamentación teórica**:** Se realiza un estudio acerca de las interfaces gráficas que facilitan la administración del servicio cortafuego, así como sus principales características y funcionalidades.

Además, se abordan conceptos claves que serán usados durante el desarrollo de la investigación, y se fundamenta acerca de la metodología y herramientas utilizadas en el desarrollo del módulo.

Capítulo 2. Análisis y diseño**:** Se fundamenta acerca de las herramientas que servirán de guía para el desarrollo del módulo de cortafuego, teniendo en cuenta sus principales funcionalidades y características. Se exponen los requerimientos funcionales y no funcionales y se define la metodología SXP para lograr el cumplimiento de los objetivos propuestos. También se especifican las historias de usuarios y se describe la arquitectura del módulo.

Capítulo 3. Implementación y prueba**:** Se implementa la solución propuesta y se describen las clases principales del sistema. Se elabora el plan de prueba, y se hacen las mismas con el objetivo de realizar las comprobaciones para validar el correcto funcionamiento de los requisitos planteados.

Capítulo 1: Fundamentación teórica

En el presente capítulo se realiza un análisis del estado del arte de los cortafuegos existentes además de una valoración de algunos puntos a tener en cuenta, tales como las principales características y funcionalidades, también se argumentan las principales interfaces gráficas que permiten y facilitan administrar el servicio cortafuego. Se documentan las principales funcionalidades de las mismas. Se describe y argumenta sobre la herramienta HMAST teniendo en cuenta su arquitectura. También se analiza el lenguaje de programación seleccionado para el desarrollo de la aplicación, proceso que se realiza de forma similar para determinar las herramientas utilizadas. Además, se fundamenta acerca del uso de la metodología SXP en la investigación.

Definición de cortafuego

Un cortafuego (*firewall* en inglés): es un sistema o grupo de sistemas que hace cumplir una política de control de acceso entre dos redes. Es decir, cualquier sistema (desde un simple *router* o enrutador hasta varias redes en serie) utilizado para separar en cuanto a seguridad se refiere un ordenador o subred del resto, protegiéndola así de servicios y protocolos que desde el exterior puedan suponer una amenaza a la seguridad. El espacio protegido, denominado perímetro de seguridad, suele ser propiedad de la misma organización, y la protección se realiza contra una red externa, no confiable, llamada zona de riesgo [1].

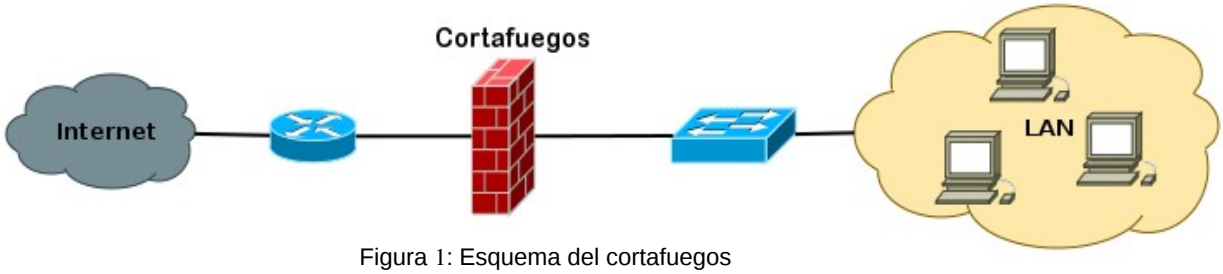


Figura 1: Esquema del cortafuegos

Características principales de los cortafuegos

Principales características de los cortafuegos [2]:

1. **Protección de la Red:** Mantiene alejados a los crackers de la red de la organización al mismo tiempo que permite acceder a todo el personal de la oficina, por lo que quedaría completamente vulnerable si dicho cracker o intruso estuviera dentro de la misma o de la organización.
2. **Control de uso de internet:** Permite bloquear el acceso a la información no adecuado, determinar que sitios puede visitar el usuario de la red interna y llevar un registro.
3. **Concentra la seguridad:** El cortafuego facilita la labor a los responsables de seguridad, dado que su máxima preocupación es la de encarar los ataques externos, vigilar y mantener un monitoreo.

4. **Control y estadísticas:** Permite controlar el uso de internet en el ámbito interno y conocer los intentos de conexiones desde el exterior y detectar actividades sospechosas.
5. **Choke-Point:** Permite al administrador de la red definir un “embudo” manteniendo al margen los usuarios no autorizados fuera de la red, prohibiendo potencialmente la entrada o salida al vulnerar los servicios de la red, y proporcionar la protección para varios tipos de ataques.
6. **Localización de un cortafuego:** Un cortafuego se encuentra frecuentemente instalado en el punto donde una red interna se conecta con internet. Todo tráfico externo de internet hacia la red interna pasa a través del cortafuego, así puede determinar si dicho tráfico es aceptable de acuerdo con sus políticas de seguridad.

Aunque el propósito principal de los cortafuego es mantener a los intrusos fuera del alcance de la información que es propiedad de un ente determinado (ya sea un usuario, una empresa o un gobierno), su posición dentro del acceso a distintas redes lo vuelve útil para controlar estadísticas de situaciones como usuarios que intentaron conectarse, tráfico que atraviesan las mismas, etc. Convirtiéndose en un sistema muy adecuado para auditar la red .

Si se tiene en cuenta que muchos de los cortafuegos existentes pueden o no contar con las características antes mencionadas, y que el objetivo de la herramienta HMAST es la administración de servicios telemáticos desde servidores, es necesario tener presente estas características para implementar el módulo propuesto, con el objetivo de asegurar la eficiencia en la administración de iptables, lo cual será explicado más adelante.

Ventajas del cortafuego

Los cortafuegos presentan las siguientes ventajas [3]:

Localizar las decisiones: Todo el tráfico de entrada y salida tiene que pasar a través de este sitio. Un cortafuego concentra las medidas de seguridad en este lugar de control: donde la red de la organización se conecta a internet.

Refuerza políticas: Un cortafuego puede reforzar las políticas de seguridad añadiendo políticas más complejas. Por ejemplo bloqueando una trasferencia de ficheros desde una parte de la red; controlando qué usuarios tienen acceso a qué sistemas. Y dependiendo de la tecnología del cortafuego, este puede ser más o menos complejo para añadir estas políticas.

Registra la actividad: Como todo el tráfico pasa a través del cortafuego, este provee un buen lugar para almacenar información sobre los usos de los sistemas y redes. Puede recopilar qué ocurre entre la zona protegida y la red externa.

Limita la exposición: Este es uno de los usos más relevantes de los cortafuegos. A veces un cortafuego

se usa para mantener una sección de la red separada de otra sección. Haciendo esto, se mantienen los problemas que puedan impactar en una sección separada del resto.

Desventajas

Las principales desventajas que poseen los cortafuegos en la actualidad son las siguientes [4]:

Los cortafuego ofrecen una excelente protección, pero no son la solución única y completa para la seguridad de los ordenadores ya que ciertos procesos están fuera del control del cortafuego y se necesitan otros métodos para protegerse de estos sucesos, ya sea incorporando otras herramientas, o conocer cuáles son los puntos débiles del cortafuego.

Dentro de la red: Un cortafuego puede prohibir a un usuario enviar información confidencial fuera de la red a través de la conexión a internet. Pero el mismo usuario puede copiar los datos en un dispositivo de almacenamiento, imprimirlos y llevárselos fuera del centro de trabajo. Si el atacante está dentro de la red el cortafuego no tiene la posibilidad de encarar la situación. Dentro los usuarios pueden apropiarse inadecuadamente de los datos, dañar *hardware* y *software*, sin pasar a través del cortafuego, por lo que es necesario protegerse con medidas internas de seguridad.

Conexiones que no pasan a través de él: Un cortafuego puede controlar el tráfico que pasa a través de él, pero no es la única forma de acceder a un dispositivo. Por ejemplo, si hay otra conexión *dial-in* para conectarse a los sistemas detrás del cortafuego, este no tiene ninguna forma de proteger la red de los intrusos que usen ese módem.

Virus: Los cortafuegos no pueden mantener a los virus alejados de la red interna. Muchos cortafuegos escanean todo el tráfico entrante para determinar si este está permitido, pero el escaneo de los datos, la mayoría de las veces se basan en las direcciones y puertos de origen y destino, no para los detalles de los datos. Incluso los cortafuegos más sofisticados no son muy prácticos contra los virus. Simplemente hay muchas maneras para enmascarar un virus entre otros datos. Determinar que existe un virus dado en un paquete que pasa a través del cortafuego es muy difícil. La forma más práctica de defenderse de los virus es mantener un *software* de protección basado en los ordenadores, y educando de los posibles peligros a los usuarios y de cómo protegerse de ellos.

Componentes de un Cortafuego

Un cortafuegos puede ser utilizado dependiendo de sus componentes, los cuales se muestran a continuación [1]:

Filtrado de paquetes: Actúa mediante la inspección de los paquetes (que representan la unidad básica de trasferencia de datos entre ordenadores en internet). Si un paquete coincide con el conjunto de reglas del filtro, el paquete se reducirá (descarte silencioso) o será rechazado (eliminando el paquete y enviando

una respuesta de error al emisor). Este tipo de filtrado de paquetes no presta atención a si el paquete es parte de una secuencia existente de tráfico. En su lugar, se filtra cada paquete basándose únicamente en la información contenida en el paquete en sí (por lo general utiliza una combinación del emisor del paquete y la dirección de destino, su protocolo, y, en el tráfico TCP y UDP, el número de puerto).

El filtrado de paquetes llevado a cabo por un cortafuego actúa en las tres primeras capas del modelo de referencia OSI, lo que significa que todo el trabajo lo realiza entre la red y las capas físicas. Cuando el emisor origina un paquete y es filtrado por el cortafuego, este último comprueba las reglas de filtrado de paquetes que lleva configuradas, aceptando o rechazando el paquete en consecuencia. Cuando el paquete pasa a través de cortafuego, este filtra el paquete mediante un protocolo y un número de puerto base.

Figura 2: Esquema de la Pila OSI

El cortafuego o el proxy de aplicación: Son aquellos que actúan sobre la capa de aplicación del modelo OSI. Este puede entender ciertas aplicaciones y protocolos (por ejemplo: protocolo de trasferencia de ficheros, DNS o navegación web), y permite detectar si un protocolo no deseado penetró a través de un puerto no estándar o si se está explotando un protocolo de forma perjudicial.

Un cortafuego de aplicación es mucho más seguro y fiable cuando se compara con un cortafuego de filtrado de paquetes, ya que repercute en las siete capas del modelo de referencia OSI. En esencia es similar a un cortafuego de filtrado de paquetes, con la diferencia de que también se puede filtrar el contenido del paquete.

Un cortafuego de aplicación puede filtrar protocolos de capas superiores tales como FTP, TELNET, DNS, DHCP, HTTP, TCP, UDP y TFTP. Por ejemplo, si una organización quiere bloquear toda la información relacionada con una palabra en concreto teniendo en cuenta que el cortafuego también actúa como *proxy*, puede habilitarse el filtrado de contenido para bloquear esa palabra en particular.

Monitorizar la actividad del cortafuego: Es algo indispensable para la seguridad de todo el perímetro protegido; la monitorización facilitará información sobre los intentos de ataque que se pueden ocasionar (origen, franjas horarias, tipos de acceso), así como la existencia de tramas que aunque no supongan un ataque no dejan de ser sospechosas [1].

Como se mencionó anteriormente el cortafuego da la posibilidad de guardar datos estadísticos. Para esto debe almacenar como mínimo la siguiente información:

- **Servicio de Información:** Fecha, y hora.
- **Información Remota:** Dirección IP del presunto intruso, así como el puerto y el protocolo utilizado.
- **Información Local:** Dirección IP de destino y puerto.
- **Información de filtro:** Actuación del filtro y qué adaptador de red lo hizo.
- **Información del Paquete:** Encabezamiento e información del paquete.

Teniendo en cuenta que estos aspectos son imprescindibles para lograr el control de una red determinada, el módulo propuesto, a través de las configuraciones de los logs podrá brindar dichas informaciones que serán almacenadas con el objetivo de controlar el tráfico entrante y saliente de la red perimetral.

Arquitecturas del cortafuego

Un cortafuego puede ser utilizados en diferentes entornos de red, debido a las arquitectura que este posee. Dichas arquitecturas se muestran a continuación [1]:

Dual-Homed Host: Consiste en un ordenador Dual-Homed Host el cual se sitúa entre la red interna a proteger y la red externa. Esta arquitectura se construye alrededor del ordenador Dual-Homed-Host, el cual es un ordenador que tiene al menos dos interfaces de red y es capaz de enrutar paquetes desde una de red hacia otra. Si se implementa una arquitectura Dual-Homed Host se restringe esta función de enrutamiento. Lo que hace que los paquetes de una red no se conectan directamente a la otra. Los sistemas dentro del cortafuego pueden comunicarse con el Dual-Homed Host, y los sistemas fuera del cortafuego (de internet) puede comunicarse con el Dual-Homed Host, pero estos sistemas no pueden comunicarse entre ellos. El tráfico IP está completamente bloqueado. Todo tráfico hacia fuera de la red local lo debe originar el cortafuego.

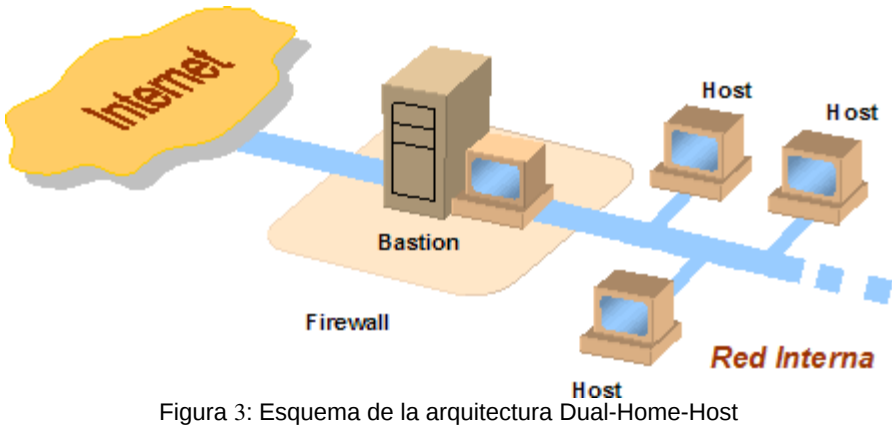


Figura 3: Esquema de la arquitectura Dual-Home-Host

Screened Host: En este caso se combina un *router* con un host bastión y el principal nivel de seguridad proviene del filtrado de paquetes. El bastión, es el único sistema accesible desde el exterior, en el cual se ejecuta el Proxy de aplicaciones y en el *choke* se filtran los paquetes considerados peligrosos y sólo se permiten un número reducido de servicios.

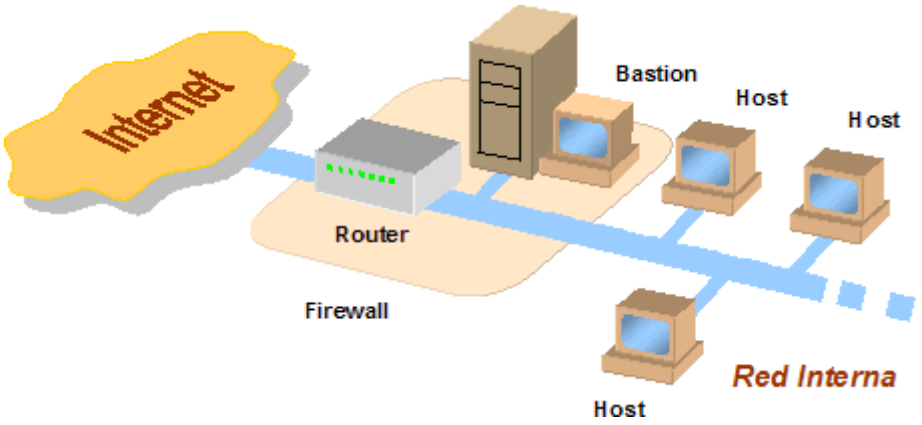


Figura 4. Esquema de la arquitectura Screened Host

Screened Subnet (DMZ): Se intenta aislar la máquina más atacada y vulnerable del cortafuego, el Nodo Bastión. Para ello se establece una Zona Desmilitarizada (DMZ) de forma tal que si una persona no autorizada accede a esta máquina no consiga el acceso total a la subred protegida. En este esquema se utilizan dos routers: uno exterior y otro interior. El Router exterior tiene la misión de bloquear el tráfico no deseado en ambos sentidos: hacia la red interna y hacia la red externa. El Router interior hace lo mismo con la red interna y la DMZ (zona entre el Router externo y el interno). Es posible definir varios niveles de DMZ agregando más routers, pero destacando que las reglas aplicadas a cada uno deben ser distintas ya que en caso contrario los niveles se simplificarían a uno solo.

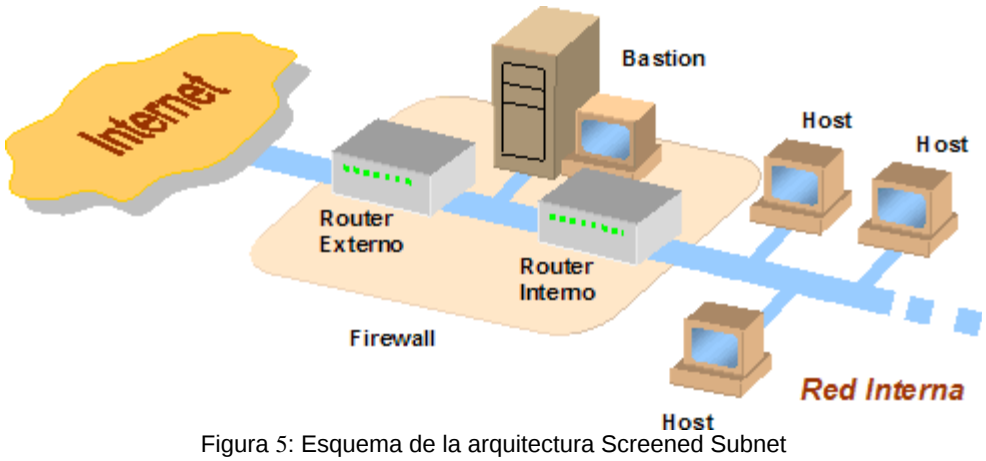


Figura 5: Esquema de la arquitectura Screened Subnet

Netfilter

Netfilter es un framework disponible en el núcleo Linux que permite interceptar y manipular paquetes de red. Dicho framework permite realizar el manejo de paquetes en diferentes estados del procesamiento. Netfilter es también el nombre que recibe el proyecto que se encarga de ofrecer herramientas libres para cortafuegos basados en GNU/Linux. El componente más popular construido sobre Netfilter es iptables [5].

Iptables

Iptables es una herramienta de cortafuego de espacio de usuario que permite no solamente filtrar paquetes, sino también realizar traducción de direcciones de red (NAT) para IPv4 o mantener registros de log. Iptables es el nombre de la herramienta de espacio de usuario mediante la cual el administrador a través de reglas de configuración puede definir políticas de filtrado del tráfico que circula por la red [6].

Para el desarrollo de la aplicación propuesta se usará Iptables teniendo en cuenta que el mismo cuenta con una gama amplia de usuarios debido a que este, con respecto a su antecesor ipchains es mucho más integral. Con la aparición del framework Netfilter, iptables mejoró los conceptos asociados a las estructuras de las reglas, se puede citar el ejemplo de ipchains que incorporó el concepto de cadena e iptables agrego el concepto de tablas. Otro aspecto a tener en cuenta es las configuraciones de las reglas, que a pesar de ser difíciles de establecer, un usuario avanzado en el tema, puede llegar a crear una sólida barrera protectora.

Regla de iptables

Regla: es un conjunto de comparaciones y un único objetivo o salto a una cadena definida por el usuario. Si el paquete analizado cumple las comparaciones, se realiza la acción asociada al objetivo o al salto.

Acciones

Acciones asociadas al objetivo o al salto [7]

- **-A** añade una cadena, la opción -i define una interfaz de tráfico entrante
- **-o** define una interfaz para tráfico saliente
- **-j** establece una regla de destino del tráfico, que puede ser ACCEPT, DROP o REJECT. La
- **-m** define que se aplica la regla si hay una coincidencia específica
- **--state** define una lista separada por comas de distinto tipos de estados de las conexiones (INVALID, ESTABLISHED, NEW, RELATED).
- **--to-source** define que IP reportar al tráfico externo
- **-s** define tráfico de origen
- **-d** define tráfico de destino
- **--source-port** define el puerto desde el que se origina la conexión
- **--destination-port** define el puerto hacia el que se dirige la conexión
- **-t** tabla a utilizar, pueden ser nat, filter, mangle o raw.

Cadenas de iptables

Cadena: es el conjunto de reglas asociados con un determinado tipo de filtro. Iptables también dispone de la posibilidad de que un usuario pueda definir sus propias cadenas y asignarle un nombre. Los nombres de cadenas predefinidos son INPUT, OUTPUT y FORWARD [7].

Cadenas predefinidas de iptables

- INPUT (ENTRADA): Todos los paquetes destinados al cortafuego, recorren esta cadena.
- OUTPUT (SALIDA) Todos los paquetes creados por el cortafuego recorren esta cadena.
- FORWARD (REDIRECCIÓN): Todos los paquetes que meramente pasan por el cortafuego para ser encaminados a sus destinos recorren esta cadena.

Tablas predefinidas de iptables

Iptables posee las siguientes tablas por defecto [7]:

Filter: esta tabla es usada para filtrar paquetes, por ejemplo con los objetivos ACCEPT (Paquete

aceptado), DROP (Paquete rechazado sin notificación) y REJECT (Paquete rechazado y se envía una notificación a través del protocolo ICMP). Cadenas predefinidas: FORWARD, INPUT y OUTPUT.

Nat: Esta tabla permite realizar traducción de direcciones, pudiendo traducir tanto origen como destino. En este caso solo el primer paquete de la conexión pasa por la tabla, después todos los paquetes de la conexión son tratados de la misma forma. Los objetivos o parámetros válidos en esta tabla son SNAT, DNAT, MASQUERADE (similar a SNAT con IP dinámica) o REDIRECT. Cadenas predefinidas: PREROUTING, POSTROUTING y OUTPUT.

- **SNAT (*Source Network Address Translation o* Traducción de direcciones de red de origen)** para traducir direcciones origen. Su uso común es permitir que una red con IPs privadas pueda acceder a internet a través de una IP pública.
- **DNAT (*Destination Network Address Translation*** o **traducción de direcciones de red de destino)** está diseñado para convertir direcciones de destino. Este tipo de traducción nos permite situar distintos servicios en distintos servidores compartiendo una dirección IP.
- **MASQUERADE:** Se utiliza cuando la dirección ip pública que sustituye a la ip de origen es dinámica.
- **REDIRECT:** Es utilizado cuando se desea cambiar o redireccionar los puertos.

Mangle: Esta tabla está diseñada para manipular paquetes. Los siguientes parámetros son solo aplicables en esta tabla: TOS, TTL. Con ellos podemos modificar los paquetes o marcarlos para luego realizar QoS o control de seguridad. Cadenas predefinidas: PREROUTING, POSTROUTING, OUTPUT, INPUT y FORWARD.

Interfaces gráficas para la administración de cortafuego.

La interfaz gráfica de usuario o GUI es un programa informático que actúa como interfaz de usuario utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de un ordenador [3].

Una interfaz fácil de utilizar y con un número mínimo de opciones de configuración reduce la posibilidad de que se produzcan errores de administración. Naturalmente, un número menor de opciones de configuración puede significar también menor flexibilidad de configuración.

Tipos de interfaces de administración de cortafuegos

Existen tres clases de interfaz del administrador de cortafuego [3]:

La interfaz basada en ficheros de texto: es la de uso más extendido en lo que respecta a los cortafuego de elaboración propia. Este tipo de interfaces permiten al administrador editar un archivo específico donde puede introducir parámetros de configuración específicos. Se trata de la interfaz más común para los administradores de los sistemas UNIX tradicionales, dado que ofrece una interfaz de control a bajo nivel con los mecanismos del cortafuego. La desventaja de dicho control a bajo nivel es que resulta mucho más fácil cometer errores, ya que, al editar un fichero, pueden producirse errores de escritura u otros errores técnicos que, en un sistema basado en menús, es menos probable que ocurran.

La interfaz de administrador basada en menús de texto: presenta un menú basado en texto que reduce la probabilidad de producirse errores pero que proporciona menor capacidad de control para el administrador. Sin embargo, la posibilidad de error no queda totalmente excluida, dado que el administrador no siempre puede ver el efecto de algunos cambios.

Principales Cortafuego y herramientas de administración en GNU/Linux

En este acápite se abordará acerca de los principales cortafuego y herramientas de administración, con el objetivo de seleccionar las características más relevantes que pudieran ser importantes a la hora de diseñar el módulo de HMAST. Para ayudar este proceso se hará una breve comparación de los mismos, con el objetivo de obtener información positiva que pueda ayudar a implementar las funcionalidades del módulo, basándose en interfaz gráfica y principales características.

IPCop

IPCop: es para usuarios de pequeñas oficinas y oficinas domésticas. Esta es una distribución Linux firewall, que requiere una PC independiente de baja potencia para ejecutar el software.

- **Última versión estable:** 1.4.21
- **Tipos de Usuarios:** Usuarios de pequeñas oficinas y oficinas domésticas
- **Principal funcionalidad:** mejorar el rendimiento del navegador web, manteniendo una cierta información de uso frecuente, filtrado de paquetes y asignar ancho de banda fija a cada puesto de trabajo.
- **Licencia:** Licencia Pública General

Teniendo en cuenta los aspectos antes mencionados IPCop posee una interfaz gráfica la cual puede ayudar al diseño de la aplicación a desarrollar.

Shorewall

Shorewall [8]**:** Regula los paquetes de entrada y salida de las computadoras que viajan a través de la red, también se define como un cortafuego y a su vez como una puerta de enlace con sus respectivos requisitos de las entradas y salidas de paquetes.

- **Última versión estable:** 4.5.9.3
- **Tipos de Usuarios:** Usuarios expertos
- **Principal funcionalidad:** la mayor parte de su fuerza reside en su capacidad de trabajar con "zonas", como la DMZ o una zona de red.
- **Licencia:** GPLv2 Licencia Pública General versión 2
- **Interfaz gráfica:** no posee interfaz gráfica.

Shorewall está basado en iptables y posee funcionalidades que pueden ser utilizadas en el diseño del módulo de HMAST como es el caso de la configuración detallada de reglas, este lo hace a través de archivos de configuración de texto plano y aunque los mismos están bien documentados se necesita un alto nivel de abstracción para realizar una configuración avanzada.

UFW

UFW (*Uncomplicated Firewall* o Firewall sin complicaciones**)** [9]: es un programa de línea de comandos que ayuda a administrar el cortafuego netfilter iptables. Esto proporciona algunos comandos simples para manejar iptables.

- **Última versión estable:** 12.04
- **Tipos de Usuarios:** puede ser utilizado por usuarios de bajo nivel.
- **Principal funcionalidad:** gestiona el tráfico entrante y saliente de paquetes, así como configuraciones de rangos de IPS y de puertos.
- **Licencia:** Licencia Pública General
- **Interfaz gráfica:** interfaz gráfica intuitiva.

Teniendo en cuenta los aspectos anteriores y sus principales funcionalidades, UFW es un cortafuego muy completo, pero es difícil para ser utilizado por usuarios de bajo nivel, problema que fue solucionado gracias a su interfaz gráfica y amigable (Gufw), la cual permite realizar configuraciones sencillas y avanzadas con facilidad.

Se pueden tomar algunas de sus características y funcionalidades para ser utilizadas en el desarrollo del módulo propuesto, como es el caso del asistente de configuración el cual se debe tener en cuenta para diseñar interfaces gráficas, para las configuraciones reglas de maneras simples y sencillas.

Zentyal

Zentyal (*anteriormente conocido como eBox Platform*) [10]:es un servidor de red unificado de código abierto.

- **Última versión estable:** 3.0
- **Tipos de Usuarios:** Usuarios de bajo, mediano y alto nivel.
- **Principal funcionalidad:** Filtrado de paquetes, NAT, Tráfico de redirección de puertos**.**
- **Licencia:** Licencia Publica General de GNU aunque posee algunas licencias privadas**.**
- **Interfaz gráfica:** interfaz gráfica amigable y muy intuitiva.

Zentyal es un servidor de red integral el cual posee un módulo de cortafuego muy completo e intuitivo, permite configurar reglas de iptables a través de entornos o plantillas predefinidas guiadas por imágenes que orientan al usuario según sus necesidades, por lo que se convierte en un ejemplo a seguir tanto para la configuración de reglas de filtrado de paquete, NAT y redirección de puerto como para la interacción de su interfaz gráfica.

Debido a esto Zentyal puede ser muy útil para tomar ideas en cuanto al diseño y las configuraciones de las reglas a través de su interfaz gráfica. No obstante, para la implementación del módulo, se tendrá en cuenta la interfaz de sus configuraciones de reglas de filtrado a través de plantillas.

HMAST

HMAST es un sistema base que permite administrar los servidores de forma remota, en los cuales se tiene las funcionalidades necesarias para administrar los usuarios, las tareas programadas y los servicios, entre otros.

Arquitectura

La arquitectura que presenta la herramienta HMAST propone el diseño de una arquitectura N-Capas basada en el dominio, compuesta por cinco capas las cuales son descritas a continuación.

La **capa de presentación** es la que presenta al usuario los conceptos de negocio mediante una interfaz de usuario. La **capa de aplicación** realiza las llamadas a los servicios de la capa inferior y tiene la responsabilidad de adaptar la información que le llega, a los requerimientos de los servicios de dominio. La **capa de dominio** es responsable de las validaciones, define las interfaces de persistencia a datos (contratos de repositorio) pero no los implementa y está compuesta por entidades del dominio que representan objetos del dominio y están definidas fundamentalmente por su identidad, servicios de dominio que contienen la lógica que trata a las entidades como un todo y los contratos de repositorios que son interfaces que especifican las operaciones que deben implementar los repositorios. La **capa de persistencia** es responsable de contener el código necesario para persistir los datos, contiene como componente los repositorios que son clases que implementan los contratos de repositorios definidos en la capa de dominio. Finalmente, la **capa Infraestructura Trasnversal** que es responsable de promover la reutilización de código, ella albergará las operaciones de seguridad, logging, monitorio del sistema,

mecanismos de persistencia reutilizables, validadores genéricos, en fin todas aquellas operaciones que se puedan llamar desde otras capas.

Consideraciones para implementar un módulo para HMAST.

- La lógica de Aplicación no deberá incluir ninguna lógica del Dominio, solo tareas de coordinaciones relativas a requerimientos técnicos de la aplicación, como conversiones de formatos de datos de entrada a entidades del Dominio, llamadas a componentes de Infraestructura para que realicen tareas complementarias.
- Se debe garantizar que no viajen hacia y desde la capa de presentación objetos de dominio, en su lugar deben viajar objetos DTO (Data Object Transfer).
- Las entidades solo pueden tener dependencias de componentes de la capa de dominio.
- Las clases de servicios deben ser las únicas responsables (vías de acceso) de acceder a los repositorios, no se puede implementar código de persistencia a datos en la capa de dominio.
- Solo se puede acceder a la información almacenada en los servidores haciendo uso de los repositorios.
- Es importante que todo el código reutilizable por más de un repositorio se ponga a disposición de todos en la capa de infraestructura transversal.

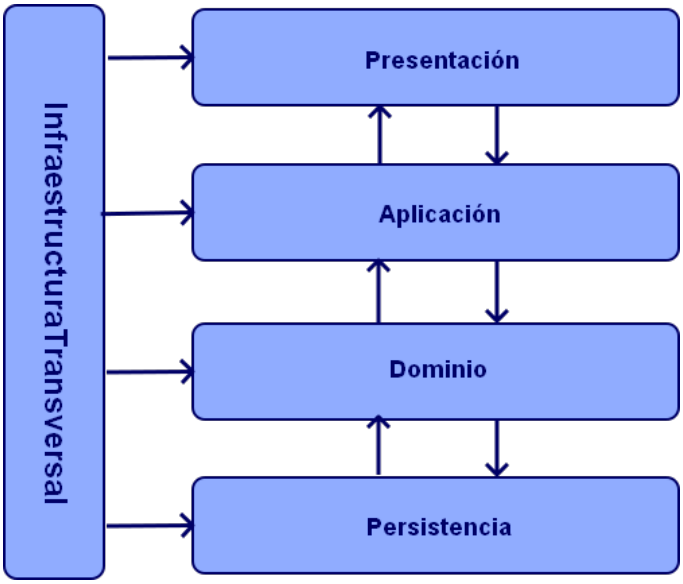


Figura 6: Esquema de la arquitectura de HMAST

Herramientas y tecnologías de desarrollo

Es necesario destacar que las herramientas y las tecnologías a utilizar para el desarrollo del módulo propuesto, fueron seleccionadas para el diseño e implementación de la base de HMAST. Para el diseño e implementación del módulo de cortafuegos se continuará con la selección de las mismas teniendo en cuenta fundamentalmente la petición del cliente.

IDE de Programación

NetBeans: El IDE NetBeans es una herramienta pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para varios lenguajes de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto de código abierto y gratuito sin restricciones de uso [13].

Visual Paradigm

Visual Paradigm es una herramienta CASE: Propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación.

Lenguaje de Programación

Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Java: es un lenguaje de programación y la primera plataforma informática creada por Sun Microsystems en 1995. Es la tecnología subyacente que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios.

Metodología SXP

Para la realización del módulo se empleará la siguiente metodología de desarrollo [11]:

SXP. Compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo.

Metodologías que la conforman:

- **SCRUM** es una forma de gestionar un equipo de manera que trabaje de forma eficiente y de tener siempre medidos los progresos, de forma que sepamos por dónde andamos.
- **XP** más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno

de los requisitos para llegar el éxito del proyecto.

Fases de SXP

- **Planificación-Definición** donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- **Desarrollo**, es donde se realiza la implementación del sistema hasta que esté listo para ser entregado.
- **Entrega**, puesta en marcha.
- **Mantenimiento**, donde se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, definición de las Historias de Usuario, diseño, implementación, pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, y existe una refactorización continua, lo que nos permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

A modo de resumen **SXP** está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo.

Conclusiones parciales del capítulo

En el desarrollo de este capítulo fueron analizados los principales cortafuegos, así como herramientas de administración del mismo mediante el estudio realizado, teniendo en cuenta algunos aspectos esenciales en el cual el autor decide que ninguna de las aplicaciones estudiadas pueden adaptarse a las condiciones de HMAST debido a que poseen arquitecturas diferentes, pero si se pueden obtener beneficios de muchas de ellas en cuanto a la estructura, interfaz y configuración. Una de las herramientas que más se destaca en este sentido es Zentyal, debido a su amigable interfaz web y la intuitiva forma de realizar las configuraciones. Por último de estableció que las herramientas a utilizar en el transcurso de proceso de desarrollo, fueran las anteriormente seleccionadas por políticas del proyecto así como las tecnologías a utilizar.

Capítulo 2: Análisis y diseño

Breve descripción del capítulo

En este capítulo se aborda sobre de la descripción del módulo. También se tiene en cuenta las funcionalidades que debe cumplir el sistema, descritas mediante las Historias de Usuario asegurando así una mejor comprensión por parte del desarrollador para la implementación del módulo. Además, se explica la arquitectura y los patrones de diseño que se utilizarán con el fin de tener una estructura y estilo estándar en la confección de dicho módulo.

Propuesta del módulo a desarrollar

Después de realizar un estudio de las principales herramientas para la administración de cortafuego en GNU/Linux, se llega a la conclusión de que se hace necesario desarrollar un módulo para la herramienta HMAST que permita administrar el servicio de cortafuego.

Para darle solución al problema planteado, el presente trabajo de diploma propone desarrollar un módulo de cortafuego mediante el uso de iptables el cual permita controlar satisfactoriamente el acceso a las redes de la organización a la cual sea destinada dicha aplicación.

Arquitectura del módulo

La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea, así como sus interfaces y la comunicación entre ellos. Por consenso y directivas del proyecto se decidió adoptar la arquitectura de HMAST para cada uno de sus módulos ver *Figura (6)*.

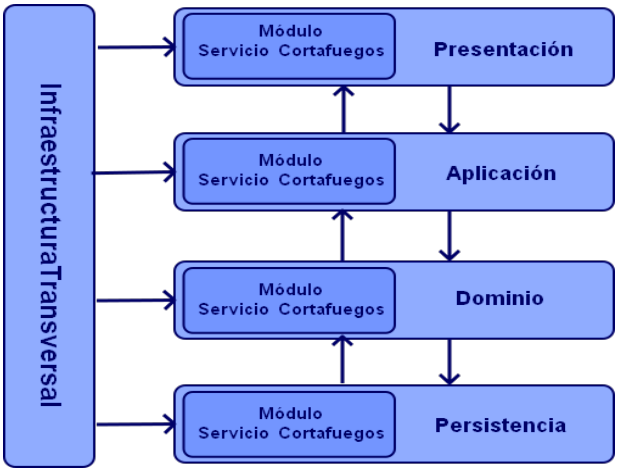


Figura 7: Esquema de la arquitectura del módulo de cortafuego para HMAST

Roles

Rol	Responsabilidad	Nombre y Apellidos
Gerente	Es el responsable de tomar las decisiones finales, participa en la definición de objetivos y requerimientos. Tiene la responsabilidad de controlar el progreso del software.	Pablo Soria Acosta
Cliente	Asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración, participa en la concepción inicial del sistema. Contribuye a definir las historias de usuario y los casos de prueba de aceptación.	Yoandy Pérez Villazon
Miembros del Equipo		
Programador	Define las tareas de ingeniería. Produce el código del sistema. Selecciona el estándar de programación. Confecciona los Manuales de Usuario y de desarrollo.	José Carlos Miranda Azcuy
Analista	Escribe la concepción del sistema y las historias de usuario. Crea el Modelo de historia de usuario del negocio y la LRP. Asigna la prioridad a las Historias de Usuario.	José Carlos Miranda Azcuy
Diseñador de interfaz	Encargado del diseño del sistema, así como el de los prototipos de interfaces, supervisa el proceso de construcción.	José Carlos Miranda Azcuy
Probador	Escribe los casos de prueba de aceptación. Ejecuta las pruebas, es responsable de las herramientas de soporte para pruebas.	José Carlos Miranda Azcuy
Consultor	Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas, además aportan ideas y experiencias para el beneficio del sistema en desarrollo.	Ing. Nelio Véliz Pedraza

Tabla 1: Roles

Requisitos funcionales y no funcionales

Requisitos de integración.

Durante el encuentro con el cliente, se llegó al acuerdo de qué requisitos deben ser cumplidos por el módulo de cortafuegos, para HMAST:

- El módulo se debe ajustar a la arquitectura definida en la herramienta HMAST.
- Se debe desarrollar con tecnologías libres.
- La comunicación entre las capas se debe realizar a través de clases de interfaces.
- Los objetos se deben acceder a través de instancias mediante inyección de dependencias.

Requerimientos funcionales: Describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del mismo y del enfoque general tomado por la organización al redactar los requerimientos. Cuando se expresan como requerimientos del usuario, habitualmente se describen de forma abstracta, sin embargo, los requisitos funcionales del sistema describen con detalle la función de este, sus entradas y salidas, excepciones, etc. [12].

Requerimientos no funcionales: Como su nombre indica, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema [12].

En el sistema, debido a la metodología a utilizar los requerimientos ya sean funcionales o no, se reflejan en la **Lista de Reserva del Producto (LRP)** donde se le asigna una prioridad, y la estimación de cada uno de los requisitos identificados.

Lista de Reserva del Producto (LRP)

Ítem *	Descripción	Estimación	Estimado por
Prioridad Muy Alta			
1	Adicionar regla de iptables	0,4	*José Carlos Miranda Azcuy*
2	Eliminar regla de iptables	0,3	
3	Modificar regla de iptables	0,4	
4	Mostrar lista de regla configuradas	0,3	
Prioridad Alta			
5	Mostrar el estado de cortafuego	0,3	
6	Iniciar cortafuego	0,3	
7	Detener cortafuego	0,3	
8	Reiniciar cortafuego	0,3	
9	Salvar configuraciones existentes en directorios locales	0,3	
10	Cargar las configuraciones realizadas.	0.3	
11	Instalar el servicio de cortafuego	0.3	
12	Desinstalar el servicio de cortafuego	0,3	
Requisitos no funcionales			
Software			
1	Sistema operativo: Nova, Ubuntu o Debian.		
2	Servidor Web: Apache-Tomcat		
3	Framework de desarrollo: Spring		
4	Lenguaje de programación: Java		
Hardware			
5	El servidor Web donde está hostiado la aplicación debe tener 512 MB de RAM y 60 GB de disco duro como mínimo.		

Tabla 2: Lista de Reserva del Producto (LRP).

Historias de Usuarios:

Las historias de usuario son una representación de los requerimientos de software, escrito en una o dos frases utilizando el lenguaje común de usuario. Estos son utilizados en las metodologías de desarrollos ágiles en este caso SXP para las especificaciones de requisitos y las pruebas de validación.

Historia de Usuario	
Número: HMAST_Firewall_1	**Nombre Historia de Usuario:** Gestionar reglas de filtrado.
Modificación de Historia de Usuario: 0	
Usuario: José Carlos Mirandas Azcuy	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 0.2
Riesgo en desarrollo: Alta	**Puntos Reales:** 0.2
Descripción: Se gestionan las reglas de filtrado, es decir, el tráfico de paquetes entrantes, salientes y que pasan a través del cortafuegos, en este caso se podrá adicionar, mostrar, modificar y eliminar una regla de filtrado.	

Observación:

Escenarios:

E1- Paquetes entrantes al cortafuego: Cuando se seleccione la opción **paquetes que entran al cortafuego** se mostrará un asistente de configuración, en el cual el usuario introduce los parámetros para adicionar dicha regla de filtrado. Estos parámetros son:

Tipo de regla: el usuario selecciona el tipo de regla que desea, en este caso se refiere a la acción que hará el paquete si coincide con los parámetros de configuración que el usuario especificó. Es obligatorio seleccionar una de estas opciones.

- **Aceptar:** Paquete aceptado.
- **Descartar:** Paquete rechazado. Sin notificación.
- **Rechazar**: Paquete rechazado. Se envía notificación a través del protocolo ICMP.
- **Registrar**: Este objetivo funciona para registrar los paquetes que pasan por el cortafuego, controlando determinada información de los mismos.

Dirección de origen: en este caso el usuario tiene tres opciones para definir su dirección de origen o en el mejor de los casos, puede seleccionar las tres opciones, las cuales son:

- **Interfaz de entrada:** Interfaz de red de entrada del servidor (eth0, eth1, etc.).

Este campo solo acepta las interfaces que se definan en el servidor, es decir, en caso de introducir una interfaz que él no sea igual a la que posee el servidor, se mostrará un mensaje de error.

- **Dirección Ip/máscara:** Dirección IP y máscara del dispositivo que originó la conexión.

En este campo el usuario solo podrá introducir cuatro números separados por un **“.”** y en caso de la máscara, solo el usuario tendrá la posibilidad de seleccionar el número.

- **Mac:** Dirección MAC del dispositivo que originó la conexión.

En este campo el usuario solo podrá adicionar seis pares de letra y dígitos separados por “**:”**, las letras solo pueden ser de la (a – f).

Dirección de destino: en este caso el usuario tiene una opción para definir su dirección de destino la cual es:

- **Dirección IP/Máscara:** el mismo caso del campo (Dirección IP/Máscara) de la opción **Dirección de origen**.

Protocolos y Puertos: En este campo el usuario tiene la posibilidad de seleccionar el protocolo que desea con su puerto asociado, ya sea de origen y destino.

- **Protocolo:** el usuario puede seleccionar el protocolo que se definió ya en el sistema.

En este campo solo son válidos los protocolos **TCP**, **UDP**, **ICMP** y **ALL**, para especificar todos los protocolos

Puerto origen: El usuario tiene la posibilidad de introducir un puerto simple, o un rango de puerto separado por **":".**

En este caso solo son válidos los puertos que se encuentran en el rango de 0-65535, no se permiten letras en este campo.

- **Puerto destino:** igual al campo puerto de origen.

Descripción: En este campo el usuario tiene la posibilidad de escribir una breve descripción de la regla que ha configurado anteriormente.

Este campo solo es válido siempre y cuando no exceda de los 256 caracteres.

E2- Paquetes que se generan y salen del cortafuegos: se mostrará un asistente de configuración, en el cual el usuario introduce los parámetros para adicionar dicha regla de filtrado. Estos parámetros son:

- **Dirección destino** en esta vista se agrega el campo **Interfaz de salida** que tiene la misma validación que el campo **Interfaz de entrada** de la ventana **Dirección de origen,** en el asistente para adicionar reglas de paquetes entrantes al cortafuego.
- No posee Dirección MAC
- El resto de los campos de este asistente de configuración es igual al anterior excepto lo antes explicado.

E3- Paquetes que pasan a través del cortafuegos: se mostrará un asistente de configuración en el cual el usuario introduce los parámetros para adicionar dicha regla de filtrado. Estos parámetros son:

- En la ventana **Dirección origen** y **Dirección destino** se agregan los campos **Interfaz de origen** e **interfaz de salida** respectivamente que tiene la misma validación que el campo **Interfaz de entrada** en la ventana **Dirección de origen,** e **Interfaz de salida** en la ventana

Dirección de destino respectivamente.

- El resto de los campos de este asistente de configuración es igual al anterior excepto los anteriormente explicados.

Mostrar reglas de filtrado:

Al adicionar reglas en cada uno de los escenarios antes expuestos se listarán dichas reglas en la parte inferior del panel que se muestra al seleccionar el escenario deseado. Para listar dichas reglas se mostraran los mismos datos introducidos por el usuario en forma de tabla.

Modificar regla de filtrado:

Luego de seleccionar la regla que se desea modificar, se muestra el mismo asistente de configuración cuando el usuario desea adicionar dicha regla, teniendo en cuenta las mismas validaciones de los campos antes expuestas en el escenario de adicionar una regla de filtrado. Se debe tener en cuenta para poder modificar, que solo es válido si se selecciona una regla en el listado de regla correspondiente.

Eliminar regla de filtrado:

Para eliminar reglas de filtrado el usuario debe seleccionar en el listado las reglas que desea eliminar y luego seleccionar el botón correspondiente a dicha acción.

Prototipos: ver anexos 1-19

Tabla 3: Gestionar reglas de filtrado

Historia de Usuario	
Número: HMAST_Firewall_2	**Nombre Historia de Usuario:** Gestionar reglas NAT.
Modificación de Historia de Usuario: 0	
Usuario: José Carlos Mirandas Azcuy	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 0.2
Riesgo en desarrollo: Alta	**Puntos Reales:** 0.2

Descripción: Este escenario está relacionado con la traducción de direcciones de red, usualmente utilizado cuando se quiere brindar un servicio a una solicitud hecha desde el exterior (DNAT) o cuando queremos establecer una conexión con un ordenador del exterior (SNAT).

En este caso se podrá adicionar, mostrar, modificar y eliminar una regla NAT o traducción de direcciones de red.

Observación:

Adicionar reglas de paquetes que aún no han sido enrutados: se mostrará un asistente de configuración, en el cual el usuario introduce los parámetros para adicionar dicha regla NAT. Estos parámetros son:

Tipo de regla: el usuario selecciona el tipo de regla que desea, en este caso se refiere al destino que tendrá el paquete si coincide con los parámetros de configuración. Es obligatorio seleccionar una de estas opciones.

- **DNAT:** Traducción de la dirección de red de destino. Esta opción consiste en cambiar la dirección de destino ya predefinida por una nueva dirección de destino. Para esto al seleccionar la opción DNAT, el usuario podrá introducir una nueva dirección o rango de IP y un puerto o rango de puerto de destino.

En este caso el usuario puede introducir una rango de IP separado por **"-"** o una dirección IP. Luego de adicionar la dirección IP puede o no adicionar un puerto o rango de puertos. Para este caso se tiene en cuenta que el Ip será válido si solo se adicionan cuatro números separados por **"."** y si cada número es menor o igual que 255, y en el caso de los puertos el número introducido este entre 0-65535.

- **Redireccionar:** Esta opción permite al usuario cambiar la dirección de un paquete, es similar a la opción DNAT, excepto que en este caso solo se pueden redireccionar puertos.

En este caso el usuario solo podrá introducir, puertos simples separados por **","** o rangos de puertos separados por **"-"**, ya seleccionada la opción no se permite dejar los campos en blanco o introducir letras.

- **Dirección de origen:** en este caso el usuario tiene tres opciones para definir su dirección

de origen o en el mejor de los casos, puede seleccionar las tres opciones, las cuales son:

Interfaz de entrada: Interfaz de red de entrada del servidor (eth0, eth1, etc.).

Este campo solo acepta las interfaces que se definan en el servidor, es decir, en caso de introducir una interfaz que no sea las que posee el servidor, se mostrará un mensaje de error.

- **Dirección Ip/máscara:** Dirección IP y máscara del dispositivo que originó la conexión.

En este campo el usuario solo podrá introducir cuatro números separados por un **"."** Y cada número introducido tiene que ser menor o igual que 255. En caso de la máscara, solo el usuario tendrá la posibilidad de seleccionar el número.

- **Mac:** Dirección MAC del dispositivo que originó la conexión.

En este campo el usuario solo podrá adicionar seis pares de letra y dígitos separados por **":"**, las letras solo pueden ser de la (a – f).

- **Dirección de destino:** en este caso el usuario tiene una opción para definir su dirección de destino la cual es:
- **Dirección IP/Máscara:** el mismo caso del campo (Dirección IP/Máscara) de la opción **Dirección de origen**.
- **Protocolo:** el usuario puede seleccionar el protocolo que se definió ya en el sistema.

En este campo solo son válidos los protocolos **TCP**, **UDP**, **ICMP** y **ALL**, para especificar todos los protocolos

- **Puerto origen:** El usuario tiene la posibilidad de introducir un puerto simple o un rangos de puertos por **":"**.

En este caso solo son válidos los puertos que se encuentran en el rango de 0-65535, no se permiten letras en este campo.

- **Puerto destino:** igual al campo puerto de origen.
- **Descripción:** En este campo el usuario tiene la posibilidad de escribir una breve descripción de la regla que ha configurado anteriormente.
- Este campo solo es válido siempre y cuando no exceda de los 256 caracteres.

Listar reglas de traducción de red:

Es similar a listar reglas de filtrado o sea en la vista se mostrará el campo correspondiente a la lista donde se mostraran los datos previamente adicionados en forma de tabla, mostrando exactamente los mimos datos que el usuario introduce en el asistente.

Modificar reglas de traducción de red:

El usuario selecciona en la lista, la regla que será modificada, y luego de esto selecciona el botón correspondiente a dicha funcionalidad. Seguidamente se muestra el asistente de configuración correspondiente con el escenario perteneciente a la regla seleccionada.

Eliminar regla de traducción de red:

El usuario marca en la lista las reglas que desea eliminar y pasado esto seleccionar el botón correspondiente a dicha funcionalidad.

Adicionar regla de paquetes ya enrutados: se mostrará un asistente de configuración, en el cual el usuario introduce los parámetros para adicionar dicha regla NAT. Estos parámetros son:

Tipo de regla: el usuario selecciona el tipo de regla que desea, en este caso se refiere al destino que tendrá el paquete si coincide con los parámetros de configuración. Es obligatorio seleccionar una de estas opciones.

- **SNAT:** Traducción de la dirección de red de origen. Esta opción consiste en cambiar la dirección de origen ya predefinida, por una nueva dirección de origen. Para esto al seleccionar la opción SNAT, el usuario podrá introducir una nueva dirección o rango de IP y un puerto o rango de puerto de destino.

En este caso el usuario puede introducir un rango de IP separado por **"-"** o una dirección IP. Luego de adicionar la dirección Ip se puede o no adicionar un puerto o rango de puertos. Para este caso se tiene en cuenta que el IP será válido si adicionan cuatro número separados por **"."** y si cada número es menor o igual que 255, y en el caso de los puertos los números introducidos deben estar entre 0-65535.

- **Enmascarar:** Esta opción es similar a **SNAT**, solo que esta es utilizada, cuando no conocemos exactamente la dirección de origen. Esto se pone en evidencia cuando se tiene direcciones dinámicas es decir servicio DHCP y por lo tanto no necesita que se le especifique una nueva dirección, solo necesita un puerto o rango de puerto.

En este caso el usuario solo podrá introducir, un puerto simple o un rango de puertos separados por **"-"**, ya seleccionada la opción no se permite dejar los campos vacíos o introducir letras.

- No posee I**nterfaz de entrada**, posee interfaz de salida
- El resto de los parámetros se adicionan similar a DNAT.

Listar reglas de traducción de red:

Es similar a listar reglas de filtrado o sea en la vista se mostrará el campo correspondiente a la lista donde se mostrarán los datos previamente adicionados en forma de tabla, mostrando exactamente los mismos datos que el usuario ha introducido en el asistente.

Modificar reglas de traducción de red:

El usuario selecciona en la lista, la regla que se modificará, y luego de esto selecciona el botón correspondiente a dicha funcionalidad. Seguidamente se muestra el asistente de configuración correspondiente con el escenario perteneciente a la regla seleccionada.

Eliminar regla de traducción de red: El usuario selecciona en la lista, las reglas que desea eliminar y pasado esto seleccionar el botón correspondiente a dicha funcionalidad.
Prototipos:

Tabla 4: Gestionar reglas NAT

Historia de Usuario Gestionar estado del cortafuego.

Historia de Usuario	
Número: HMAST_Firewall_3	**Nombre Historia de Usuario:** Gestionar estado del cortafuego.
Modificación de Historia de Usuario: 0	
Usuario: José Carlos Mirandas Azcuy	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 0.2
Riesgo en desarrollo: Alta	**Puntos Reales:** 0.2
Descripción: Este escenario está relacionado con la gestión del servicio de cortafuego. En este caso se refiere al proceso de instalación, desinstalación, iniciar, detener y reiniciar el servicio, así como mostrar el estado del mismo.	
Observación: **E1: Instalar el servicio cortafuego.** El usuario tiene la posibilidad de poder instalar el servicio de cortafuegos en caso de no estar instalado en el servidor, para esto solo es necesario seleccionar la opción correspondiente a dicha funcionalidad. **E2: Desinstalar el servicio cortafuego.** El usuario tiene la posibilidad de poder desinstalar el servicio de cortafuegos en caso de estar instalado en el servidor, para esto solo es necesario seleccionar la opción correspondiente con dicha funcionalidad. **E3: Mostrar el estado del servicio cortafuego.** En este caso se refiere a si el cortafuego está detenido, iniciado, el sistema debe poder mostrar al usuario su estado actual. Se mostrará un mensaje indicando el estado del mismo. **E4: Iniciar servicio cortafuego.** Luego de realizar todas las actividades deseadas, el usuario tiene la posibilidad de iniciar el servicio cortafuegos el cual pondrá en marcha las configuraciones establecidas. **E5: Detener servicio cortafuego.** Una vez iniciado el servicio, si el usuario desea puede detener el mismo. En este caso se anularan	

las configuraciones hasta que el servicio sea iniciado nuevamente.
Prototipos:

Tabla 5: Gestionar estado del cortafuego

Historia de Usuario Gestionar configuraciones existentes.

Historia de Usuario	
Número: HMAST_Firewall_4	**Nombre Historia de Usuario:** Gestionar configuraciones existentes.
Modificación de Historia de Usuario: 0	
Usuario: José Carlos Mirandas Azcuy	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 0.2
Riesgo en desarrollo: Alta	**Puntos Reales:** 0.2
Descripción: Este escenario está relacionado con la gestión de las configuraciones existentes. En este caso se refiere al proceso de guardar y cargar las configuraciones realizadas.	
Observación: **E1: Cargar configuraciones.** El sistema en este caso podrá cargar las configuraciones de las reglas que se encuentran en el fichero de configuración, pasado esto el usuario puede realizar normalmente las acciones que desee. **E2: Desinstalar el servicio cortafuego.** El usuario tiene la posibilidad de guardar las configuraciones de reglas en un archivo determinado con fines de ser cargadas en otra ocasión para realizar determinadas operaciones.	
Prototipos:	

Tabla 6: Gestionar estado del cortafuego

Diagrama de Paquetes

Un diagrama de paquetes es aquel que muestra cómo un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones. Los paquetes están organizados con el objetivo de maximizar la coherencia interna de cada paquete y minimizar el acoplamiento externo entre ellos.

Descripción del diagrama de Paquetes del módulo de cortafuego

El diagrama de paquetes del módulo de cortafuego está distribuido en tres paquetes: *Application*, *Domain* y *Persistence* respectivamente. Cada uno de ellos posee un subpaquete con el nombre del módulo (Para más información ver el anexo digital).

En la capa de aplicación (***Application*** en inglés) se encuentra la clase interfaz **IRuleAppService**, que contiene los métodos que serán llamados desde la capa de presentación e implementados en por la clase **RuleAppService**. Dicha clase es la encargada de adaptar la información que le llega desde la interfaz de usuario a los requerimientos de los servicios de dominio. Para acceder a los servicios del dominio se debe recurrir al atributo **IRules** de tipo **IRuleService,** ya que la comunicación entre las capas será a través de las interfaces y usando inyecciones de dependencias. Esta capa de aplicación también contiene el paquete **DTO** el cual contiene clases que representan la información que llega desde la capa de presentación.

La capa de Dominio (***Domain*** en inglés) contiene tres subpaquetes: ***Entity***, ***Service*** y ***RepositoryContracts***. En el paquete **Entity**, se encuentra la clases **Rule** que representa a una regla de iptables. En el paquete **Service** se encuentra la clase **IRuleService** que contiene los métodos que son accedidos desde la capa de **Aplicación** los cuales son implementados por la clase **RuleServices** que es la encargada de las validaciones de los datos antes de ser leídos o escritos en el o los repositorios. Para acceder a los repositorios desde esta clase, se hace a través del atributo **iRuleR** de tipo **IRuleRepository.** En el paquete **RepositoryContracts** se encuentra la clase interfaz **IRuleRepository** la cual tiene definidos los métodos que son accedidos desde la el subpaquete **Service** y que serán implementados por la clase **RuleRepository** contenida en la capa de persistencia, dicha clase es la encargada de gestionar los datos contenidos en los repositorios.

La anterior explicación está relacionada con la HU: Gestionar reglas de filtrado y gestionar reglas Nat, para el caso de las HU restantes el flujo de comunicación entre las capas es el mismo independientemente que contengan funcionalidades diferentes.

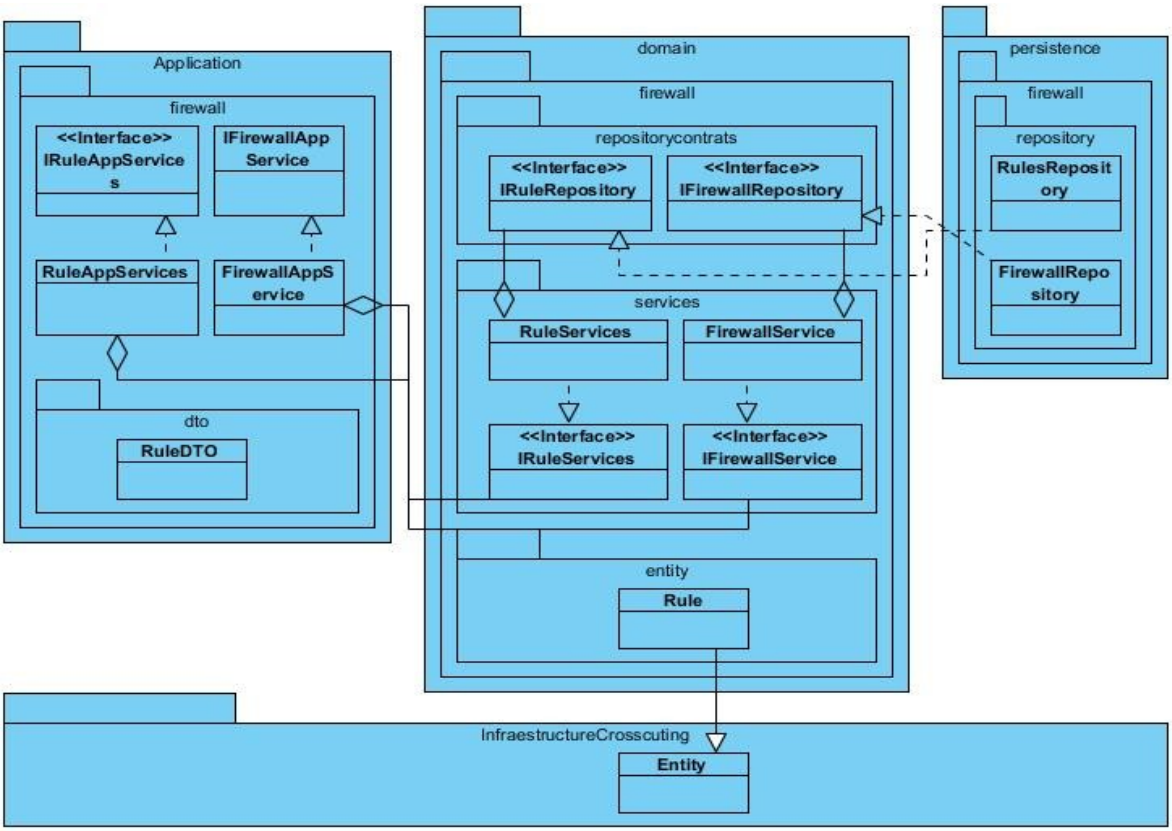


Figura 8: Esquema del diagrama de paquetes del módulo de cortafuego

Patrones de diseño

Un patrón es una *pareja de problema/solución* y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas. Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno y después describe la esencia de la solución a dicho problema, de tal forma que puedas usar dicha solución reiteradamente de distintas maneras.

Patrones GRASP

Patrones generales de software para asignación de responsabilidades (*General Responsibility Assignment Software Patterns* en inglés), no son considerados solo patrones propiamente dichos, sino buenas prácticas de aplicación recomendable en el diseño de *software*.

Experto: A modo de ejemplo, este patrón nos indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo obtendremos un diseño con mayor cohesión y así la información se mantiene

encapsulada (disminución del acoplamiento).

- Con la utilización de este patrón, se hizo posible definir dónde colocar en cada clase del módulo las funcionalidades que necesitan de esa información, dicha clase sería el experto en información.

Creador: El patrón creador nos ayuda a identificar ¿quién debe ser el responsable de la creación de instancias o de nuevos objetos o clases? La instancia deberá ser creada por la clase que tiene la información necesaria para realizar la creación del objeto, o usa directamente las instancias creadas del objeto, o almacena o maneja varias instancias de la clase.

- Este patrón se utilizó para la creación de los objetos y de los objetos que estos contengan como atributo en las diferentes capas del módulo lo cual se realiza a través de la inyección de dependencia.

Bajo acoplamiento: Se basa en la idea de tener las clases lo menos mezcladas posible. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

- En el módulo propuesto el bajo acoplamiento se logra mediante el uso de interfaces utilizando inyecciones de dependencia. En este caso, cada vez que se necesite instanciar un objeto para ser llamado desde otras clases o capas.

Alta cohesión: este patrón se basa en que la información que almacena una clase debe ser coherente y debe estar en la medida de lo posible relacionada con la clase.

- En este caso, las clases que se identificaron por ser muy extensas, fueron divididas con el objetivo de disminuir el nivel de responsabilidad y asegurando la coherencia entre ellas.

Patrones GoF

Los patrones GoF se descubren como una forma indispensable de enfrentarse a la programación a raíz del libro *"Design Patterns Elements of Reusable Software"* de *Erich Gamma, Richard Helm, Ralph Jonson y John Vlissides,* a partir de entonces estos patrones son conocidos como los patrones de la pandilla de los cuatro. El mismo cuenta con tres clasificaciones según su propósito: Creación, Estructurales y de Comportamiento.

- En el módulo propuesto se utilizó el patrón solitario (*singlenton*) el cual permite hacer una única instancia del objeto conexión, garantizando el acceso global a la misma con el mismo objeto en varios momentos y lugares. Con una única instancia del objeto conexión se pueden hacer operaciones sobre el **Logicalserver**.

Plan de iteraciones

Después de realizarse un análisis detallado de las HU y estimado el tiempo para la implementación de las mismas se procede a la realización de la planificación de las etapas de desarrollo del sistema. En esta planificación detallada se dividen las HU por el nivel de prioridad que poseen las mismas, para ser ubicadas en las diferentes etapas de dicho plan. Teniendo en cuenta esto a continuación se muestra la distribución de las HU por las diferentes etapas del Plan de iteraciones.

- **Iteración 1:** En esta primera iteración se planifica la implementación de las Historias de Usuario de alta prioridad, debido a que las mismas son las principales funcionalidades que debe cumplir el sistema. Las Historias de Usuario de alta prioridad son de la 1 a la 8.
- **Iteración 2:** En esta segunda iteración se planifica la implementación de las Historias de Usuarios de mediana prioridad, ya que las mismas, son las funcionalidades que no son imprescindibles para el funcionamiento del sistema. Las Historias de Usuario de mediana prioridad son de la 9 a la 16.

CAPÍTULO 3: Implementación y Prueba

Se describe el proceso realizado para la implementación, se exponen las tareas de ingenierías correspondientes a las Historias de Usuarios. Se desarrolla el código de una manera certificada, en el cual se definen estándares de programación dando como resultado el código fuente de la aplicación a desarrollar. También se realiza la codificación y pruebas a nivel unitario obteniendo un soporte de pruebas de unidad, sin dejar de mencionar las pruebas del módulo desarrollado con el objetivo de documentar el mismo y verificar su correcto funcionamiento.

Tareas de Ingeniería

No	Nombre de la tarea	HU	Tipo de tarea
1	Verificar y probar el funcionamiento de los parámetros válidos para adicionar reglas de filtrado.	1	*Desarrollo*
2	Verificar y probar el funcionamiento de los parámetros válidos para adicionar reglas de traducción de dirección de red.	2	*Desarrollo*
3	Verificar y probar comandos para la creación de los archivos y carpetas necesarios para la instalación del servicio.	3	*Desarrollo*
4	Verificar y probar comandos para la creación del demonio de iptables	3	*Desarrollo*
5	Verificar y probar comandos para eliminar los archivos y carpetas creados anteriormente en el proceso de instalación del servicio.	3	*Desarrollo*
6	Probar comandos para mostrar el estado de iptables.	3	*Desarrollo*
7	Probar comandos para iniciar el servicio de iptables.	3	*Desarrollo*
8	Probar comandos para parar el servicio de iptables.	3	*Desarrollo*
9	Verificar y probar comandos para reiniciar el servicio de iptables	3	*Desarrollo*

Tabla 7: Tareas de ingeniería

Prueba

Una prueba no es más que ejecutar el software con determinados datos de entrada y producir resultados que luego serán comparados con los teóricos, con el objetivo de encontrar errores. Las pruebas de software se realizan durante todo el ciclo de vida de este, pero requieren mayor esfuerzo durante la fase de implementación.

Dentro de este flujo de trabajo se realiza una secuencia de actividades desarrolladas por los trabajadores involucrados. Aquí se definen un conjunto de métodos, niveles, estrategias y tipos de pruebas enfocadas a garantizar la eficiencia del producto final.

Para la realización de las pruebas del módulo desarrollado se utilizó la prueba de unidad y de este el método de Caja Blanca con el objetivo de utilizar el método del camino básico. No se utilizó el método de Caja Negra debido a que el alcance del trabajo de diploma no incluye la implementación de la interfaz gráfica de dicho módulo.

Nivel de prueba

Las **pruebas de unidad** centran el proceso de verificación en la menor unidad del diseño del software [12]. Esta pretende probar cada función en un archivo de programa simple. Esto quiere decir que un módulo que tiene una prueba de unidad se puede probar independientemente del resto del sistema.

Método a utilizar

Para la realización de las pruebas se utiliza el método de caja blanca. Estas son pruebas diseñadas después que existe un código fuente, comprobando los caminos lógicos y proponiendo casos de prueba que examinen que están correctas todas las condiciones y/o bucles para determinar si el estado real coincide con el esperado o afirmado.

La prueba de caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental y está dirigido a las funcionalidades internas de un sistema [12]. Esta se basa en el minucioso examen de los detalles procedimentales. Se comprueban los caminos lógicos del software proponiendo casos de pruebas que ejerciten conjuntos específicos de condiciones y/o bucles.

Mediante el método de Caja Blanca se pueden obtener casos de pruebas que garanticen lo siguiente:

- Todos los caminos independientes sean visitados almenos una vez.
- Que se ejerciten todas las decisiones lógicas, ya sea en con sus valores verdaderos o falsos.
- Que se ejecuten todos los bucles en sus límites y con sus límites operacionales.
- Que se tengan en cuenta y se ejecuten la estructura interna de los datos para asegurar su validez.

Para diseñar los casos de pruebas de caja blanca se utilizó la siguiente técnica:

Prueba del camino básico:

Esta es una técnica de prueba propuesta que permite tener la medida de la complejidad lógica de un diseño procedimental y usarla como guía para definir un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos, garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

Caso de Prueba

Específica la forma de probar un sistema incluyendo las entradas, salidas y resultados esperados, así como bajo qué condiciones debe probarse el sistema.

Para probar la calidad del módulo se realizó una etapa de prueba inicial correspondientes a la primera iteración del Plan de iteraciones. Este contiene las HU críticas del módulo, en las cuales se encontraron

tres no conformidades asociadas a errores de validación. En esta primera etapa de prueba fueron solucionados estos errores, dando paso a una segunda etapa de pruebas correspondiente a la segunda iteración del plan de iteraciones. En esta segunda iteración se identificaron errores semánticos los cuales fueron solucionados al concluir la dicha etapa de prueba.

Las pruebas que se le aplicaron a las funcionalidades del módulo realizaron mediante la técnica del camino básico abordado anteriormente. Para aplicar esta técnica, primeramente se enumera el código de la funcionalidad a robar como se muestra en la siguiente figura.

`public void delAllRules(LogicalServer logicalServer) throws JSchException, EActionWrong, ENotRulesFindall, ECantNotDeleteFile, EInvalidMacAddress, EInvalidMask, EInvalidPath, EIpNotMatch, ENot-Ports, ERangeAndSingleIP, EWrongChains, EprotocolEmpaty, IOExcep-tion, InterruptedException, JDOMException, SftpException, IOExcep-tion, InterruptedException, FileNotFoundException, EActionWrong, ECantNotDeleteFile, EInvalidMacAddress, EInvalidMask, EInvalid-Path, EIpNotMatch, ENotPorts, ERangeAndSingleIP, EWrongChains, EprotocolEmpaty, JDOMException, SftpException, InvalidAddress, EList {`  `if (rules.get(logicalServer).isEmpty()) {`	**1**
`throw new EList("The list is empty");`	**2**

`} else {`
`rules.get(logicalServer).clear();`
`}`
	3

`}`	**4**

Tabla 8: Enumeración del código

Luego de obtener la enumeración del código de la funcionalidad, se construye el grafo de flujo el cual representa el flujo de control lógico.

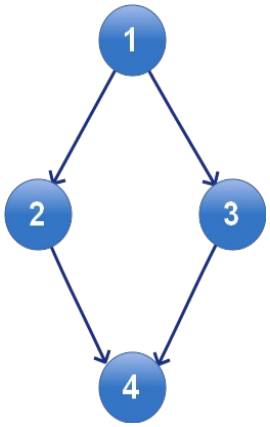


Figura 9: Esquema del grafo de flujo

Cálculo de la complejidad ciclomática

La Complejidad Ciclomática es una métrica de software que proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa. Se obtiene un límite superior para el número de pruebas que se deber realizar para asegurar que se ejecuta cada sentencia almenos una vez [12].

La complejidad ciclomática se puede calcular de tres formas:

V (G)= Número de regiones del grafo de flujo=3

V (G)= Aristas – Nodos + 2 =4 – 4 + 2

V (G)= Nodos predicados + 1= 1 + 1

V (G)= 2

El resultado de esta operación es el límite superior para el número de pruebas que se deben diseñar y ejecutar con el propósito de garantizar que se cubran todas las sentencias de los procedimientos. Este número también corresponde con el valor mínimo de casos de pruebas que deben realizarse teniendo en cuenta el procedimiento escogido. Dicho caso de prueba será descrito a continuación:

Caminos	Secuencia de nodos
Camino #1	1,2,4
Camino #2	1,3,4

Tabla 9: Caminos básicos

Caso de prueba para el camino básico #1	
Probador: José Carlos Miranda Azcuy	**Estado de la evaluación:** Satisfactorio
Descripción: Para realizar la prueba se debe introducir un objeto de tipo LogicalServer, el cual tiene como funcionalidad: conectarse al servidor que se desea administrar y traer hacia la maquina local el o los ficheros en los cuales se desea realizar la operación. En este caso el fichero de configuración de las reglas. Luego de realizar las operaciones dicho LogicalServer es el encargado de devolver los ficheros hacia su origen, es decir, hacia el servidor.	
Entradas: logicalServer	
Condición de Ejecución: Primeramente se debe realizar la conexión al servidor de forma satisfactoria. Pasado esto se comprueba si la lista de reglas está vacía. En caso de estar vacía, se lanza un mensaje de error ("The list is empty") y el método termina.	
Resultado: Se obtiene una lista vacía.	

Tabla 10: Caso de prueba para el camino básico #1

Caso de prueba para el camino básico #2	
Probador: José Carlos Miranda Azcuy	**Estado de la evaluación:** Satisfactorio
Descripción: Para realizar la prueba se debe introducir un objeto de tipo LogicalServer, el cual tiene como funcionalidad: conectarse al servidor que se desea administrar y copiar hacia la maquina local el o los ficheros en los cuales se desea realizar la operación. En este caso el fichero de configuración de las reglas. Luego de realizar las operaciones dicho LogicalServer es el encargado de devolver los ficheros hacia su origen, es decir, hacia el servidor.	
Entradas: logicalServer	
Condición de Ejecución: Primeramente se debe realizar la conexión al servidor de forma satisfactoria. Pasado esto se comprueba si la lista de reglas está vacía. En caso, de no estar vacía, se ejecuta el método clear() que contiene la Estructura de Datos lista de reglas con el objetivo de borrar todas las reglas que esta contiene.	
Resultado: Se obtiene una lista vacía.	

Tabla 11: Caso de prueba para el camino básico #2

Satisfacción del Cliente

Luego del proceso de prueba por parte del programador, se realizaron las pruebas por parte del cliente con el objetivo de garantizar el correcto funcionamiento de los requerimientos solicitados por su parte. Debido a que el alcance de trabajo, no incluye la implementación de la interfaz gráfica, dichas pruebas se realizaron a través de una clase Main de prueba. En esta clase el cliente pudo comprobar todas las funcionalidades del módulo, probando en estas los posibles valores que se le pueden introducir para el correcto funcionamiento de cada método. También tuvo en cuenta los posibles mensajes de errores que muestran cuando se introducen valores no válidos. Al concluir este proceso el cliente dio como resultado recomendaciones que se pueden tener en cuenta para optimizar algunas funcionalidades y su satisfacción con el módulo desarrollado autorizando su integración con la herramienta HMAST.

Conclusiones

Concluyendo con el proceso de investigación y de implementación del módulo se puede concluir que:

- La herramienta de administración de cortafuego más idónea para mejorar el entendimiento de las configuraciones del mismo por parte de los administradores y/o usuarios medios es Zentyal debido a intuitiva gráfica.
- A través de metodología, las tecnologías, herramientas, lenguajes e IDE de programación predefinidas por el cliente, se obtuvo el módulo de administración de cortafuego integrable con HMAST.
- El módulo desarrollado posee las funcionalidades necesarias para configurar satisfactoriamente un servicio de cortafuego ya que el mismo permite gestionar reglas de filtrado, traducción de direcciones de red y redirección de puertos.

Recomendaciones

Con la realización de este trabajo de diploma se desarrolló un módulo de Cortafuego para la herramienta HMAST cumpliendo así con los objetivos propuestos. Teniendo en cuenta la gama aplica de contenido asociado con las configuraciones de reglas de iptables se recomienda:

- Agregar la administración de la tabla de modificación de paquetes.
- Integrar el mismo con un servidor proxy.
- Adicionar los módulos de iptables para el tratamiento de seguimiento de conexiones, multipuertos y gestión de los logs.

Referencias bibliográficas

[1] «RedIRIS - Cortafuegos: Conceptos teóricos». [En línea]. Disponible en: http://www.rediris.es/cert/doc/unixsec/node23.html. [Accedido: 24-may-2013].

[2] C. H. Karanjit Siyan, *Internet y la seguridad en redes", Prentice-hall Hispanoamericana*. Firewalls y la seguridad en internet", Prentice-hall Hispanoamericana.

[3] William R. Cheswick, *Firewalls and Internet Security: Repelling the Wily Hacker*.

[4] K. Siyan y Chris Hare, *Firewalls y la seguridad en internet", Prentice-hall Hispanoamericana*.

[5] «Netfilter». [En línea]. Disponible en: http://netfilter.org. [Accedido: 24-may-2013].

[6] «Iptables». [En línea]. Disponible en: http://www.netfilter.org/projects/iptables/index.html. [Accedido: 24-may-2013].

[7] «man iptables». [En línea]. Disponible en: http://netfilter.org. [Accedido: 24-may-2013].

[8] «Shorewall». [En línea]. Disponible en: http://www.shorewall.net/Introduction.html. [Accedido: 24-may-2013].

[9] «UFW». [En línea]. Disponible en: http://gufw.org. [Accedido: 24-may-2013].

[10] «Zentyal». [En línea]. Disponible en: http://doc.zentyal.org. [Accedido: 24-may-2013].

[11] Gladys Marsi Peñalver Romero, «SXP, metodología de desarrollo de software», Universidad de las Ciencias Informáticas, La Habana.

[12] Roger S. Pressman, *Ingeniería del Software, un enfoque práctico*, 6ta ed. .

[13] Welcome to NetBeans. In: [online]. [Accessed 15 December 2012]. Available from: http://netbeans.org/.

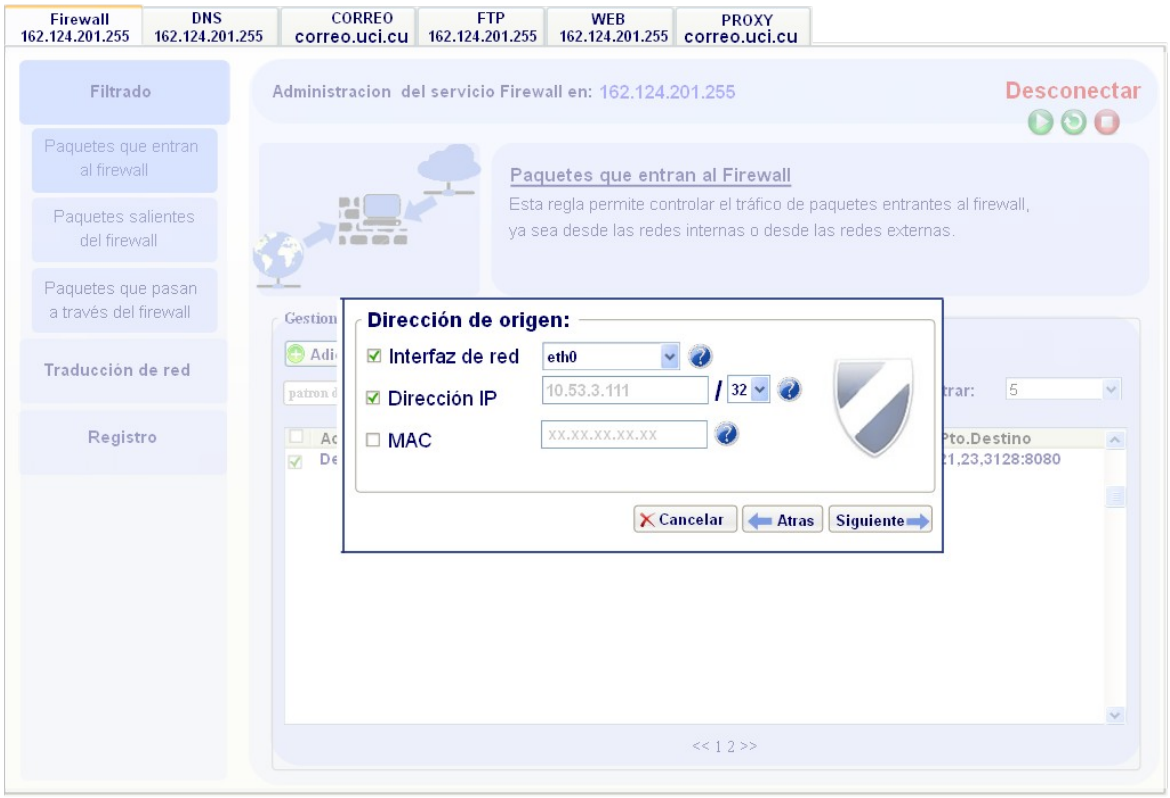
Bibliografía

1. Gladys Marsi Peñalver Romero, Sergio Jesús García de la Puente, Abel Meneses Abad. SXP, metodología de desarrollo de software. 2010.
2. Reidiel Castillo Ravelo y Pablo Soria Acosta. Herramienta para la Migración y Administración de Servicios Telemáticos (HMAST). 2012.
3. http://ubunto.es
4. http://solucionesinformaticas.com
5. http://esdebian.es
6. https://help.ubuntu.com/community/IptablesHowTo
7. https://help.ubuntu.com/6.10/ubuntu/serverguide/es/firewall-configuration.html
8. http://shorewall.net/
9. http://firehol.sourceforge.net/
10. http://www.fwbuilder.org/
11. http://www.fs-security.com/
12. http://dag.wieers.com/home-made/dwall/
13. http://www.solsoft.com/netfilterone
14. http://kmyfirewall.sourceforge.net/
15. http://www.simonzone.com/software/guarddog/
16. http://weblog.tinixtech.com.ar/
17. http://www.tutorial-es.com/seguridad/sistema/39-seguridad-con-reglas-iptables-firewall
18. http://fedoraproject.org/wiki/SystemConfig/firewall
19. http://www.yougetsignal.com/
20. http://www.canyouseeme.org/
21. http://shorewall.net/shorewall_index.htm#Releases
22. http://www.shorewall.net/News.htm
23. http://www.shorewall.net/shorewall_index.htm#License
24. http://www1.shorewall.net/pub/shorewall/4.4/shorewall-4.4.3/releasenotes.txt
25. http://xanderboy.esdebian.org/30317/entorno-escritorios-gnulinux-conocelos
26. http://miblogofinet.wordpress.com/2010/03/18/conoce-los-entornos-graficos-de-gnulinux/
27. http://www.linuxboricua.com/?p=1213
28. http://www.terra.es/personal/lermon/cat/articles/evin0259.htm
29. http://gufw.org/
30. http://gufw.tuxfamily.org/

31. http://www.fs-security.com/
32. http://www.kmyfirewall.org/
33. http://www.simonzone.com/software/guarddog/
34. http://www.fwbuilder.org/
35. http://freshmeat.net/projects/nuapplet/
36. http://proyectopinguino.blogspot.com/2008/12/escanea-los-puertos-de-tu-ordenador-con.html
37. http://usemoslinux.blogspot.com/
38. https://help.ubuntu.com/
39. http://www.balabit.com/network-security/zorp-gateway/
40. www.turtlefirewall.com
41. firewall.lutel.pl
42. http://www.zelow.no/floppyfw/
43. http://www.simonzone.com/software/guarddog/#introduction
44. http://man-es.debianchile.org/index.html
45. http://dns.bdat.net/documentos/cortafuego/t1.html
46. http://www.alcancelibre.org/
47. http://www.ite.educacion.es/formacion/materiales/85/cd/linux/m6/cortafuego_iptables.html
48. http://guimi.net/blogs/hiparco/
49. http://www.punto-libre.org/2010/12/implementando-un-firewall-basico-con.html
50. http://www.aclantis.com
51. http://www.aclantis.com/articulo.php?sid=2110
52. http://roble.pntic.mec.es/~sgonzale/linux/cortafuego.html
53. http://www.monografias.com/trabajos3/firewalls/firewalls.shtml#arriba
54. http://benavent.homeip.net:8080/links/herramientas.htm
55. http://glub.ehu.es/seguridad/
56. http://glub.ehu.es/seguridad/filtrado.html
57. http://glub.ehu.es/seguridad/deteccion.html
58. http://glub.ehu.es/seguridad/cgi.html
59. http://www.microsoft.com/latam/seguridad/proteccion/firewall.asp
60. http://www.tress.com.mx/boletin/julio2003/firewall.htm
61. http://www.delitosinformaticos.com/especial/seguridad/politica.shtml
62. http://www.arcert.gov.ar/curso_firewalls/curso_f.htm

Anexos

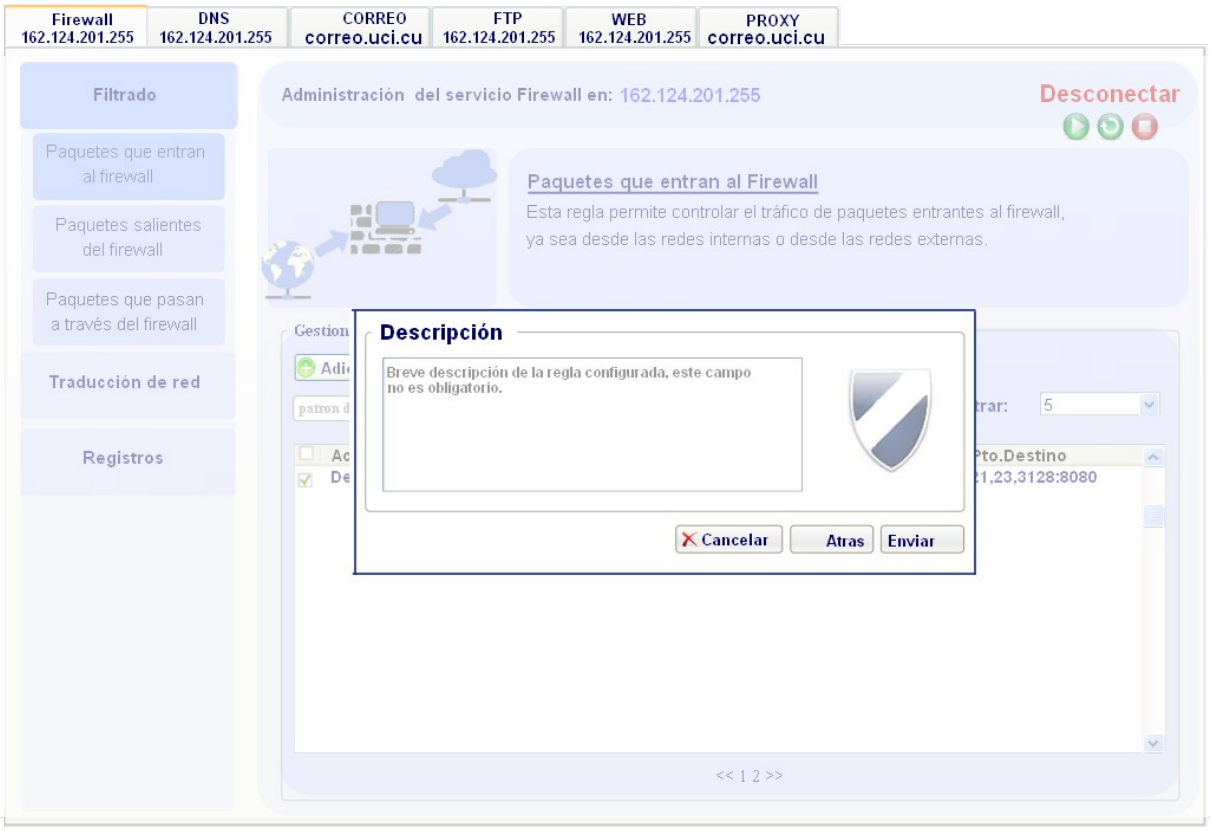
Anexo 1: Página principal de filtrado



Anexo 2: Dirección de origen (Paquetes entrantes al cortafuego)

Anexo 3: Dirección de destino (Paquetes entrantes al cortafuego)

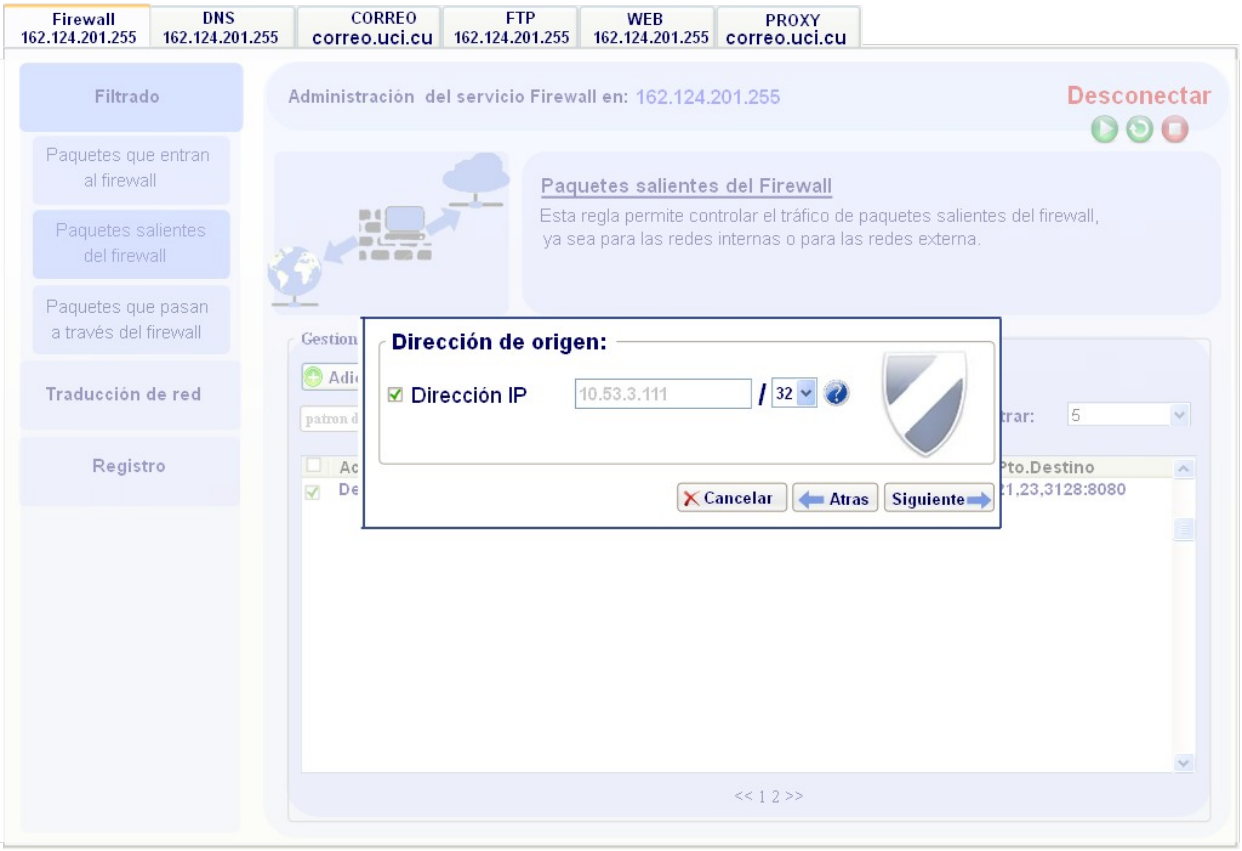
Anexo 4: Protocolos y puertos (Paquetes entrantes al cortafuego)



Anexo 5: Descripción (Paquetes entrantes al cortafuego)

Anexo 6: Lista de reglas (Paquetes que entrantes al cortafuego)

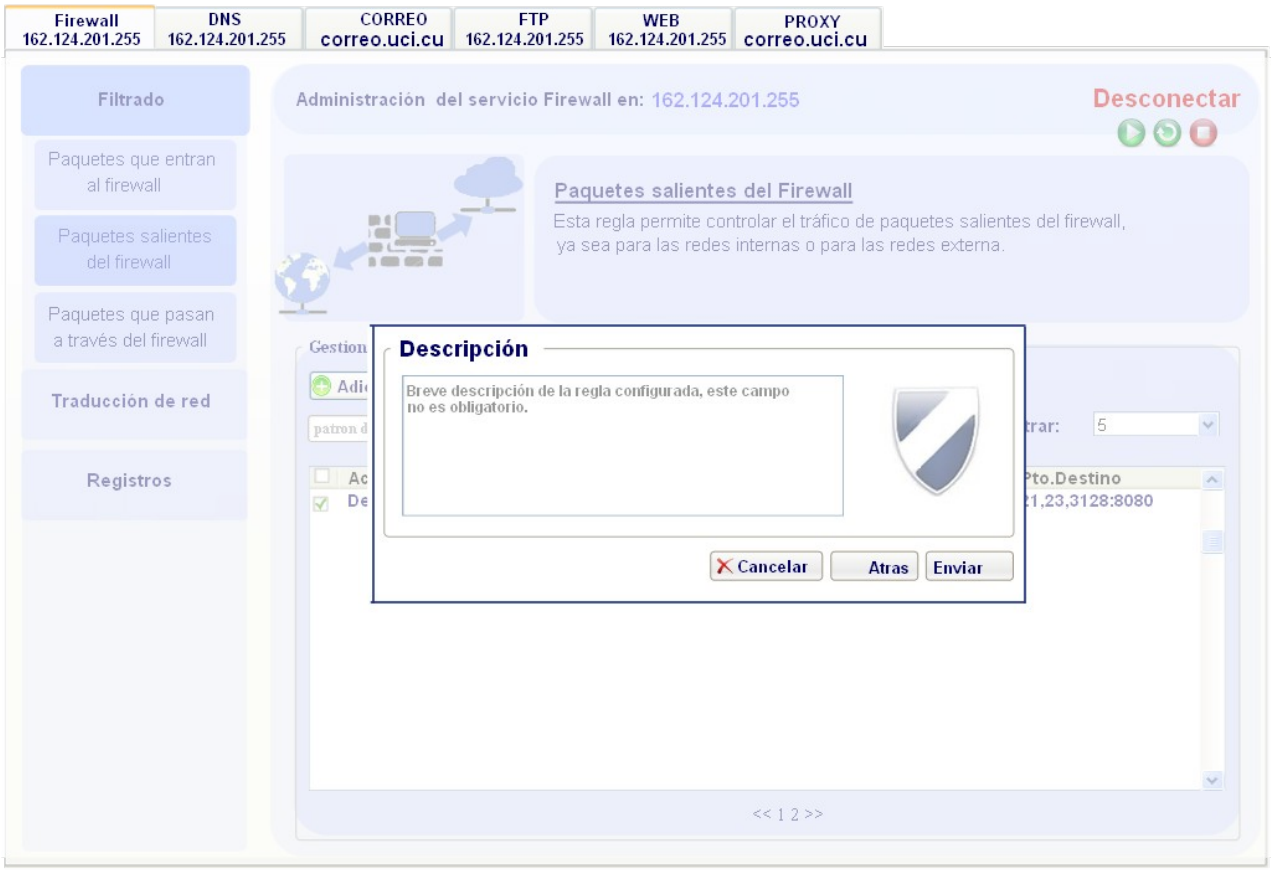
Anexo 7: Tipo de regla (Paquetes salientes del cortafuego)



Anexo 8: Dirección de origen (Paquetes salientes del cortafuego)

Anexo 9: Dirección de destino (Paquetes salientes del cortafuego)

Anexo 10: Protocolos y puertos (Paquetes salientes del cortafuego)



Anexo 11: Descripción (Paquetes salientes del cortafuego)

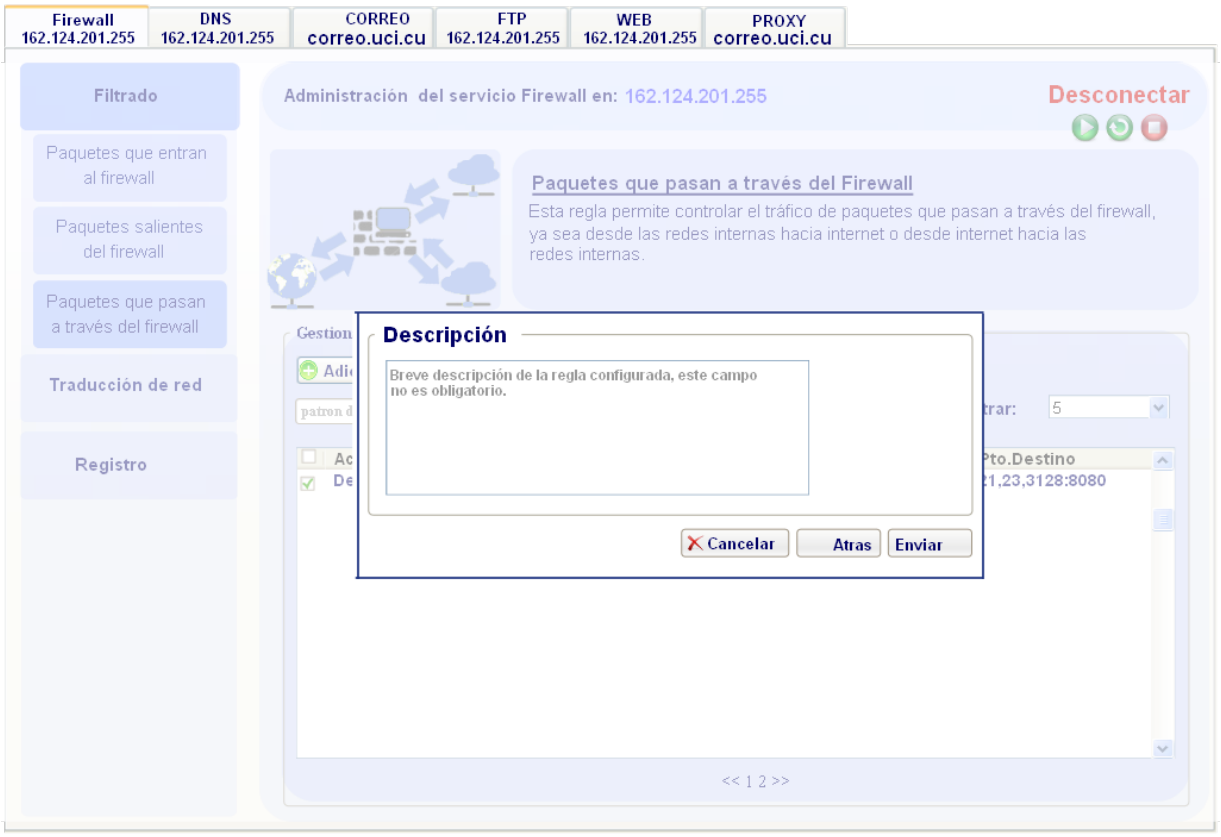
Anexo 12: Lista de reglas (Paquetes salientes del cortafuego)

Anexo 13: Tipo de regla (Paquetes que pasan a través del cortafuego)

Anexo 14: Dirección de origen (Paquetes que pasan a través del cortafuego)

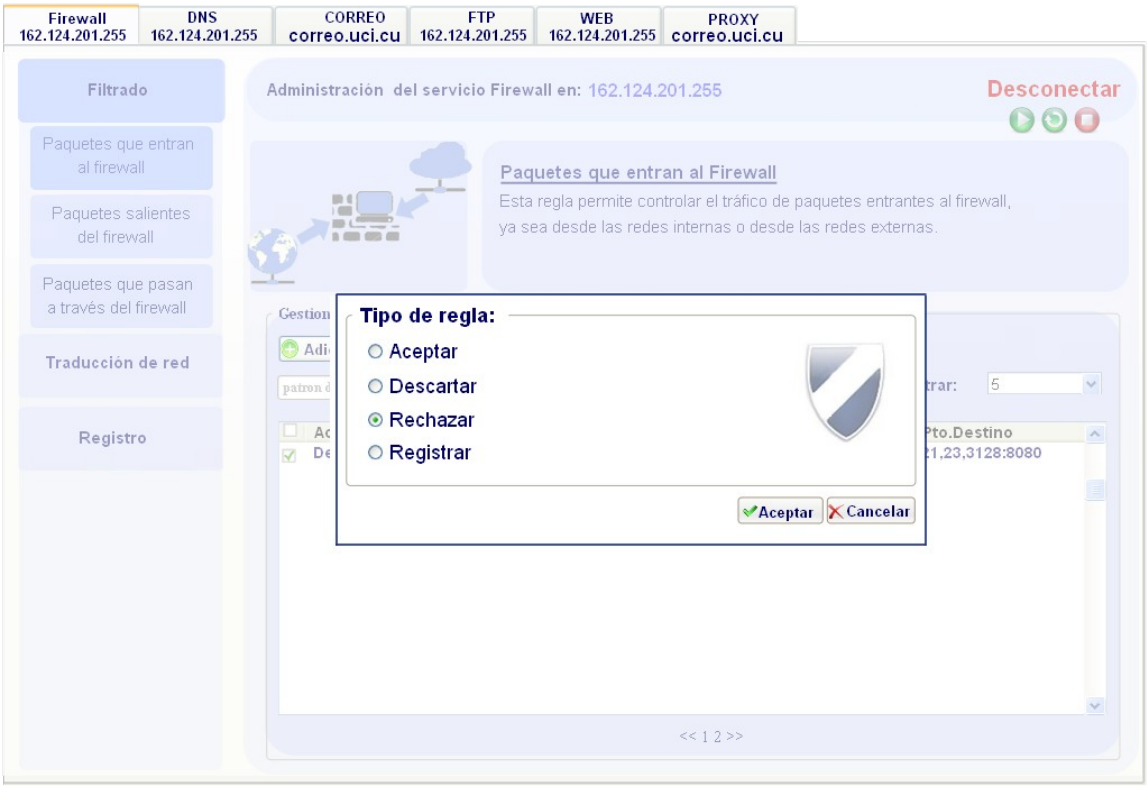
Anexo 15: Dirección de destino (Paquetes que pasan a través del cortafuego)

Anexo 16: Protocolos y puertos (Paquetes que pasan a través del cortafuego)

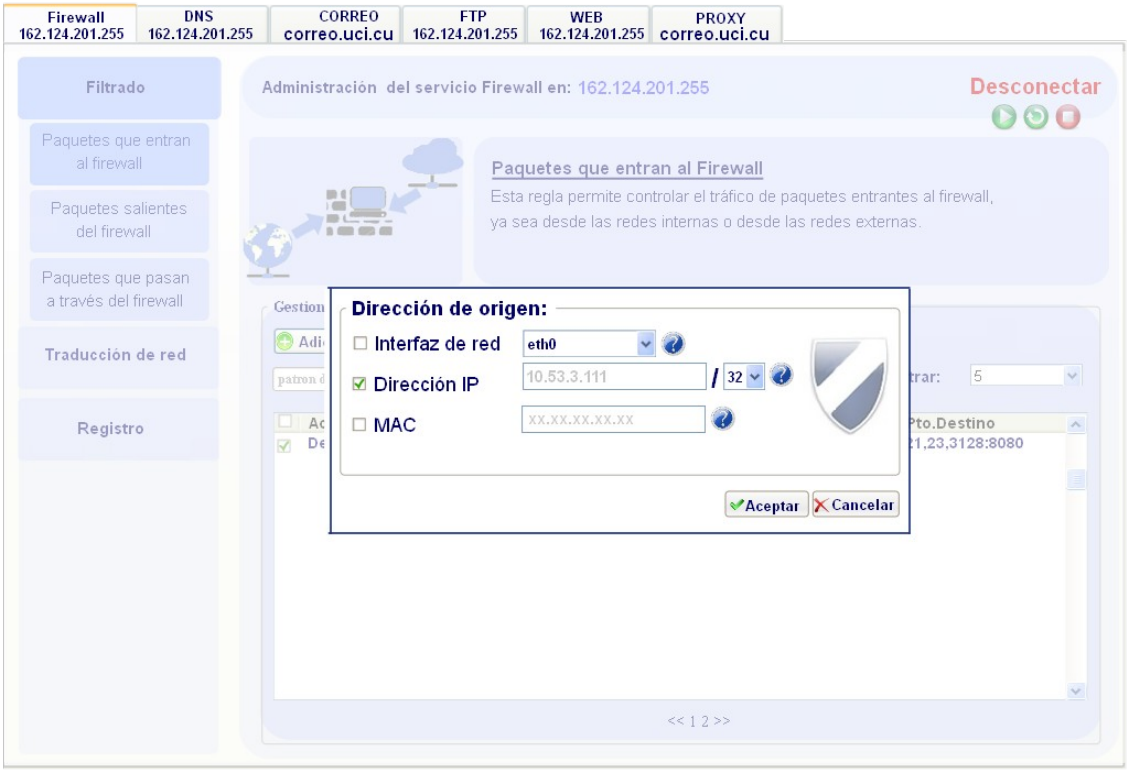


Anexo 17: Descripción (Paquetes que pasan a través del cortafuego)

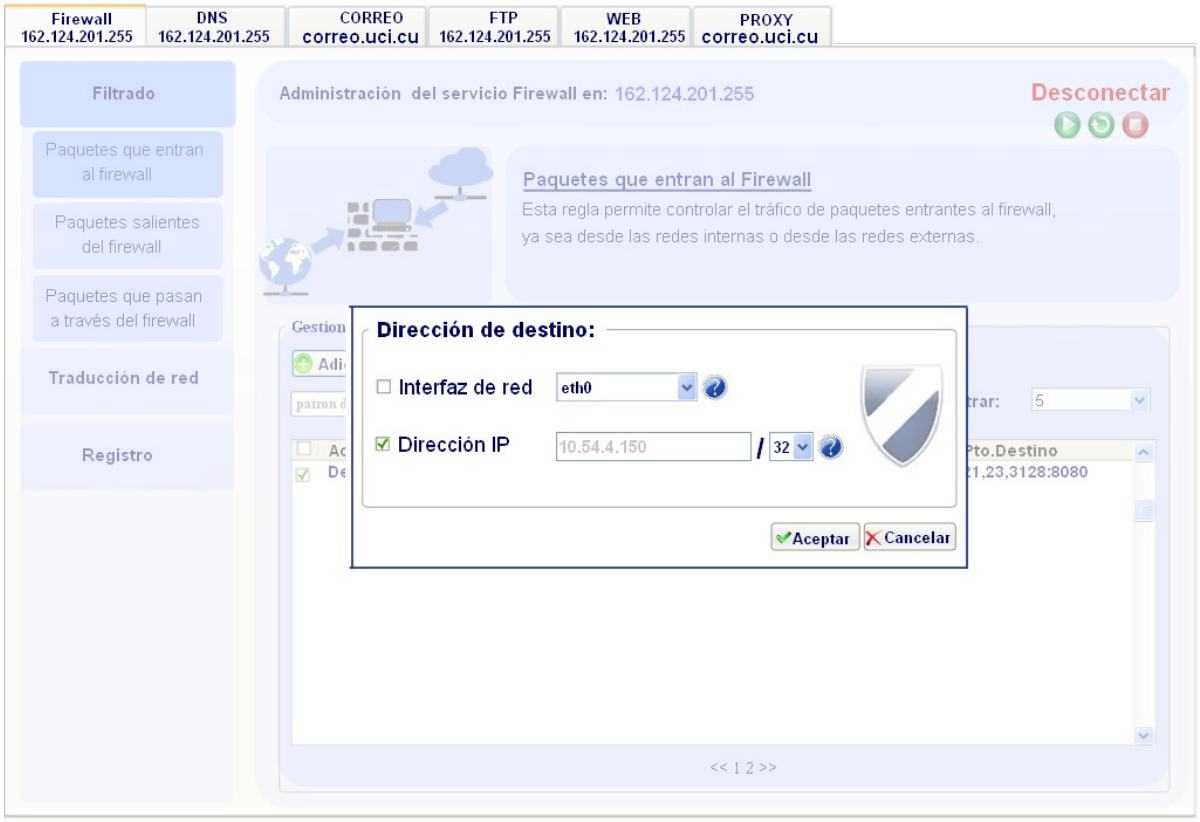
Anexo 18: Lista de reglas (Paquetes que pasan a través del cortafuego)



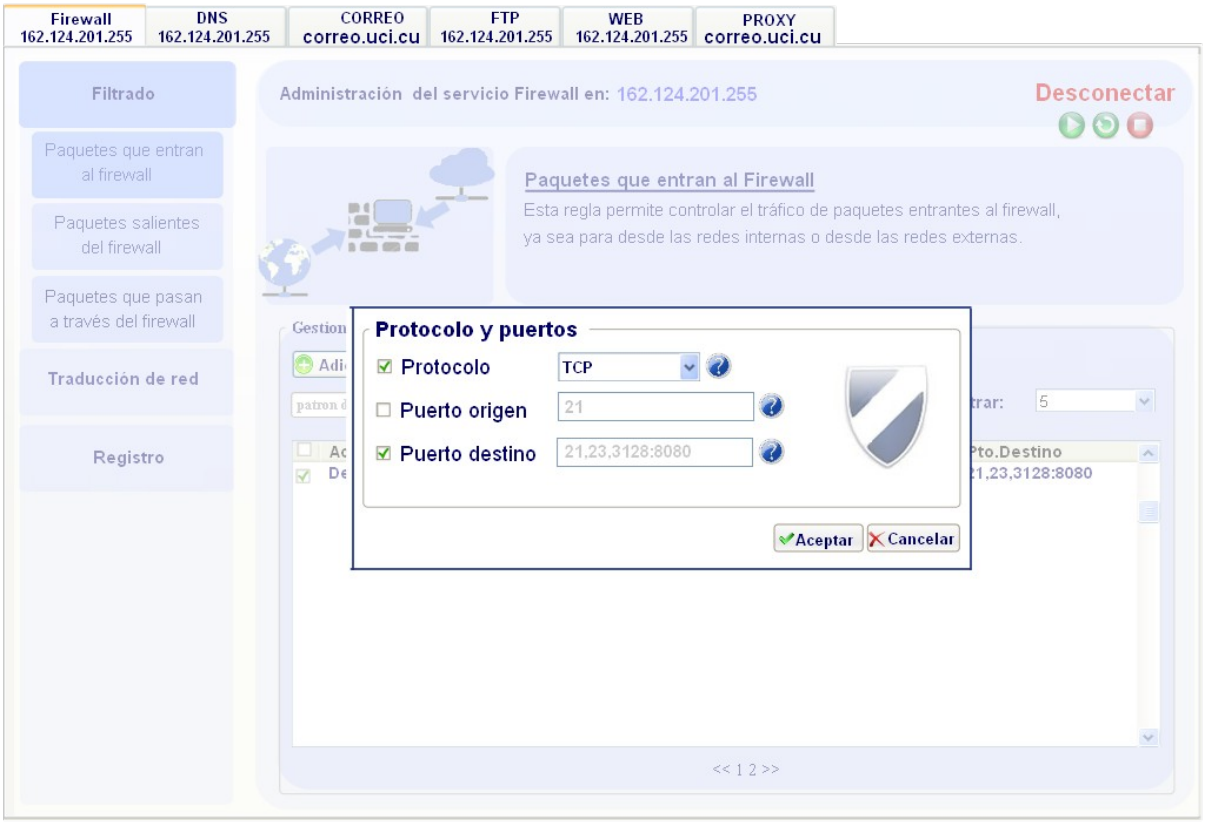
Anexo 19: Modificar: Tipo de regla (Paquetes entrantes al cortafuego)



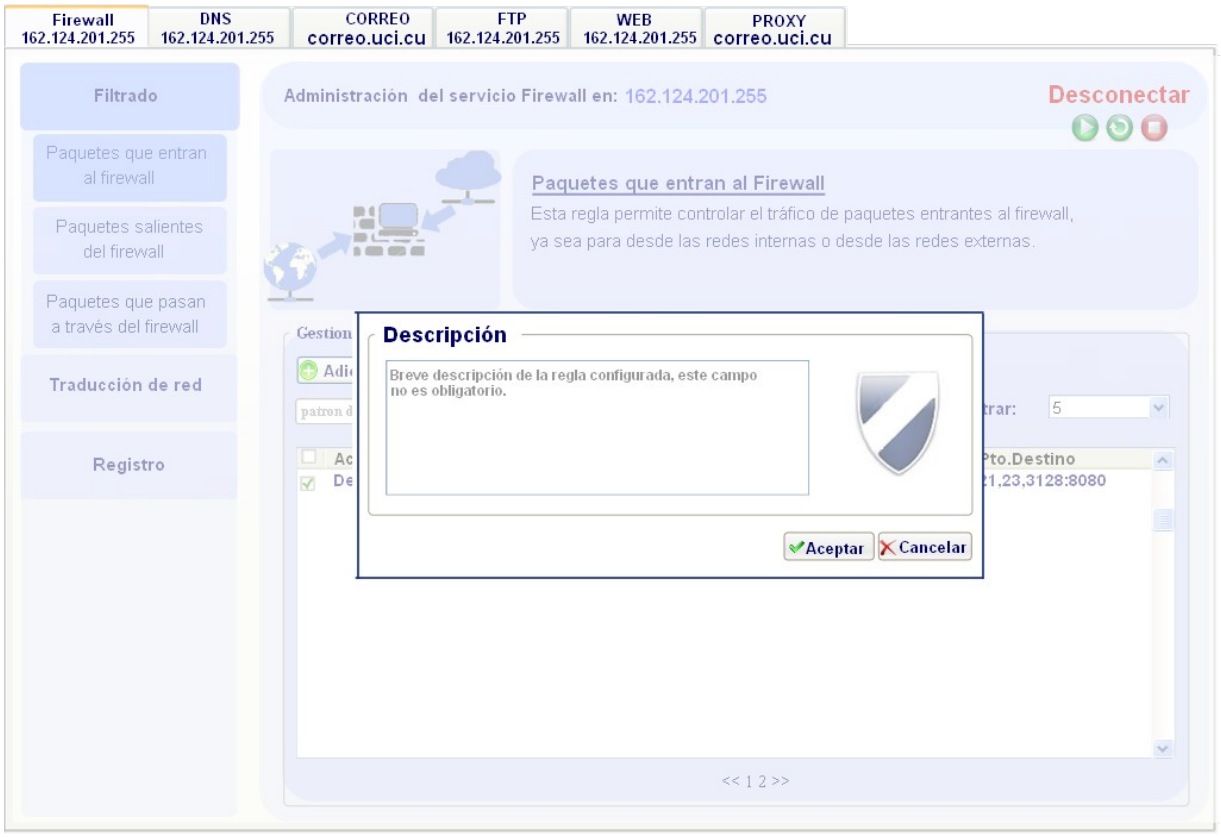
Anexo 20: Modificar: Dirección de origen (Paquetes entrantes al cortafuego)



Anexo 21: Modificar: Dirección de destino (Paquetes entrantes al cortafuego)

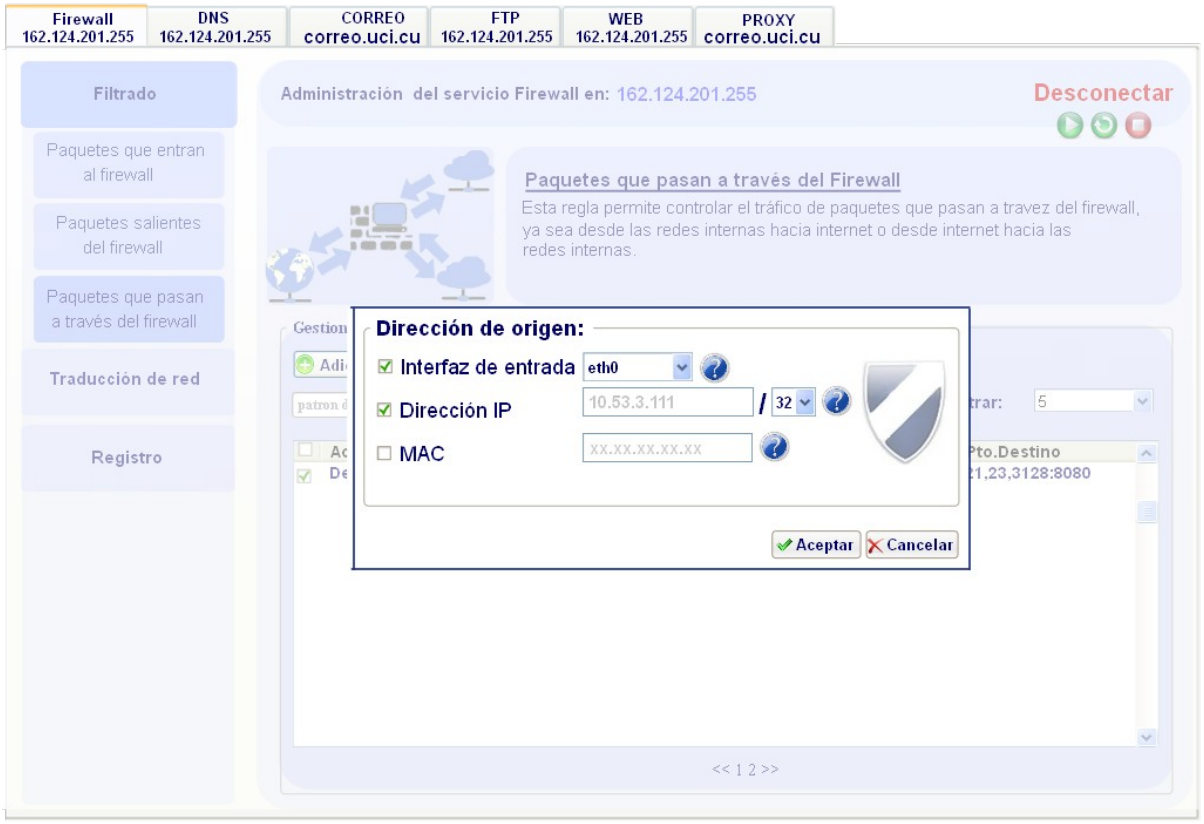


Anexo 22: Modificar: Protocolos y puertos (Paquetes entrantes al cortafuego)

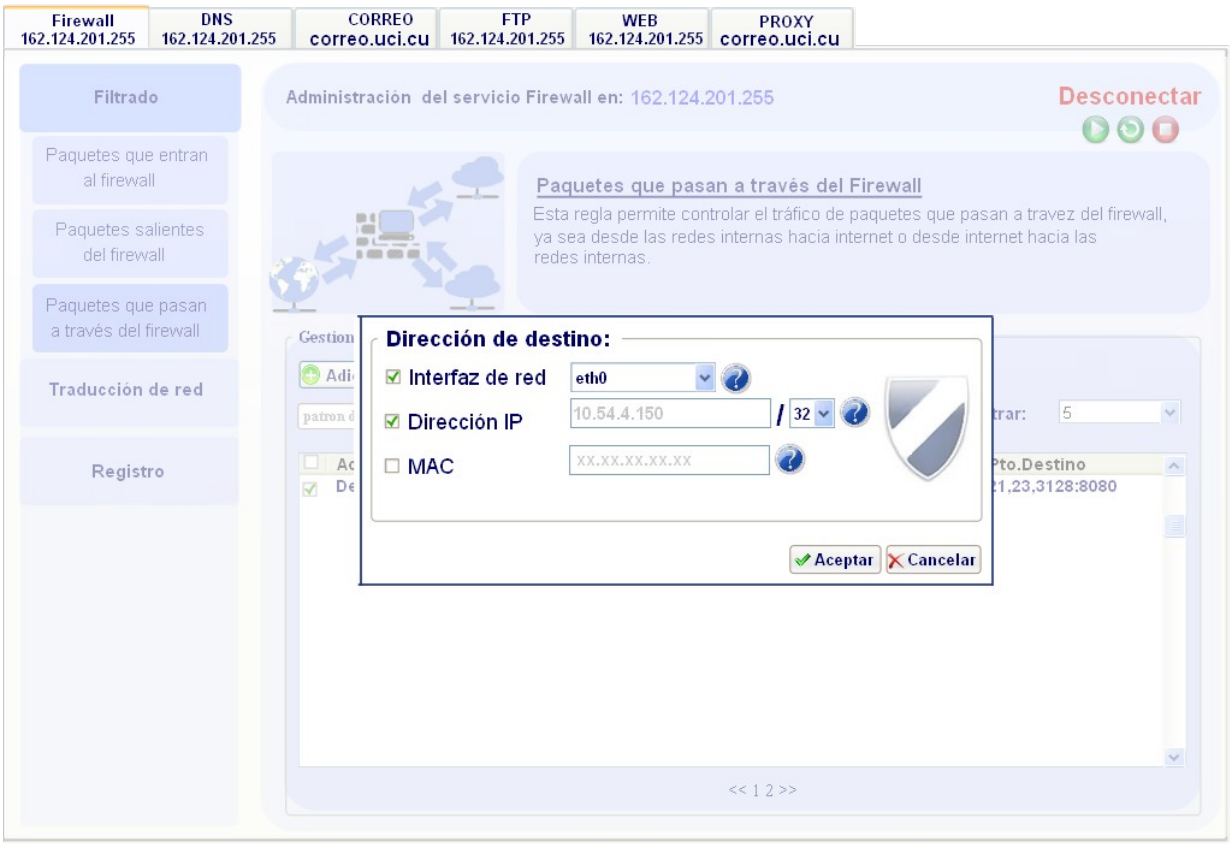


Anexo 23: Modificar: Descripción (Paquetes entrantes al cortafuego)

Anexo 24: Modificar: Tipo de regla (Paquetes que pasan a través del cortafuego)

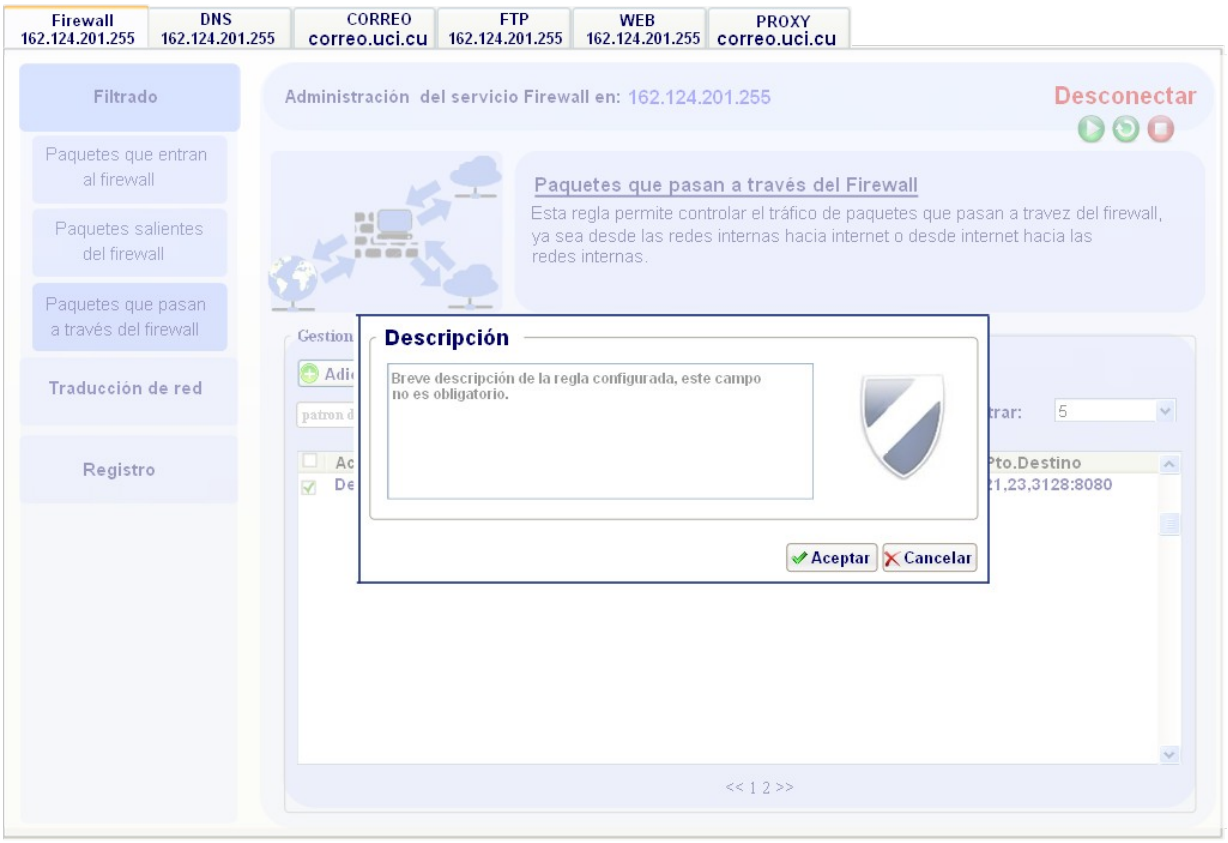


Anexo 25: Modificar: Dirección de origen (Paquetes que pasan a través del cortafuego)



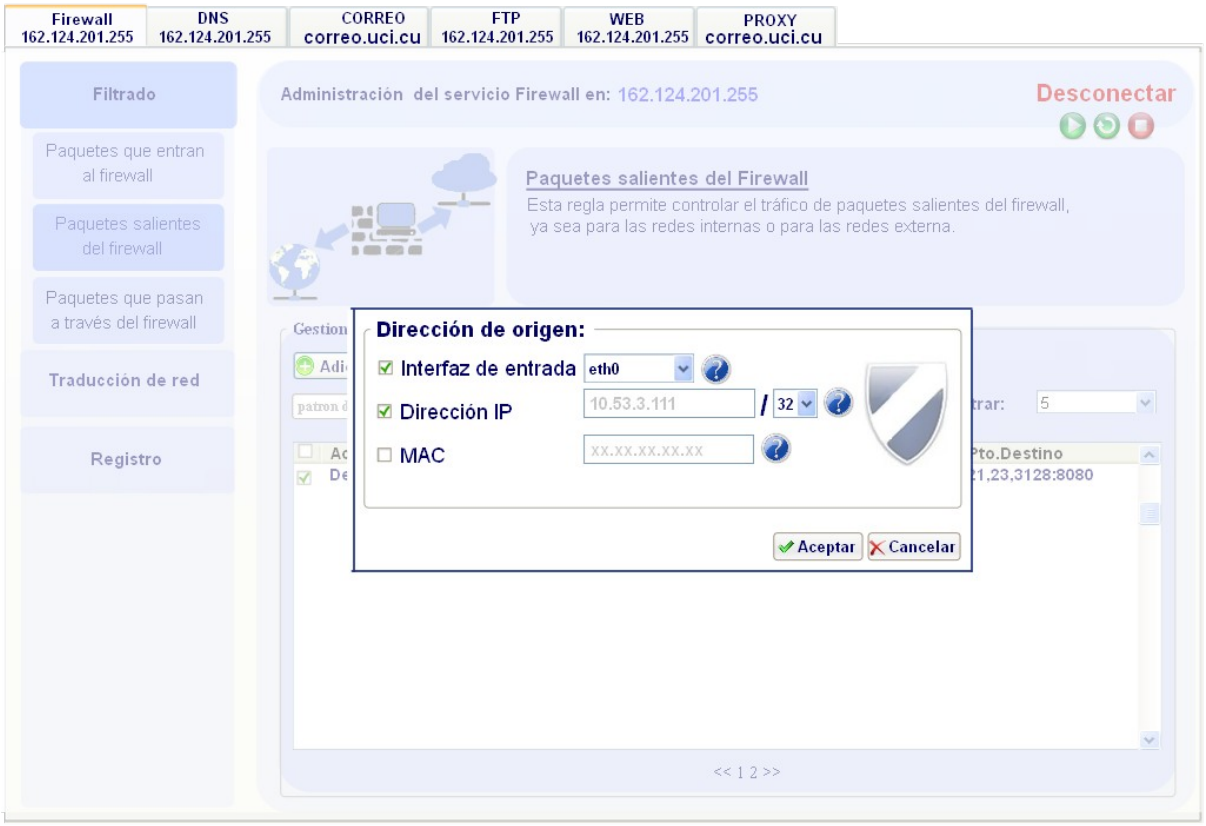
Anexo 26: Modificar: Dirección de destino (Paquetes que pasan a través del cortafuego)

Anexo 27: Modificar: Protocolos y puertos (Paquetes que pasan a través del cortafuego)

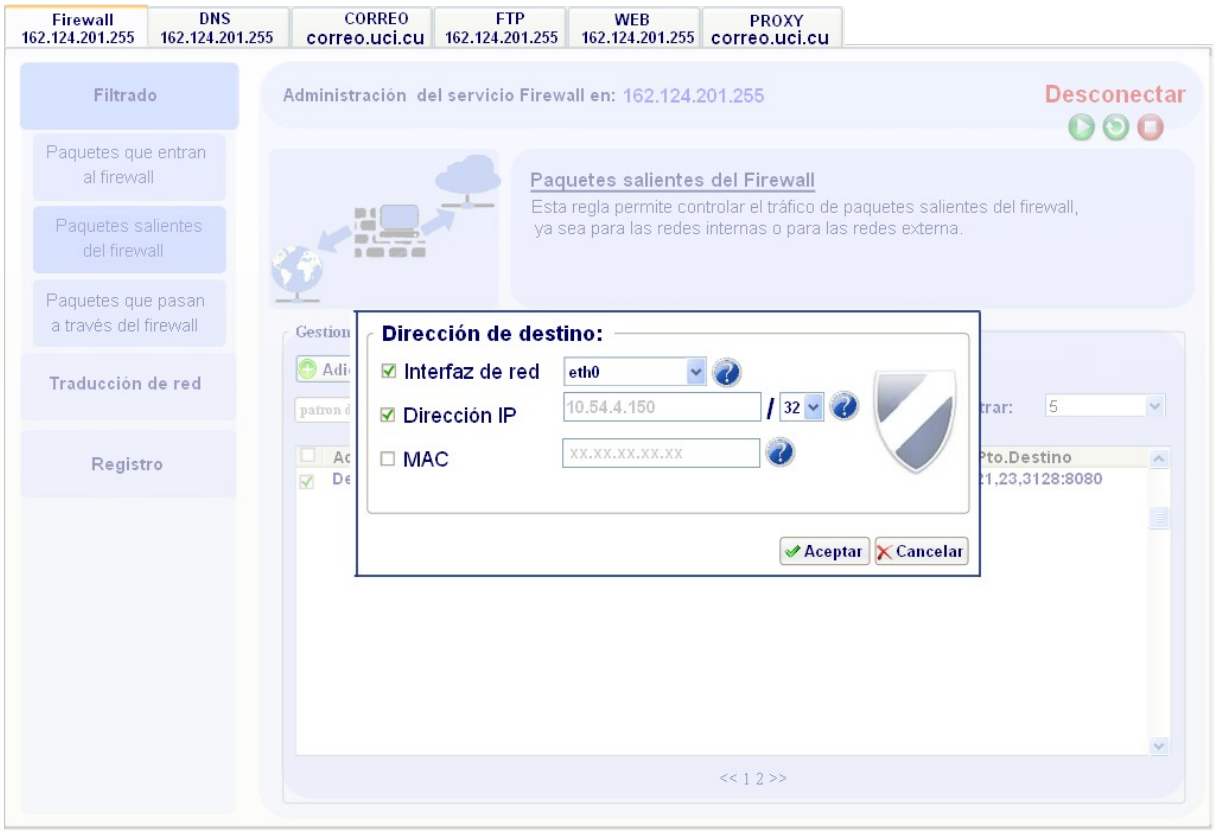


Anexo 28: Modificar: Descripción (Paquetes que pasan a través del cortafuego)

Anexo 29: Modificar: Tipo de Regla (Paquetes que salen del cortafuego)

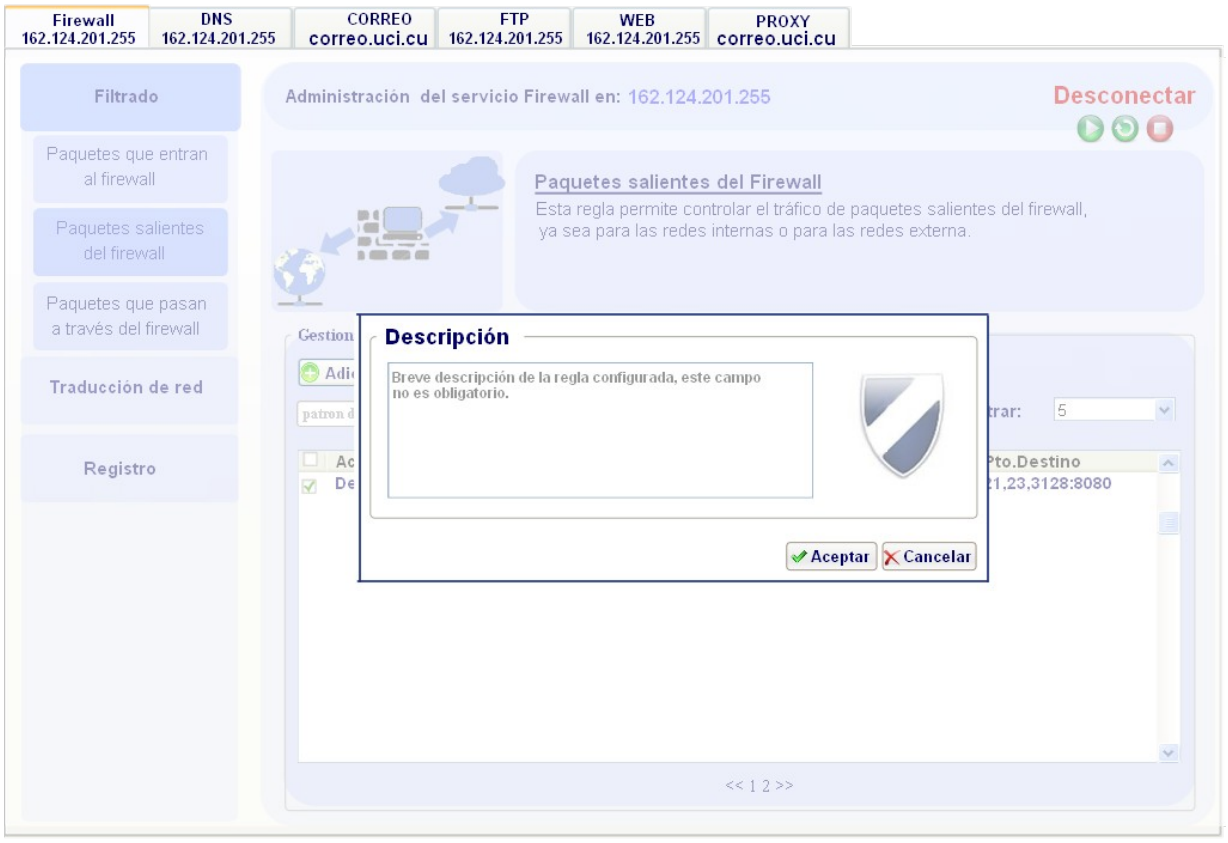


Anexo 30: Modificar: Dirección de origen(Paquetes que salen del cortafuego)

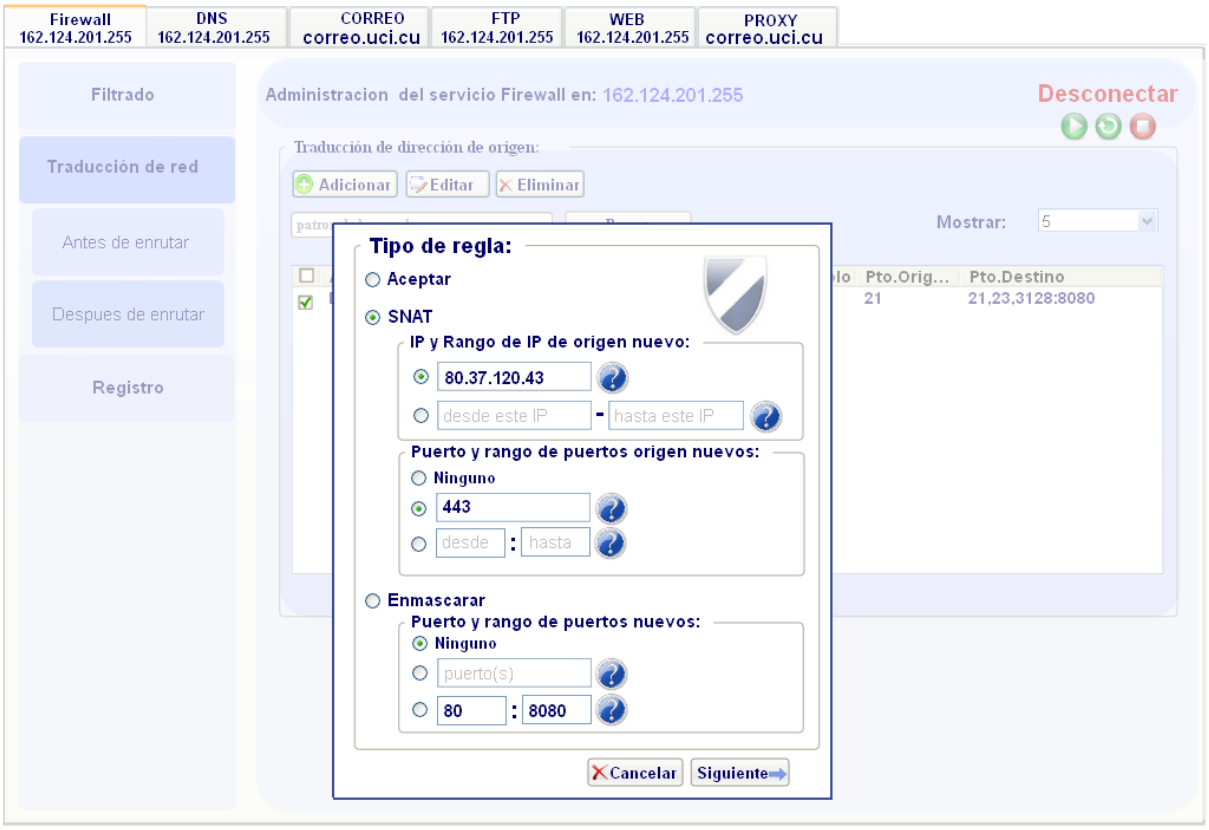


Anexo 31: Modificar: Dirección de destino (Paquetes que salen del cortafuego)

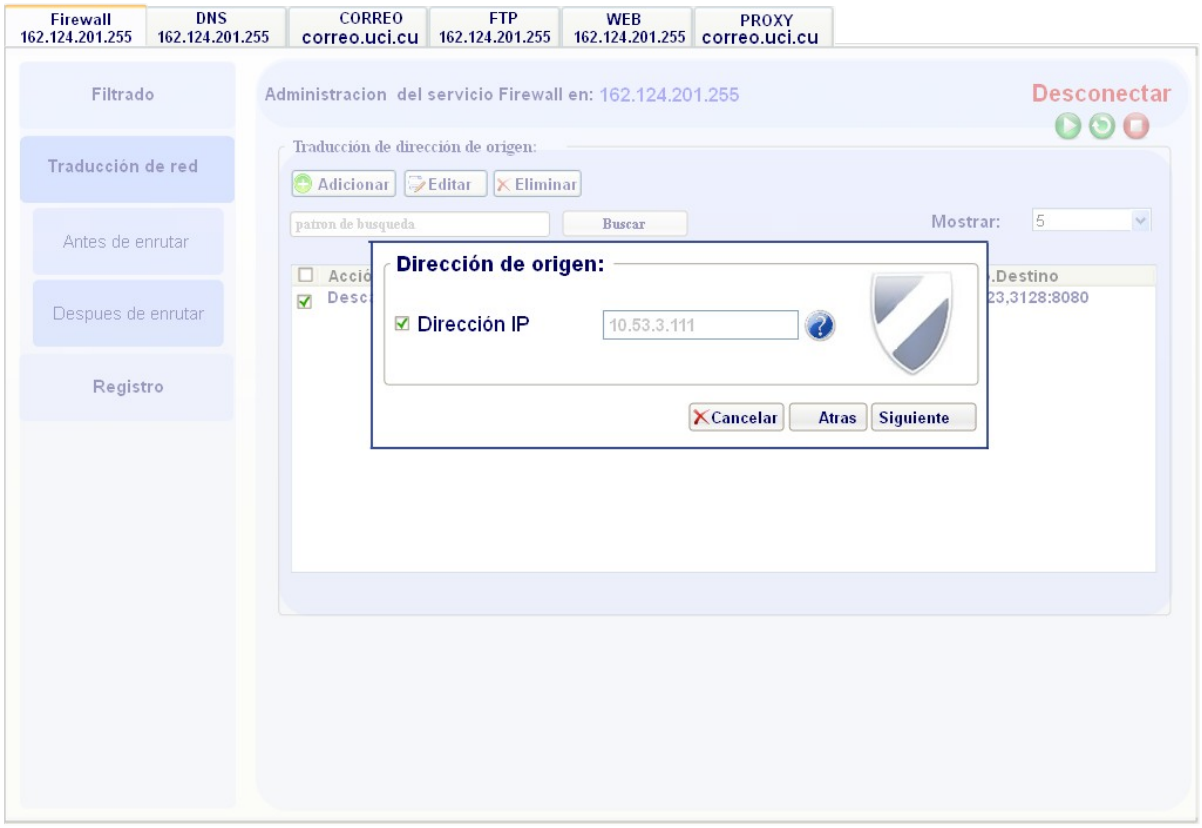
Anexo 32: Modificar: Protocolo y Puertos (Paquetes que salen del cortafuego)



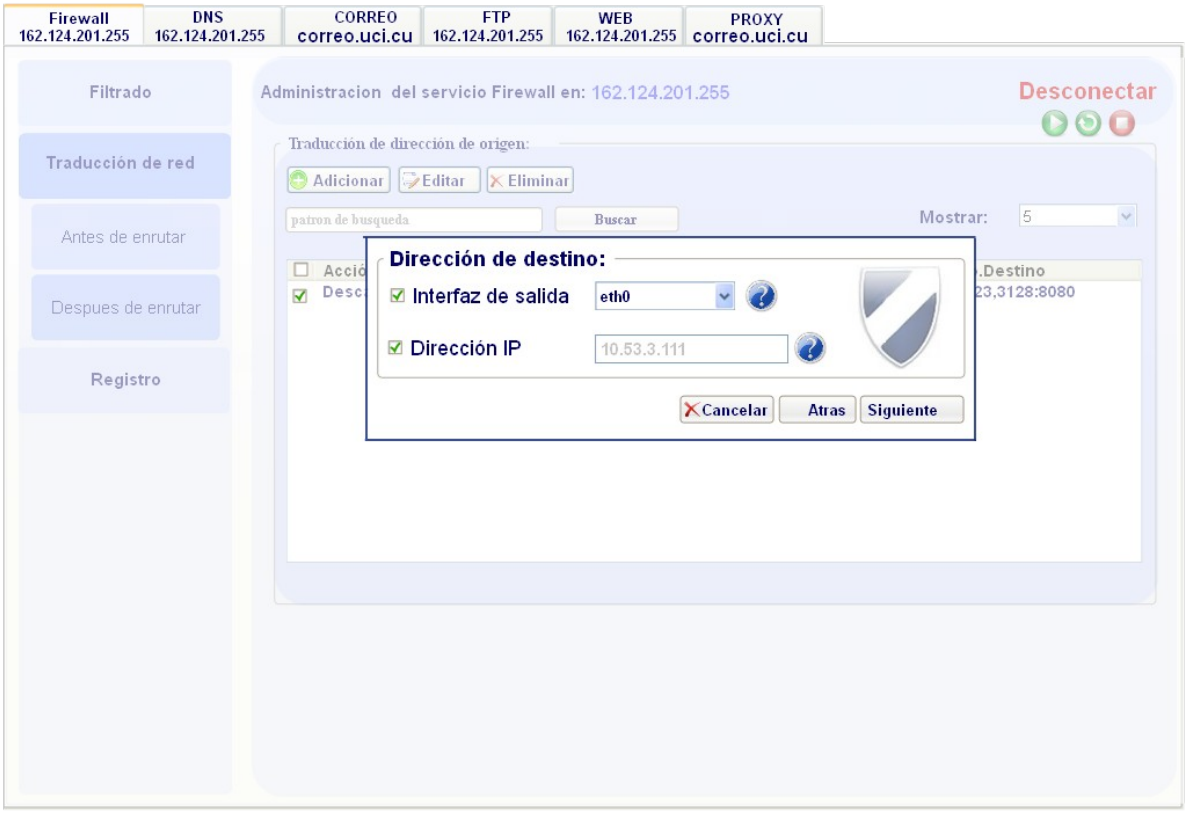
Anexo 33: Modificar: Descripción (Paquetes que salen del cortafuego)



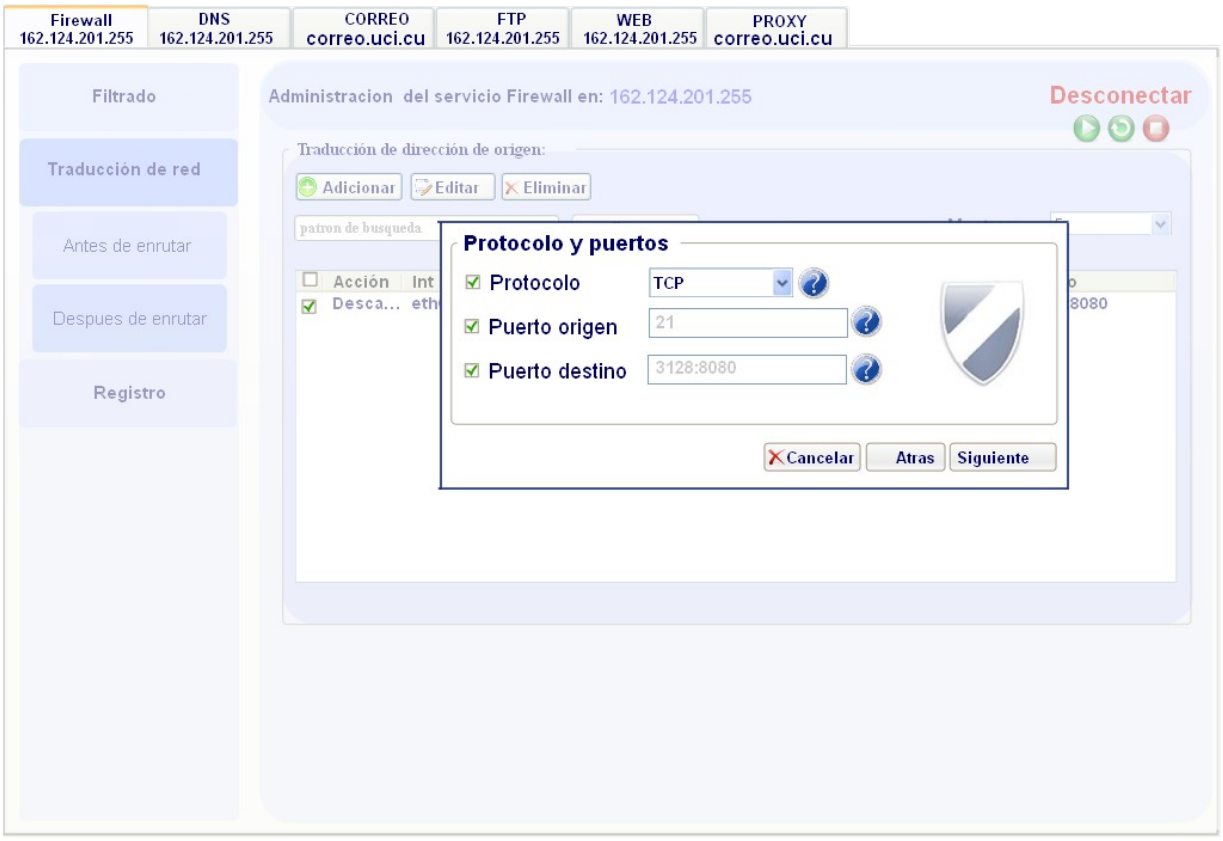
Anexo 34: Tipo de regla(Paquetes que aún no han sido enrutados)



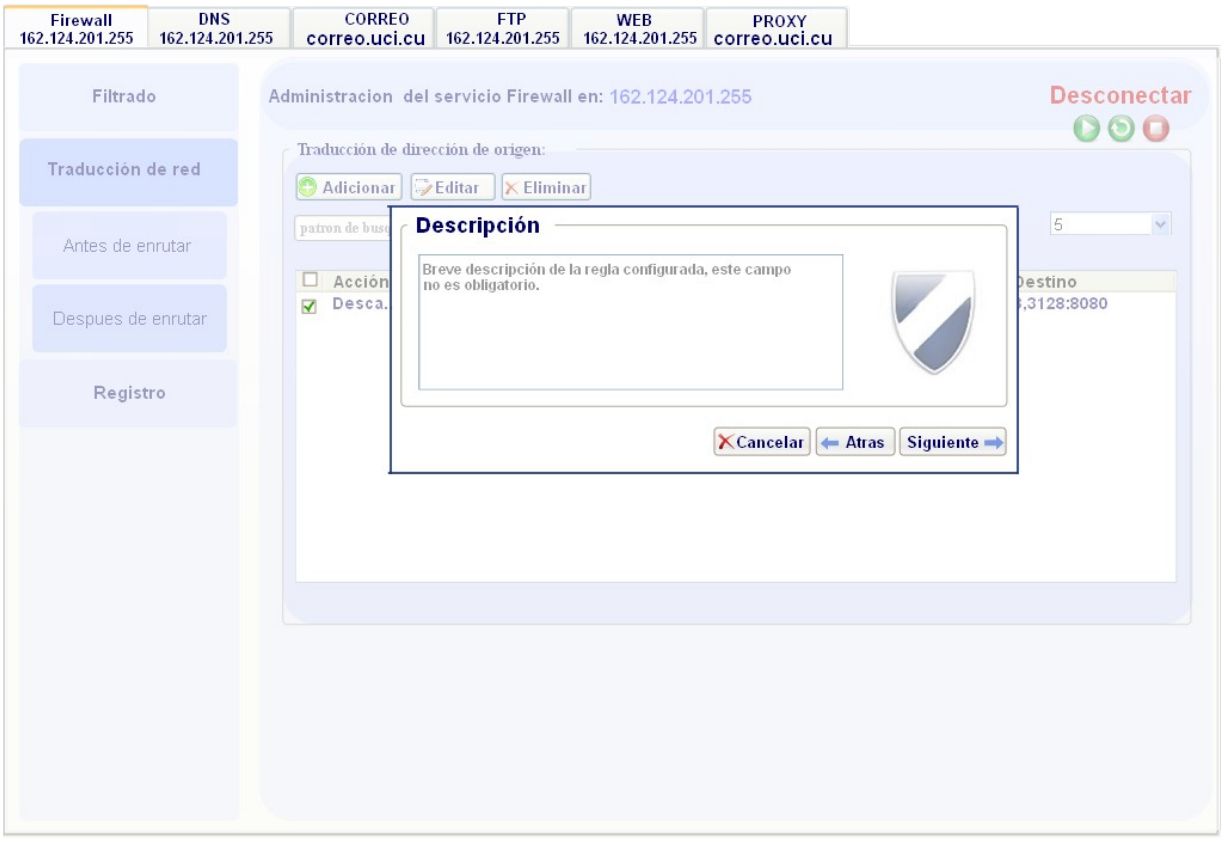
Anexo 35: Dirección de origen(Paquetes que aún no han sido enrutados)



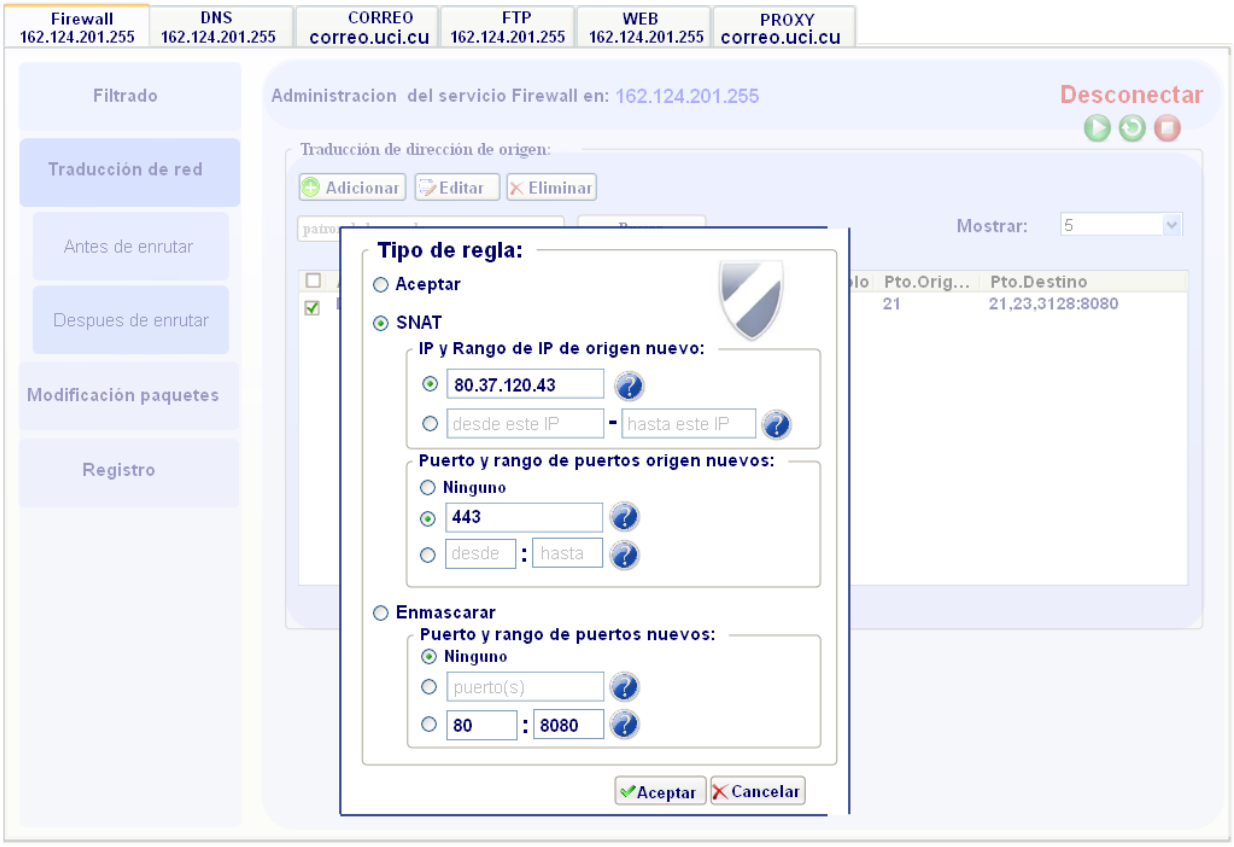
Anexo 36: Dirección de destino (Paquetes que aún no han sido enrutados)



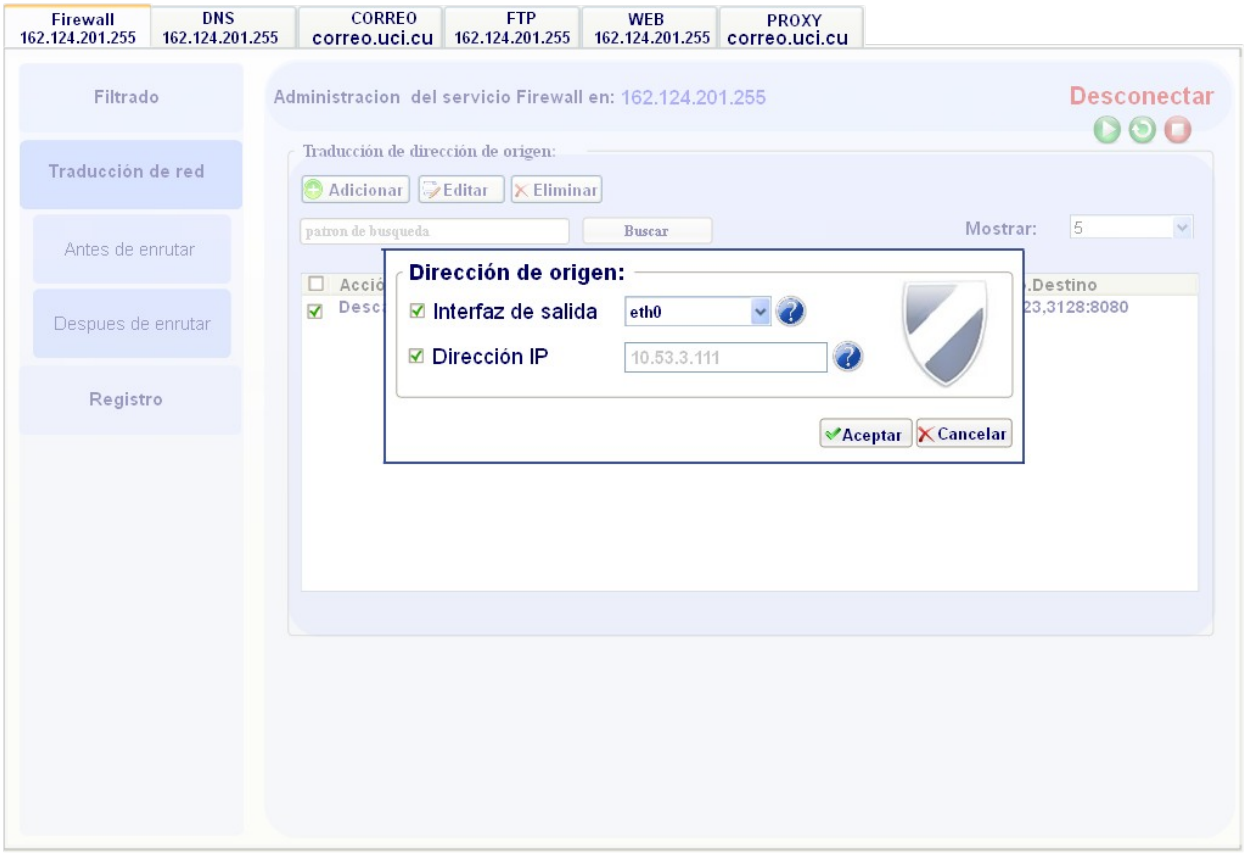
Anexo 37: Protocolo y puertos (Paquetes que aún no han sido enrutados)



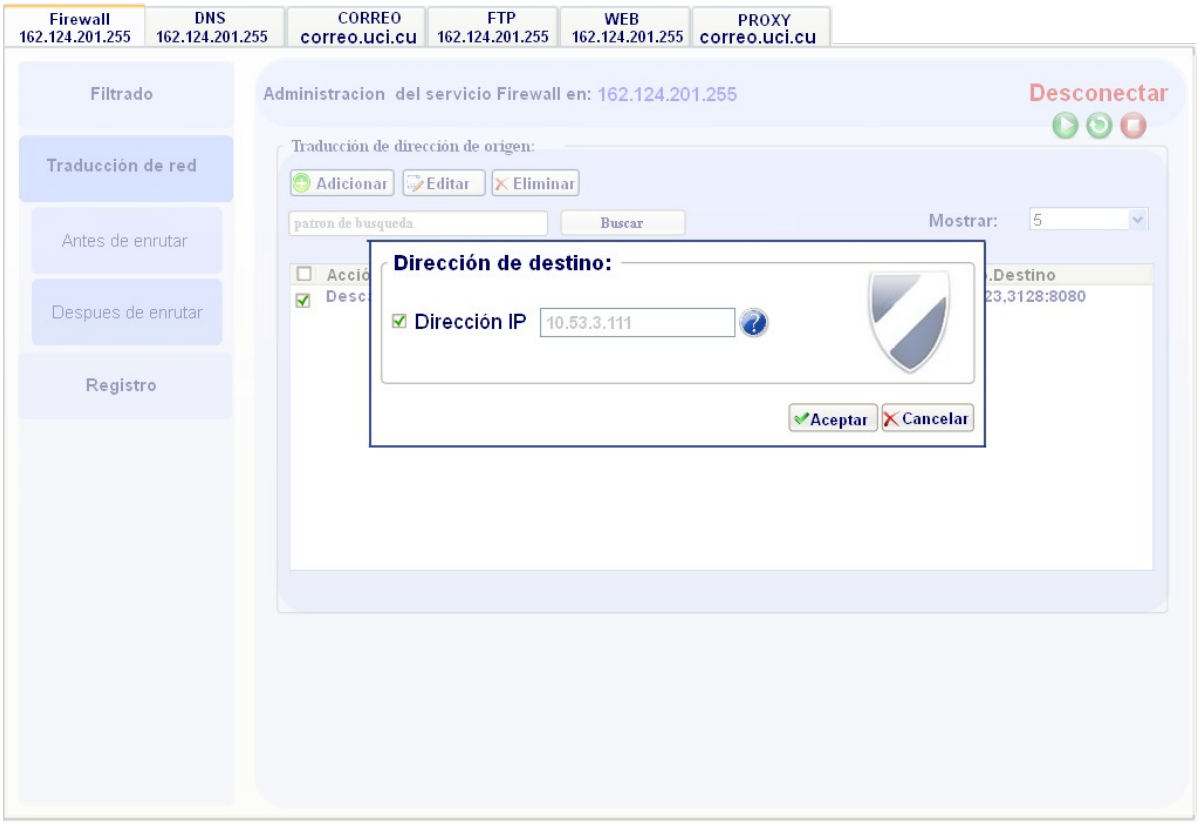
Anexo 38: Descripción (Paquetes que aún no han sido enrutados)



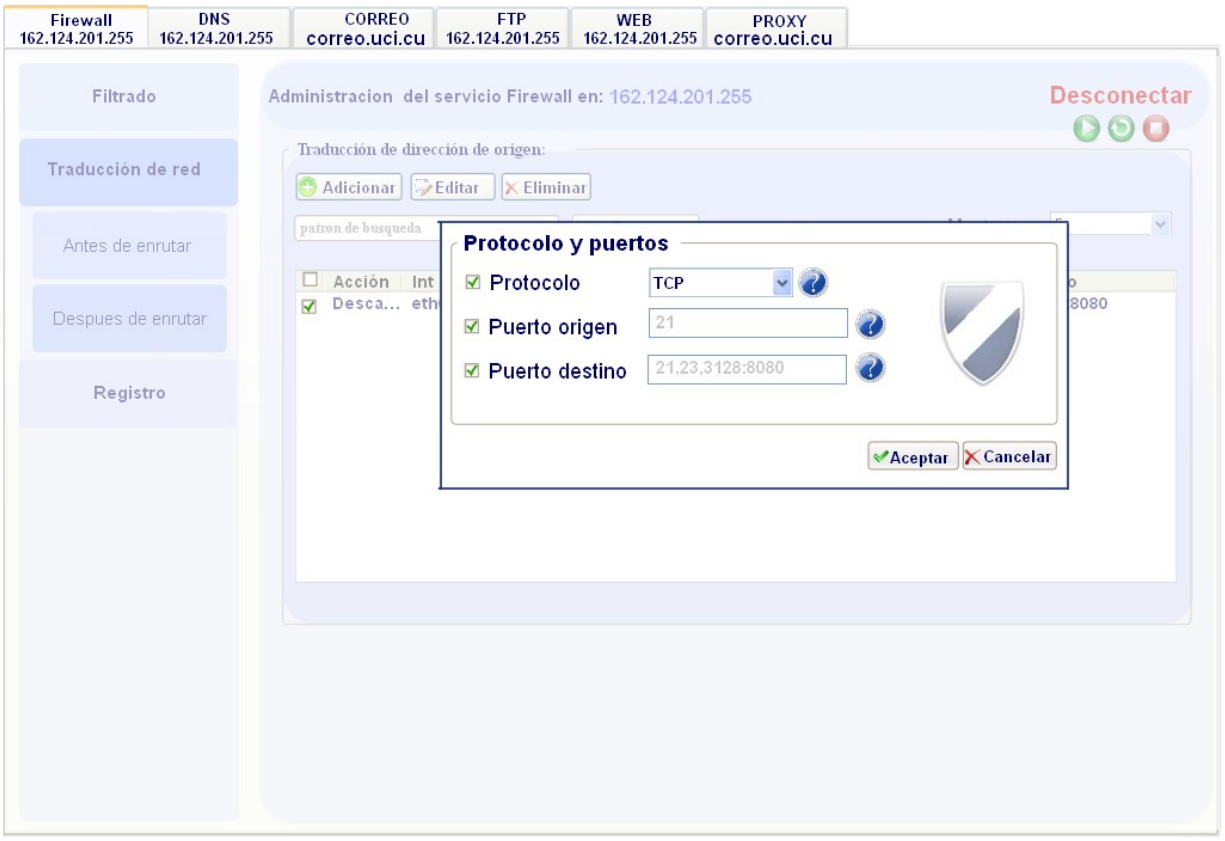
Anexo 39: Modificat:Tipo de regla(Paquetes que aún no han sido enrutados)



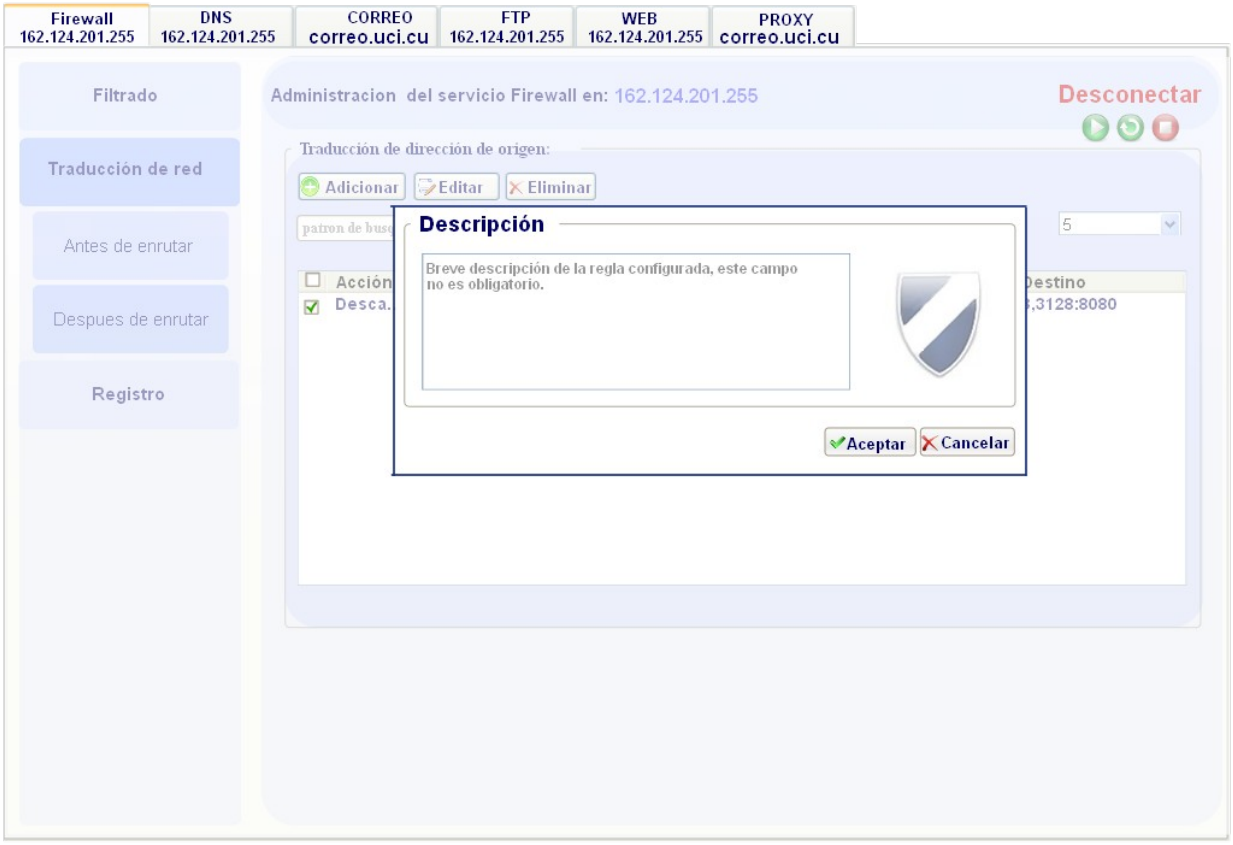
Anexo 40: Modificar: Dirección de origen (Paquetes que aún no han sido enrutados)



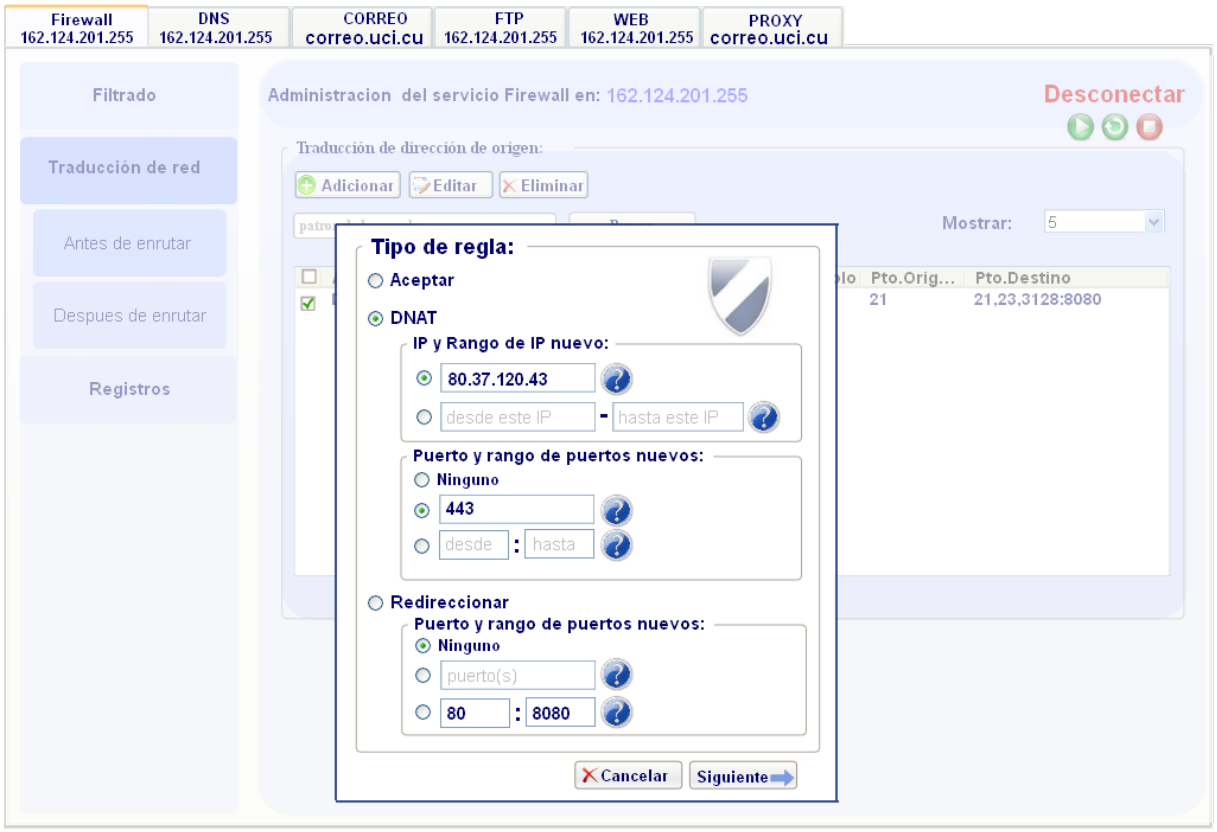
Anexo 41: Modificar: Dirección de destino (Paquetes que aún no han sido enrutados)



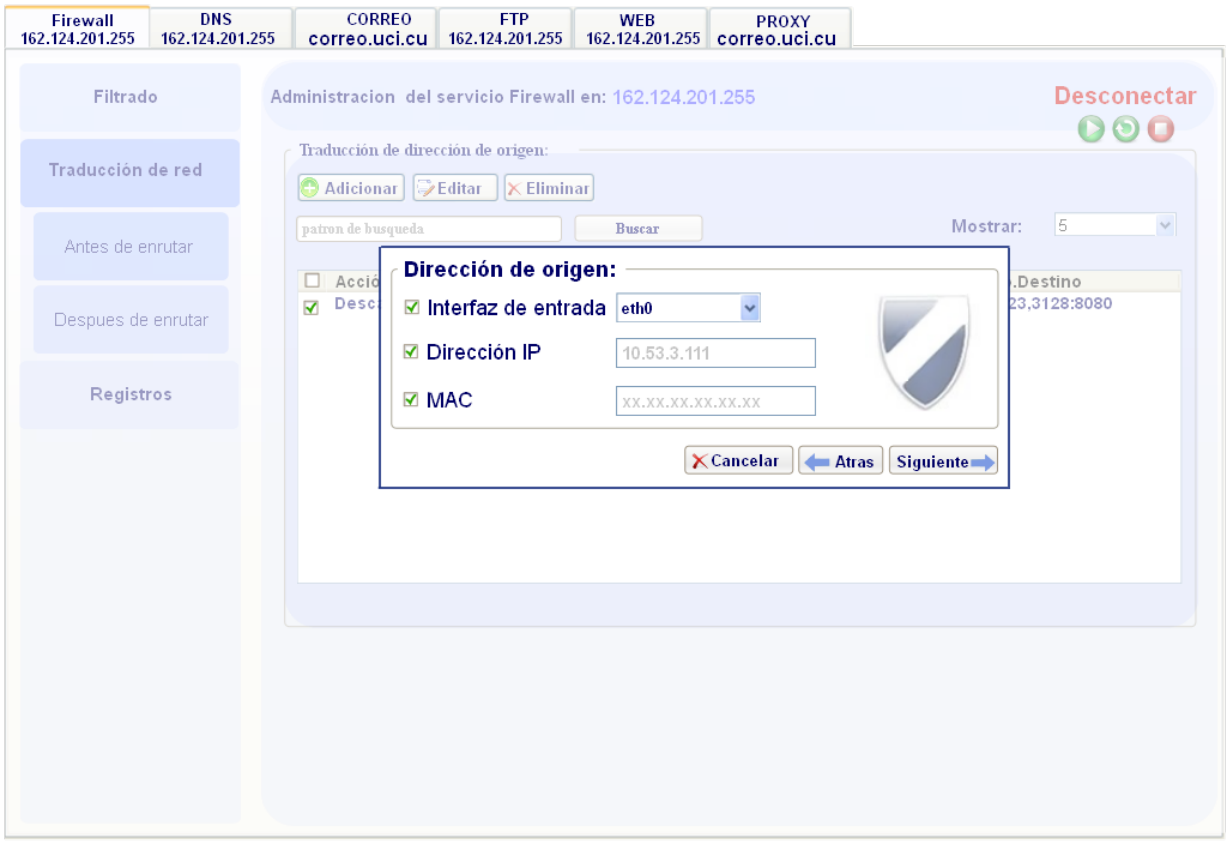
Anexo 42: Modificar: Protocolo y Puertos (Paquetes que aún no han sido enrutados)



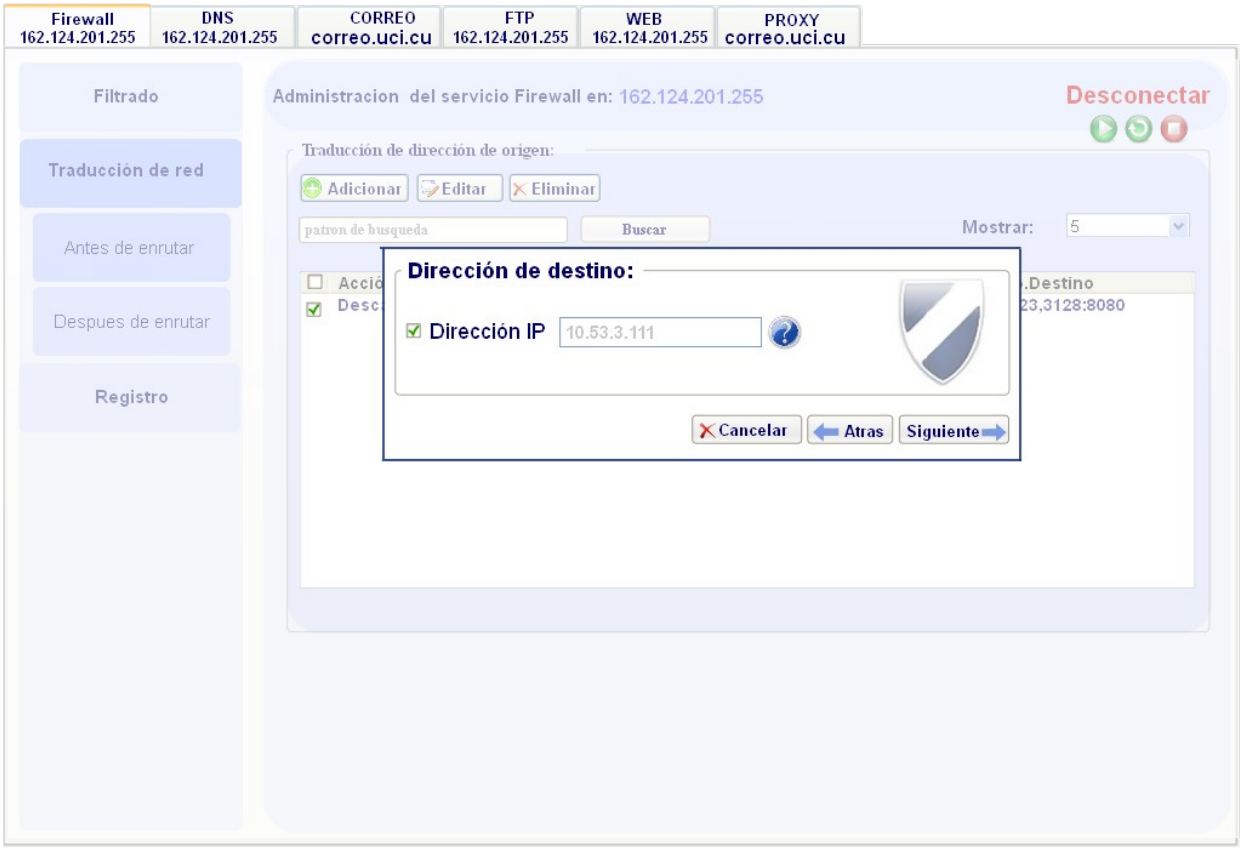
Anexo 43: Modificar: Descripción (Paquetes que aún no han sido enrutados)



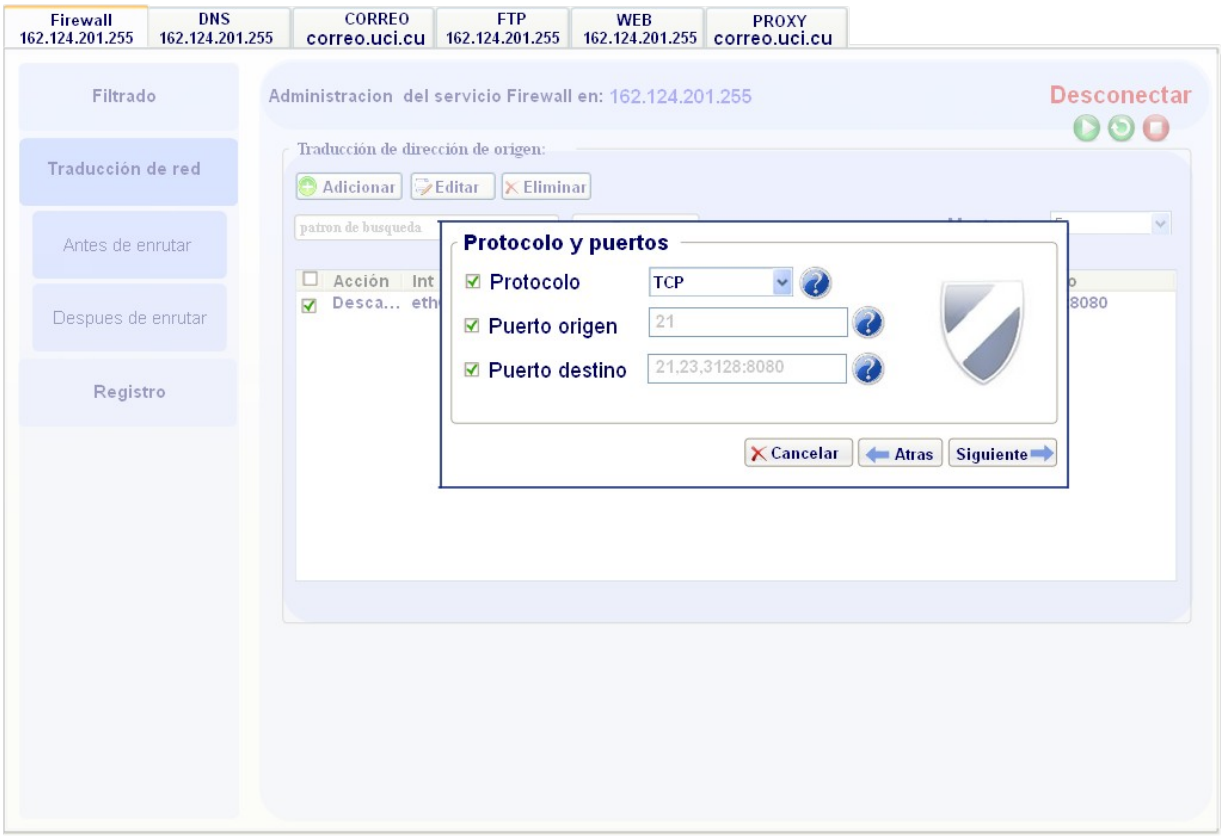
Anexo 44: Tipo de regla(Paquetes enrutados)



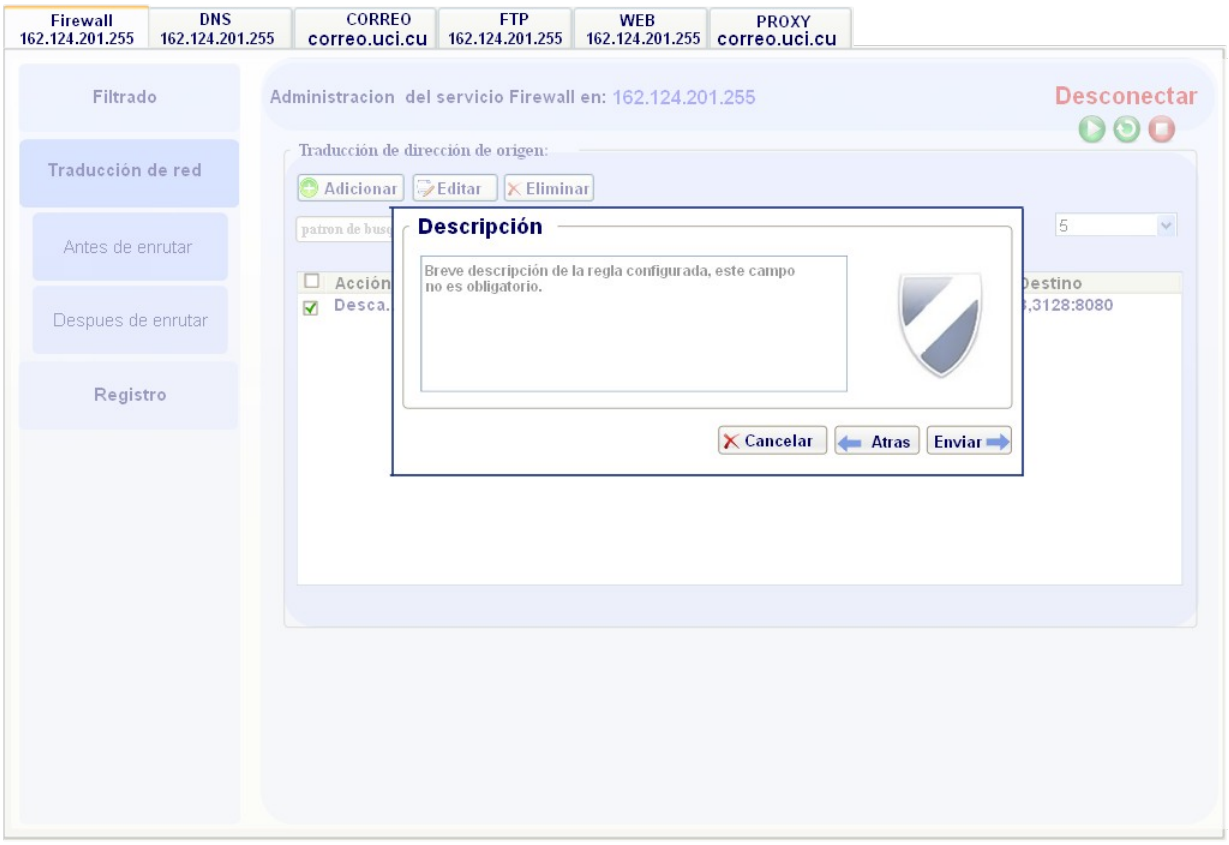
Anexo 45: Dirección de origen(Paquetes enrutados)



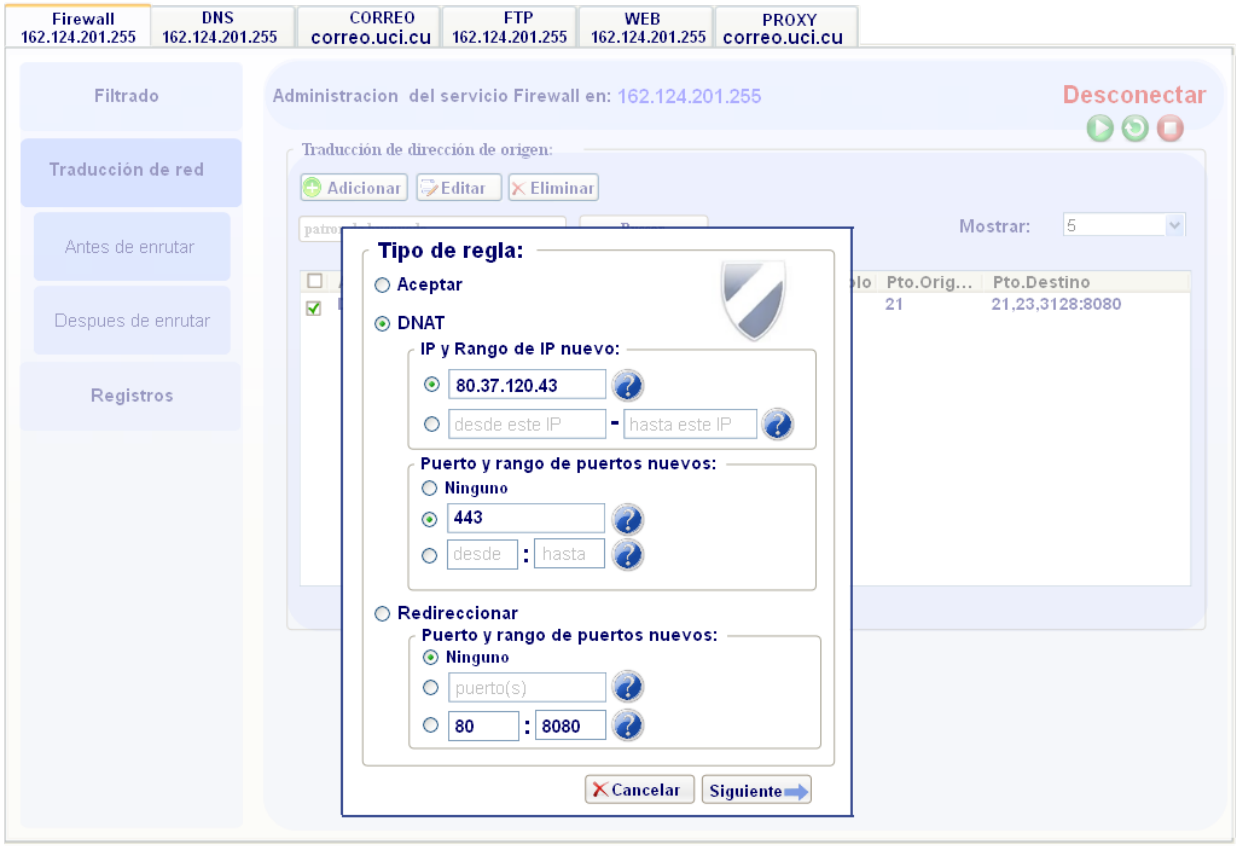
Anexo 46: Dirección de destino(Paquetes enrutados)



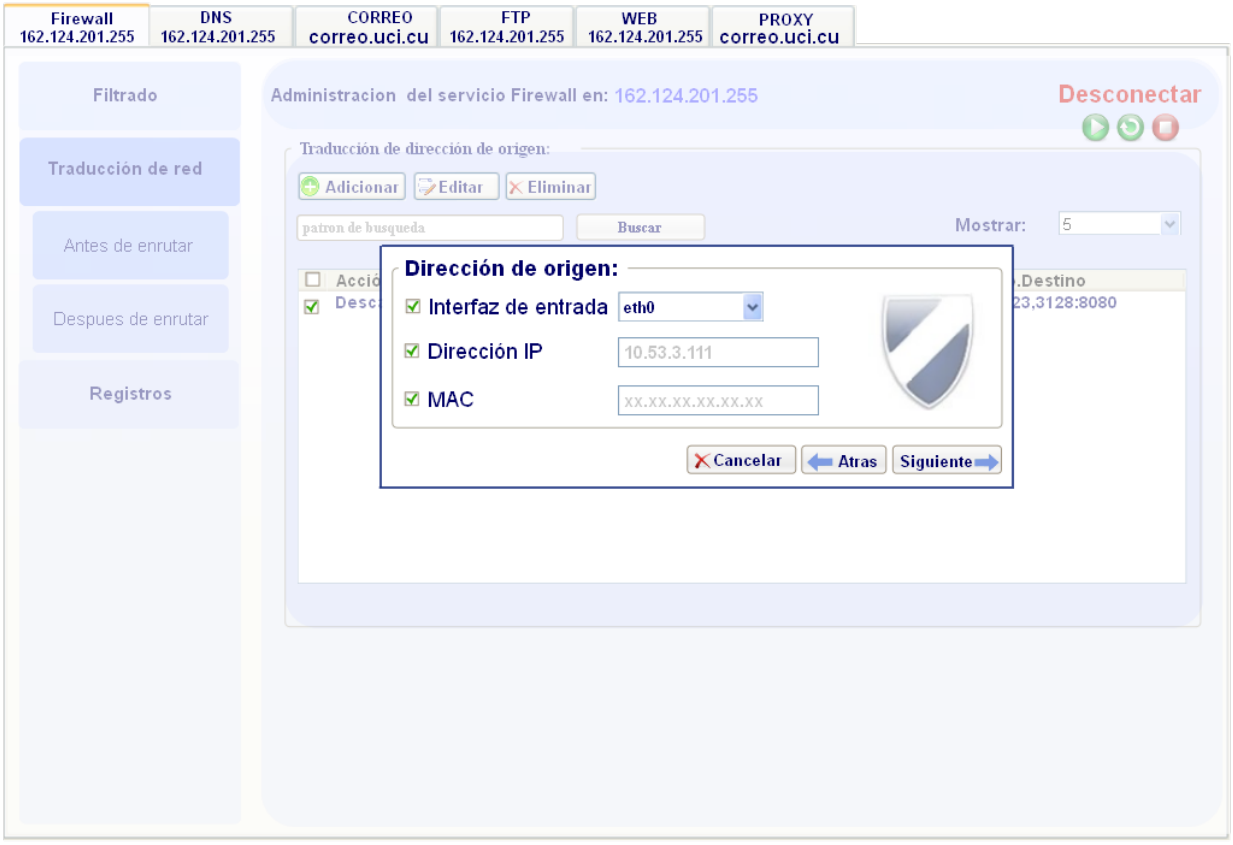
Anexo 47: Protocolo y Puertos(Paquetes enrutados)



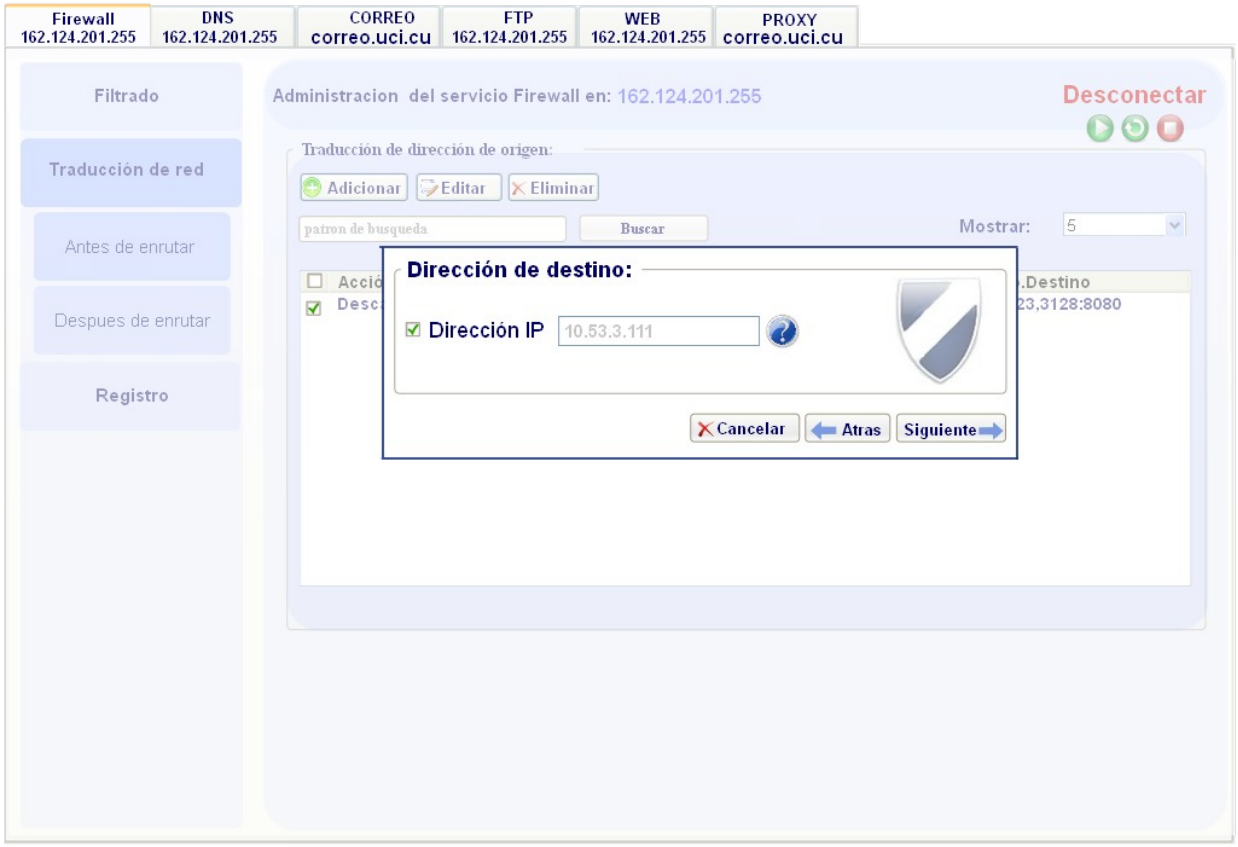
Anexo 48: Descripción(Paquetes enrutados)



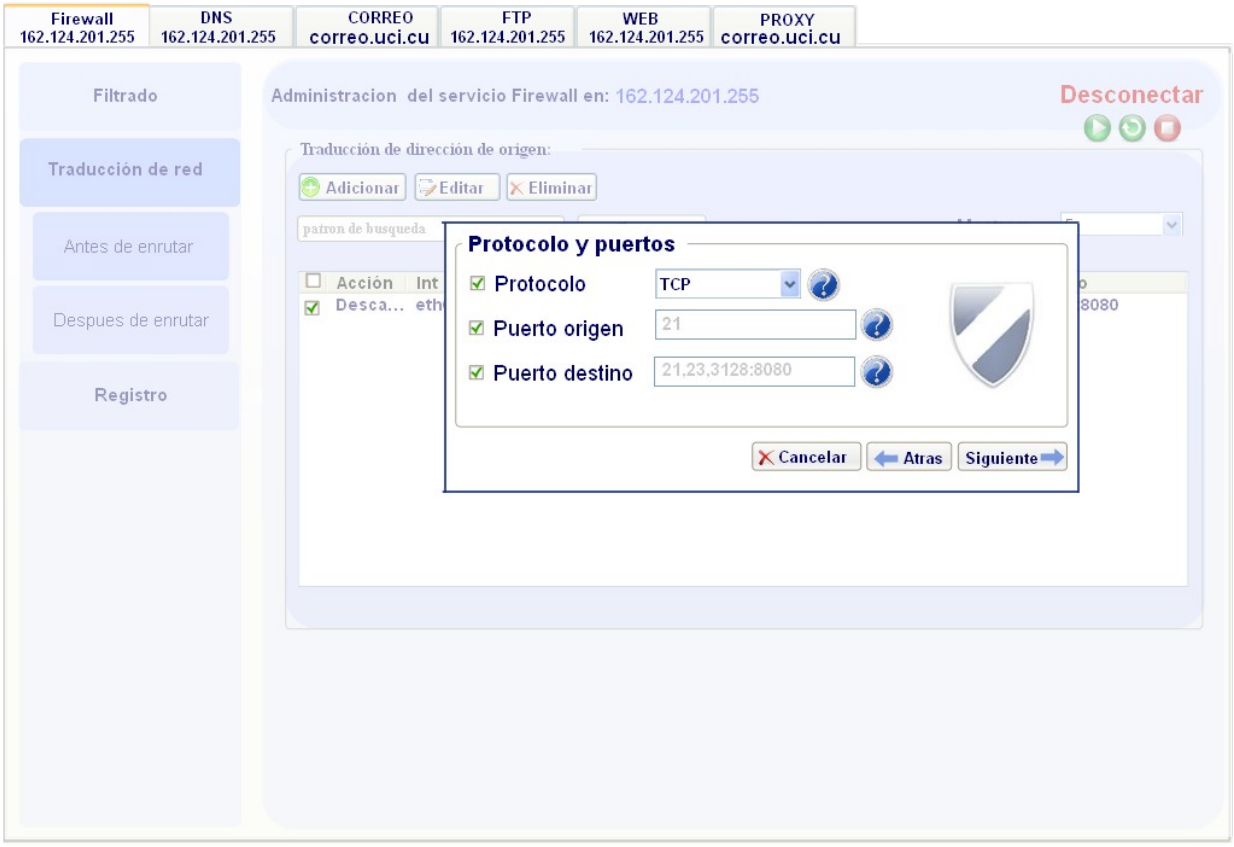
Anexo 49: Modificar: Tipo de regla (Paquetes enrutados)



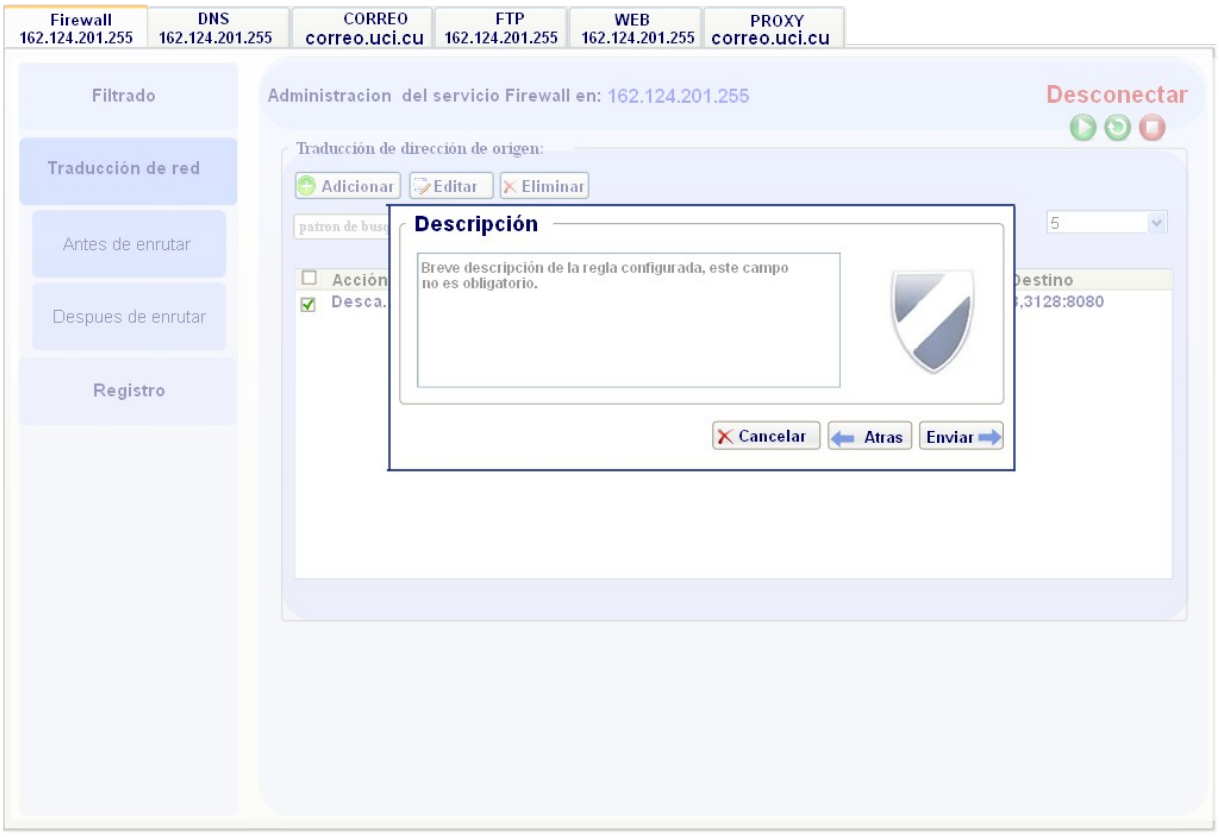
Anexo 50: Modificar: Dirección de origen (Paquetes enrutados)



Anexo 51: Modificar: Dirección de destino (Paquetes enrutados)



Anexo 52: Modificar: Protocolo y Puertos (Paquetes enrutados)



Anexo 53: Modificar: Descripción (Paquetes enrutados)

GLOSARIO DE TÉRMINOS

Router: Elemento Hardware que trabaja a nivel de red y entre otras cosas se utiliza para conectar una LAN a una WAN.

Crackers: persona que intenta acceder a un sistema informático sin autorización.

Dial-In: Conexión a internet que se establece a través de un módem y una línea telefónica. A cada usuario se le asigna un número IP dinámico, válido sólo durante la comunicación.

Modelo OSI: El modelo de referencia de Interconexión de Sistemas Abiertos (OSI, Open System Interconnection) fue el modelo de red descriptivo creado por la Organización Internacional para la estandarización lanzado en 1984. Es decir, fue un marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

Paquete IP: Cantidad mínima de datos que se transmiten en una red o entre dispositivos. Tiene estructura y longitud variable según el protocolo utilizado. En este caso dicho paquete utiliza el protocolo IP.

Host bastión: es una computadora que tiene al menos, una interfaz con la red de confianza y otra interfaz con la red de dudosa confianza.

Choke: se trata del modelo de cortafuegos más antiguo, basado simplemente en aprovechar la capacidad de algunos routers denominados screening routers para hacer un enrutamiento selectivo.

Espacio de Usuario: se refiere a un espacio de aplicación, típicamente en Unix o en sistemas operativos tipo Unix, el cual es externo al núcleo. Algunas veces la expresión espacio de usuario puede referirse a una aplicación que lleva a cabo sus propias llamadas al sistema o su propia entrada y salida (E/S), pero por lo común, el espacio de usuario como parte de una aplicación hará llamadas al sistema y otras actividades del sistema desde el núcleo.

NAT: es un mecanismo utilizado por routers IP para intercambiar paquetes entre dos redes que asignan mutuamente direcciones incompatibles.

GUI: interfaz gráfica de usuario.