Universidad de las Ciencias Informáticas

Facultad 1

“Módulo de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0”

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

Autor:

John Claro Andrade

Tutores:

Ing. Yaima Oval Riverón

Ing. Luis Daniel Sierra Corredera

La Habana, junio 2017

“Año 59 de la Revolución”

Declaración de autoría

Declaro por este medio yo John Claro Andrade, con carné 92062534243, ser el autor del presente Trabajo de Diploma titulado ‘‘Módulo de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0’’ y se reconoce a la Universidad de las Ciencias Informáticas, los derechos patrimoniales del mismo con carácter exclusivo.

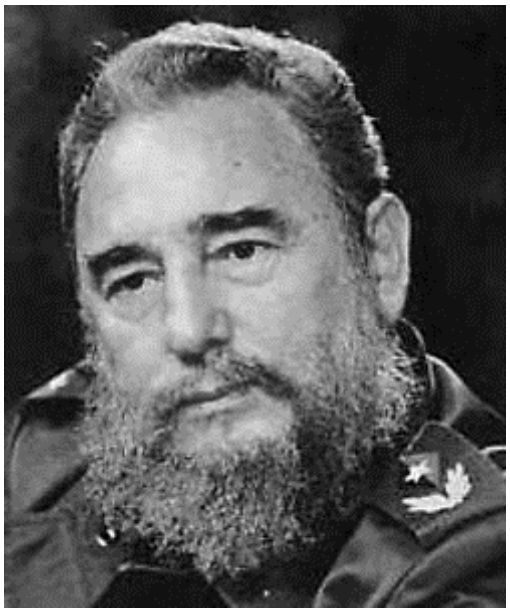
Para que así conste firmo la presente declaración jurada de autoría en La Habana a los _____ días del mes de _____ del año _____.

_____

John Claro Andrade

_____ _____

Ing. Yaima Oval Riverón Ing. Luis Daniel Sierra Corredera



«El éxito ha premiado nuestros esfuerzos».

Comandante Fidel Castro Ruz

Dedicatoria

Dedico el presente trabajo de diploma a mi mamá y a mi papá, quienes me han apoyado incondicionalmente a lo largo de mi vida y siempre me han impulsado a superarme. Son mi razón de ser, mi mayor orgullo, mi gran inspiración para luchar por lo que quiero. Muchas gracias mami y papi por todo ese apoyo que siempre me han dado, sin ustedes esto no hubiera sido posible. Los amo mucho a los dos.

A mi hermano, Ely y su familia.

Esta tesis va dedicada a toda mi familia.

Agradecimientos

A mis padres por todo el esfuerzo y sacrificio que han hecho para que me convierta hoy en un profesional.

A mi Ely, amistades y a mi hermano por estar siempre presentes en mi vida.

A Alfredo, Teresa y Yunier por apoyarme y ayudarme en el transcurso de la carreara.

A mis tutores Yaima y Luis.

A mis compañeros de la carrera: Alejandro Campo, Aldy, Enmanuel, Asney, Rubén Lage, Javier Hernández y compañeros de aula.

A las personas que me han ayudado a la realización de esta investigación: Yanet, Yasmary Prieto y a los profesores que a lo largo de la carrera han contribuido a mi preparación como ingeniero.

A los profesores del tribunal y el oponente que tanto trabajo les he dado revisando la tesis.

Resumen

En el centro de software libre, perteneciente a la Universidad de las Ciencias Informáticas, se desarrolla Nova, una distribución cubana de GNU/Linux. La versión 3.20.1 del centro de control de GNOME estará instalado por defecto en la distribución cubana GNU/Linux Nova Escritorio 6.0 como administrador de la configuración del sistema. El centro de control brinda centralización y acceso a diferentes funcionalidades y configuraciones para el trabajo con la distribución. En el mismo se ha detectado que existe una dependencia de software de terceros para la configuración de las reglas de un cortafuegos, por lo que no le facilita al usuario común, no experto en el campo de la informática el acceso a estas configuraciones. La presente investigación propone desarrollar un módulo que permita la configuración de las reglas de un cortafuegos, con el objetivo de perfeccionar este centro de control, para elevar el nivel de satisfacción de los usuarios con el uso el sistema operativo y proporcionar mejoras en proceso de migración. Se define como metodología de desarrollo AUP-UCI, la cual guiará el proceso de desarrollo de software. Para la implementación de la solución propuesta se define como entorno de desarrollo integrado Builder, como herramienta de modelado Visual Paradigm y como lenguajes de programación C y XML. Finalmente se obtuvo como resultado práctico de la investigación, un módulo con una interfaz de usuario, el cual permite la configuración de un cortafuegos en el centro de control de GNOME 3.20.1.

Palabras clave: centro de control, GNOME, migración, módulo, Nova.

Índice

Índice de ilustraciones

Índice de tablas

Introducción

El proceso de inserción de Cuba, en la expansión del uso de las tecnologías de información y las comunicaciones (Tic), ha sido un proceso paulatino, promovido y acompañado por el estado cubano, en los cuales se destaca su dimensión educativa y la comprensión de las mismas como un medio de impulso a los procesos de desarrollo económico. Una revisión de este proceso histórico, se pone de frente, al menos, a dos realidades necesarias; una es, que el inicio del proceso de desarrollo de estas tecnologías, (a escala mundial y ritmo vertiginoso), se remonta a la década de los setenta en el siglo pasado en los Estados Unidos de América (Hafner, 1998), y por ende, en el proceso de competencia de desarrollo tecnológico es considerable, más cuando en esta dinámica hay que reconocer la historia de relaciones políticas injerencistas, que han limitado considerablemente el acceso de Cuba a estos bienes.

Una segunda realidad, es que el cambio de la dinámica socioeconómica cubana, donde el nuevo actor cuentapropista, cooperativista o propio de otras formas de gestión económicas no estatales también se sirve de estos recursos para la ejecución de sus funciones y siendo un usuario final, precisa de que se le haga factible y eduque en el uso de los medios de protección y seguridad informática. La motivación para la selección del objetivo de investigación se encuentra en un análisis de contexto que presenta los siguientes términos a tener en cuenta:

- La expansión y promoción del uso de las tecnologías informáticas, tanto en el desarrollo de la dinámica económica del país, como en el uso cotidiano de la sociedad.
- La intencionalidad de insertar y generalizar el uso de la distribución cubana GNU/Linux Nova, por su condición de software libre, (representa un posicionamiento político ante el carácter hegemónico de la expansión y distribución de otros sistemas operativos privativos como lo es *Microsoft Windows*) como una opción a la autonomía y la soberanía informática.
- La comprensión de que la propuesta de la generalización del uso de estas tecnologías se pone frente al reto, de que el usuario común, sujeto no profesional de estas ciencias, precisa de la viabilidad en su interacción con el soporte tecnológico, donde por consiguiente la interfaz entraría a jugar su papel como puente idiomático entre el lenguaje técnico de la programación y las necesidades operativas del usuario.

Nova es la distribución cubana de GNU/Linux desarrollada en la Universidad de las Ciencias Informáticas (UCI), para facilitar la migración al código abierto y el software libre en Cuba y garantizar la soberanía tecnológica. Actualmente se encuentra en desarrollo la versión de Nova 6.0, que se realiza con el objetivo de avanzar en el proceso de migración. En ella estarán instaladas aplicaciones por defecto como el entorno de escritorio de GNOME 3.20.1 y su respectivo centro de control. Este centro de control brinda centralización y acceso a diferentes funcionalidades y configuraciones para el trabajo con la distribución, adaptándose a las

necesidades del usuario, lo que facilita su uso.

El diseño del centro de control de GNOME 3.20.1, provoca que usuarios procedentes de sistemas operativos privativos no se adapten fácilmente al cambio; afectando el proceso de migración al software libre y dificultando las acciones de capacitación y entrenamiento. Un ejemplo de ello se evidencia en que para la configuración de las reglas de un cortafuegos, se hace necesario tener conocimientos avanzados en el campo de la informática y redes. Un cortafuegos es un sistema de seguridad que permite o deniega ciertos accesos que puedan suponer una amenaza en la red (Ziegler, 2001). En las distribuciones libres para la configuración de un cortafuegos solo se pueden realizar mediante líneas de comandos o creando dependencia de otras aplicaciones externas que permitan la configuración de las reglas del cortafuegos mediante interfaces de usuario. Un usuario común no tiene el tiempo, interés o la paciencia para aprender todos los aspectos necesarios de la seguridad, por lo que es muy importante que dicho usuario pueda contar rápidamente con medidas de seguridad, sin necesidad de convertirse en un experto de redes.

Teniendo en cuenta lo anteriormente planteado se define como **problema de investigación:** ¿Cómo perfeccionar el funcionamiento del centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0?

Para la resolución del problema científico se asume como **objeto de investigación**, el proceso de configuración de las reglas de un cortafuegos en los centros de control, teniendo como **campo de acción** la configuración de las reglas de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.

Para resolver el problema se plantea como **objetivo general:** Desarrollar un módulo en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0 que permita la configuración de las reglas del cortafuegos.

Para lograr un mejor desarrollo se plantean las siguientes **preguntas científicas.**

1. ¿Cuáles son los supuestos teóricos que sustentan la implementación de un módulo para la configuración de las reglas de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0?
2. ¿Qué características presenta el análisis y diseño del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0?
3. ¿Cómo validar la contribución del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0?

Con el propósito de darle cumplimiento a las preguntas científicas planteadas, se definen las siguientes **tareas de investigación:**

1. Recopilar las principales características de los centros de control de diferentes entornos de escritorios, distribuciones de software libres y sistemas privativos.
2. Comparar los sistemas de GNU/Linux y sistemas privativos con un centro de control que permita la configuración del cortafuegos.
3. Estudiar las principales herramientas que son utilizadas para la configuración de reglas de un cortafuegos en los sistemas de GNU/Linux.
4. Identificar los requisitos funcionales y no funcionales del módulo que permita la configuración de las reglas de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.
5. Diseñar el módulo en el centro de control que permita la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.
6. Implementar un módulo en el centro de control que permita la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.
7. Ejecutar los casos de prueba a las funcionalidades implementadas.

Los métodos científicos utilizados en el desarrollo de este estudio se describen a continuación:

Métodos teóricos:

- Histórico-lógico: Este método permitirá seguir la evolución de los centros de control desde su surgimiento y las herramientas que se utilizan para configurar un cortafuegos, además de profundizar en los elementos que predominan en su evolución y desarrollo.
- Analítico-sintético: Permitirá descomponer el problema en sus partes para un mejor análisis de los centros de control de diferentes distribuciones de GNU/Linux y sistemas privativos, además de las herramientas para la configuración de las reglas de un cortafuegos, con el objetivo de determinar cuáles son las características que debe tener la solución.

Métodos empíricos:

- Observación: Este método permitirá chequear el estado actual de varios centros de control GNU/Linux y sistemas operativos privativos más relevantes, a través de su instalación y pruebas.
- Tormenta de ideas: Este método permitirá el levantamiento de requisitos, así como para la determinación de las restricciones de software que debe tenerse en cuenta en el proceso de

desarrollo.

El presente trabajo estará estructurado de la siguiente forma:

Capítulo 1: La configuración de las reglas de cortafuegos desde los centros de control.

Se realiza un estudio acerca de los centros de control de los sistemas operativos más usados y las herramientas que administran los cortafuegos, así como sus principales características y funcionalidades. Además, se abordan conceptos claves que serán usados durante el desarrollo de la investigación y se describen las tecnologías, lenguajes de programación, metodología y herramientas utilizadas en el desarrollo del sistema.

Capítulo 2: Análisis y diseño del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.2.1 para la distribución GNU/Linux Nova Escritorio 6.0.

Haciendo uso de la metodología seleccionada, se propone una solución al problema planteado, realizando su análisis y diseño. Se exponen los requerimientos funcionales y no funcionales, así como principales resultados del proceso de ingeniería del software. Se explican el diagrama de componente y la arquitectura que se utilizó y el momento en que fueron empleados.

Capítulo 3: Validación del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.

Se elabora el plan de pruebas, y se realizan las mismas con el objetivo de comprobar y validar el correcto funcionamiento de los requisitos planteados.

Capítulo 1. La configuración de las reglas de cortafuegos desde los centros de control.

En este capítulo se crean las bases necesarias para el correcto entendimiento de conceptos relacionados en la expresión, “centro de control en sistemas GNU/Linux y privativos”. Se estudian las características que presentan los diversos centros de control y entornos de escritorio, enfocado esencialmente en la configuración de las reglas del cortafuegos.

1.1 Sistema operativo GNU/Linux

La sigla GNU significa: GNU *is not* Unix. En 1984, Richard Stallman fundó el proyecto GNU con el objetivo de conseguir un sistema operativo libre y abierto. Esto es, un sistema operativo tal, que los usuarios puedan usarlo, leer el código fuente, modificarlo y redistribuirlo. A partir de ese momento, un gran número de colaboradores se fueron sumando al proyecto, desarrollando software libre para reemplazar cada una de las herramientas del sistema Unix. La filosofía GNU apoya el crecimiento de la sociedad como un conjunto, haciendo especial énfasis en la valoración de las libertades personales, aun cuando esto puede estar en conflicto con intereses empresariales (Perpiñan, 2003).

1.2 Distribuciones GNU/Linux

Un sistema operativo basado en una distribución de GNU/Linux puede definirse como: “Una distribución de software basada en el núcleo Linux que incluye determinados paquetes de software para satisfacer las necesidades de un grupo específico de usuarios” (Koch, 2005). El código fuente del sistema GNU y del kernel[1] de Linux está accesible a todo el mundo, sin embargo, hacer funcionar un sistema a partir del código fuente es bastante difícil. Por eso, un sistema operativo GNU/Linux se distribuye en formato binario, es decir ya compilado.

1.3 Distribución cubana GNU/Linux Nova

Nova es la distribución cubana que utiliza el núcleo Linux e incluye determinados paquetes de aplicaciones informáticas para satisfacer las necesidades de la migración a plataformas de código abierto que experimenta Cuba, como parte del proceso de informatización de la sociedad. Su proceso de construcción, distribución y mantenimiento están enfocado a alcanzar niveles de excelencia en los siguientes aspectos (Fuentes, 2011):

- Seguridad: El modelo de desarrollo colaborativo, el acceso al código fuente y el exhaustivo proceso de revisión y auditoría de código garantiza un sistema seguro de virus y sin puertas traseras.
- Soberanía tecnológica: Mediante la formación de recursos humanos capacitados en esta distribución, con capacidad de decisión de las tecnologías utilizadas y desarrolladas.
- Socio-adaptabilidad: Es un sistema operativo hecho por cubanos para cubanos, alineado a las

[1] *Es un software que constituye una parte fundamental del sistema operativo, y se define como la parte que se ejecuta en modo núcleo.*

políticas que orienta la informatización nacional y optimizada para las condiciones tecnológicas del país.

- Sostenibilidad: Mantendrá un proceso flexible y versátil, en constante innovación y consonancia con las nuevas tendencias tecnológicas internacionales, garantizando modelos de comercialización que permiten el ingreso de divisas, por el concepto de exportación de productos y servicios.

1.3.1 Nova Escritorio

Desde el año 2009 hasta la actualidad se han creado cinco versiones de Nova, en las que incorpora tres variantes: Nova Escritorio, Nova Servidor y Nova Ligero. Nova Escritorio es la edición estándar de Nova dedicada a los ordenadores personales. La versión 6.0 cuenta con el entorno de escritorio GNOME en su versión 3.20.1 para usuarios con experiencia en Linux o nuevos entusiastas que disfruten de las bondades que presenta este entorno (Rodriguez, 2012).

1.4 Centro de control de sistemas GNU/Linux

Desde la creación de los centros de control siempre vienen asociados a una distribución GNU/Linux. Perfeccionándose cada día más, con la integración de nuevos módulos que permitan la configuración de funcionalidades principales. Mediante la interfaz gráfica los usuarios pueden administrar todo lo relacionado con la configuración en cuanto a apariencia, red, antivirus, periféricos, dispositivos conectados, todo lo referido con el control total de software y hardware (Fuentes, 2011).

Con la creación de los sistemas operativos, sus centros de control han sido parte inherente a ellos, convirtiéndolos en un acontecimiento cotidiano, comenzando a ser parte importante del modo de vida de las personas. El centro de control constituye una herramienta que facilita la configuración del sistema operativo de acuerdo a gustos, preferencias y necesidades de los usuarios. Esta herramienta permite que el usuario común se encuentre con todas las configuraciones del ordenador en un solo lugar. Dentro de las configuraciones que se deben ejecutar se encuentran las asociadas a un cortafuegos, una de las más importantes; ya que está relacionada con la protección de la computadora de los ataques externos de la red. La seguridad es una cuestión especialmente importante para los usuarios de Linux con conexiones directa a la red. Su propósito fundamental y su filosofía fue el poder compartir información en un entorno de investigación y desarrollo.

1.5 Cortafuegos

Un cortafuegos (*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* [2] 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

2 Es un dispositivo que proporciona conectividad a nivel de red o nivel tres en el modelo OSI.

seguridad. El espacio protegido, es denominado perímetro de seguridad, y la protección se realiza contra una red externa, no confiable, llamada zona de riesgo (Ziegler, 2001).

Para el autor, se trata de cualquier sistema empleado para separar un sistema cómputo del resto, permitiendo o denegando servicios y protocolos que puedan suponer una amenaza a la seguridad desde el exterior. Por tanto, un usuario normal podría bloquear direcciones de red que quieran acceder a la computadora de forma remota. Para mayor entendimiento ver la ilustración 1.

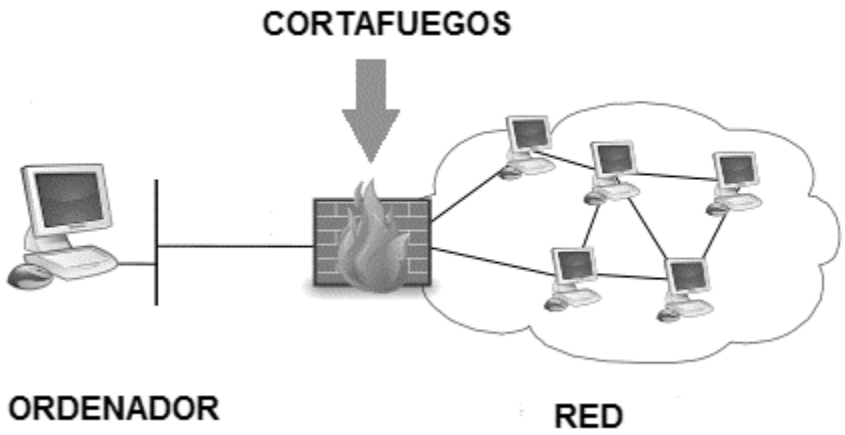


Ilustración 1: Gráfica del cortafuegos (OSI, 2016).

1.5.1 Características de los cortafuegos

Principales características:

1. **Protección de la red:** Mantiene alejados a los intrusos 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 intruso estuviera dentro de la misma o de la organización.

2. **Control de uso de internet:** Permite bloquear el acceso a la información no adecuada, determinar que sitios puede bloquear el usuario y llevar un registro.

3. **Concentra la seguridad:** El cortafuegos facilita vigilar y mantener un monitoreo de seguridad, dado que su máxima preocupación es la de encarar los ataques externos.

4. **Control y estadísticas:** Permite controlar el uso de la red en el ámbito interno y conocer los intentos de conexiones desde el exterior y detectar actividades sospechosas.

5. **Localización de un cortafuegos:** Un cortafuegos se encuentra frecuentemente instalado en cualquier punto donde una red interna esté establecida. Todo tráfico en la red interna pasa a través del cortafuegos, así puede determinar si dicho tráfico es aceptable de acuerdo con sus políticas de seguridad (Goncalves, 2016).

1.6 Estudio de sistemas homólogos

Conocidos los conceptos de centros de control y cortafuegos, se hace necesario analizar algunos de los principales centros de control que se utilizan en las distribuciones. Se han escogido para el estudio los siguientes centros de control por pertenecer a las distribuciones GNU/Linux más usadas (Metaphor, 2017). El estudio de estos centros de control ha sido realizado enfocándose en los elementos que posibilitan la configuración de los cortafuegos.

1.6.1 Centro de control de KDE

El proyecto KDE fue iniciado en octubre de 1996 por el programador alemán Matthias Ettrich, quien buscaba crear una interfaz gráfica unificada para sistemas Unix. En sus inicios imitó a CDE (*Common desktop environment*), un entorno de escritorio utilizado por varios usuarios de Unix. El centro de control KDE, también conocido como KControl, es el administrador de la configuración para el entorno de escritorio KDE, ver ilustración 2. KControl tiene una arquitectura modular y es parte del paquete kdeadmin[3]. Aunque KControl era instalado por defecto en Kubuntu, esta distribución GNU/Linux utilizaba una versión no estándar, que fue rediseñada para ser similar al administrador de las configuraciones de Mac OS X. Más tarde este configurador fue integrado en KDE 4 con el nombre de *System Settings* (KDE, 2016).

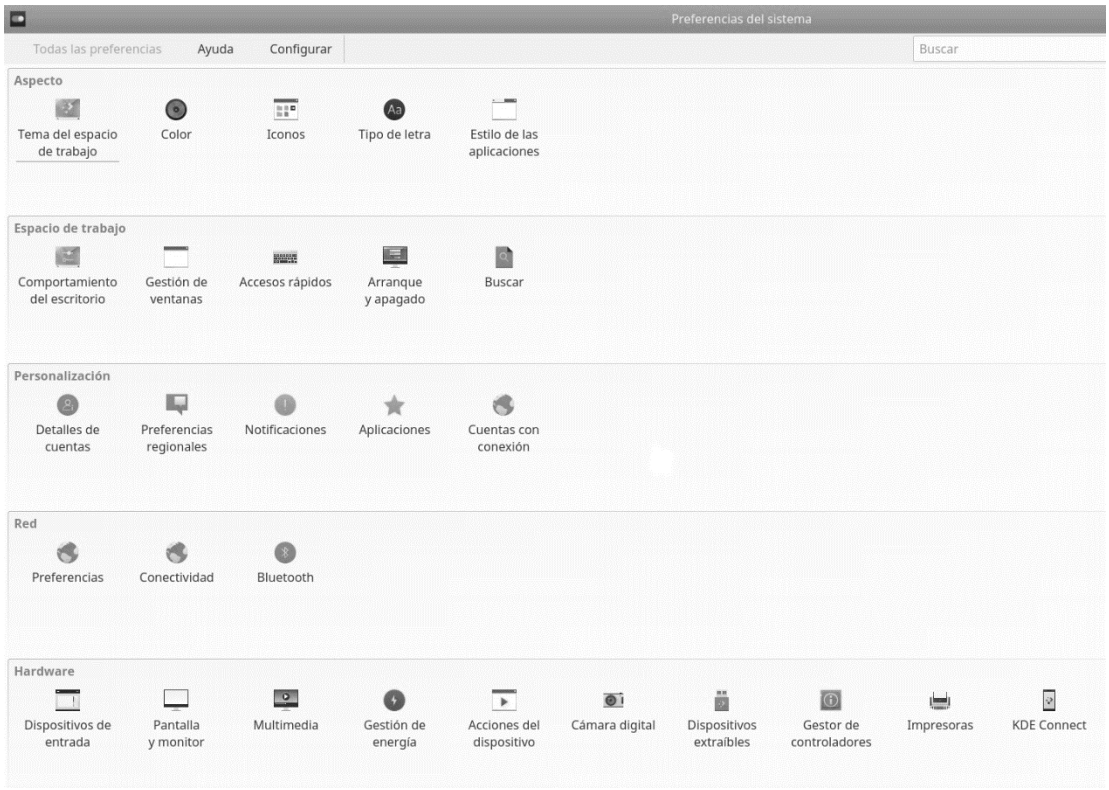


Ilustración 2: Centro de control de KDE. Ilustración capturada del entorno de escritorio KDE.

Su interfaz gráfica está dividida en cinco sesiones: aspecto, espacio de trabajo, personalización, red y

3 Es un paquete de KDE que contiene herramientas administrativas.

hardware. Como se puede apreciar dentro del centro de control de KDE no existe un módulo que en su funcionalidad permita configurar las reglas de un cortafuegos. En la sesión de red que es donde debería estar el módulo, solo se pueden configurar direcciones proxy[4], direcciones *internet protocol* (IP) o trabajar con bluetooth[5].

1.6.2 Centro de control de Unity

En mayo de 2010, Mark Shuttleworth anunció Unity, una interfaz de usuario para el escritorio de Ubuntu. El centro de control de Unity está basado en el centro de control de GNOME, debido a que, el entorno de escritorio Unity es realizado por Canonical, ver ilustración 3. El centro de control de Unity es la aplicación para ajustes del sistema, que recoge los principales elementos de configuración de Ubuntu. El mismo está dividido en tres sesiones: personal, hardware y sistema, cada parte tiene sus respectivos módulos (Unity, 2010).

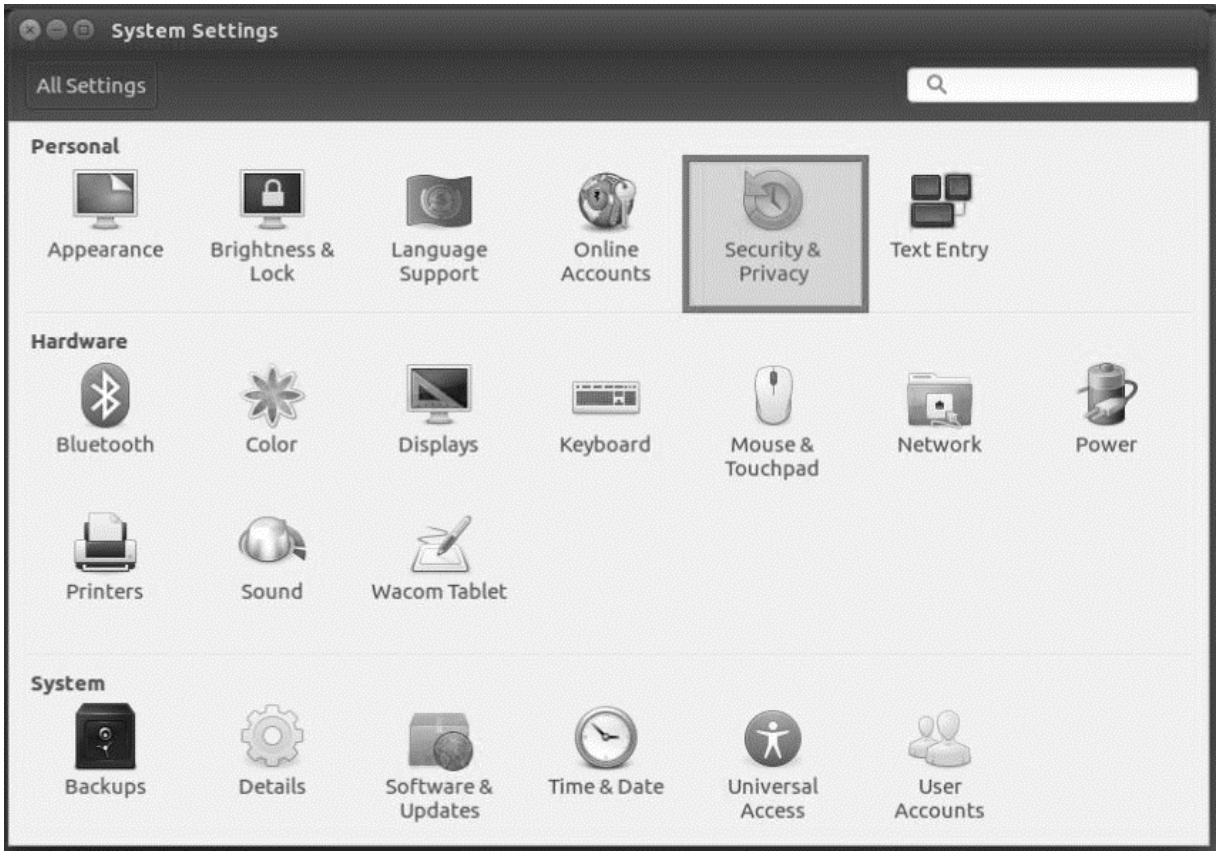


Ilustración 3: Centro de control de Unity. Ilustración tomada del entorno de escritorio Unity. Elaboración propia.

Como se puede apreciar dentro del centro de control de Unity, no existe un módulo que permita configurar las reglas de un cortafuegos. Y dentro del módulo seguridad y privacidad, se puede apreciar que no tiene la funcionalidad de configurar las reglas de un cortafuegos, ver ilustración 4.

4 en una red informática, es un servidor, programa o dispositivo, que hace de intermediario en las peticiones de recursos que realiza un cliente a otro servidor.

5 Es una especificación para redes inalámbricas de área personal.

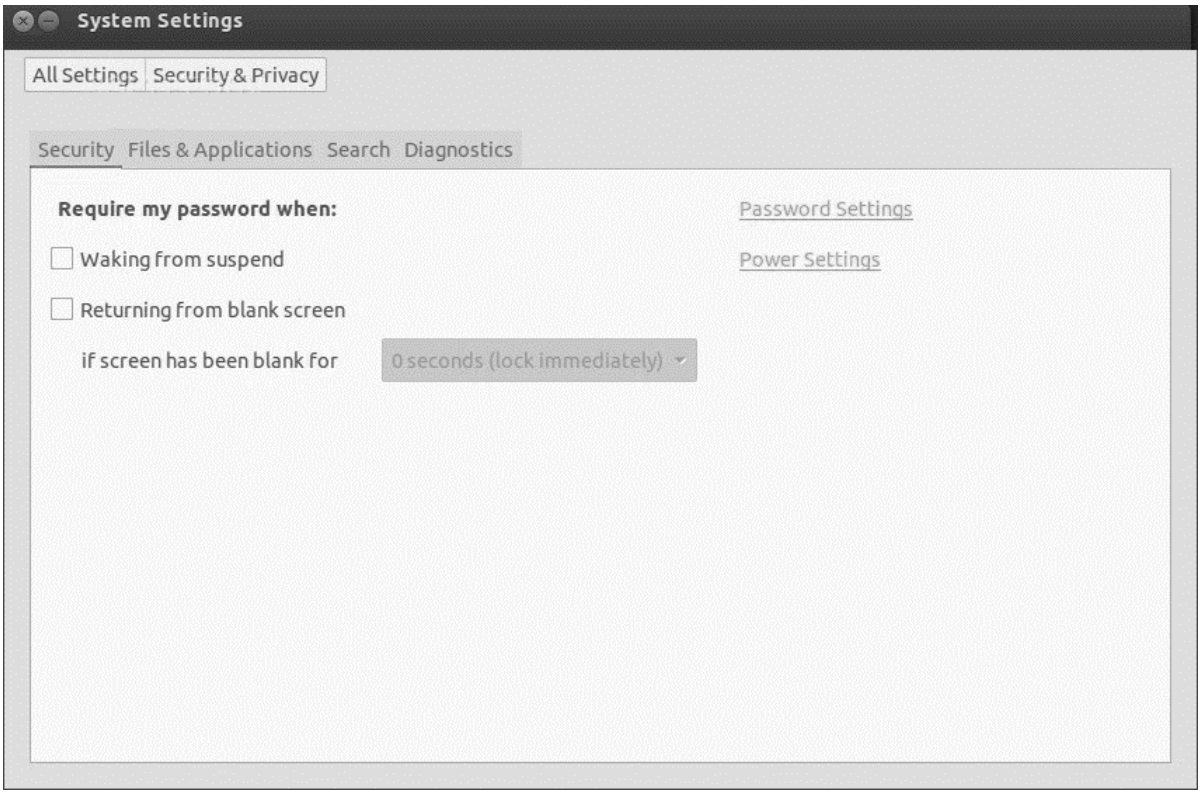


Ilustración 4: Centro de control de Unity. Ilustración tomada del entorno de escritorio Unity.

1.6.3 Centro de control Yast

YaST es la herramienta de instalación y configuración de openSUSE y de la distribución SUSE Linux Enterprise. Su historia se remonta desde los inicios de la distribución. Yast se encuentra entre las herramientas más potentes y de uso más sencillo para la administración de sistemas GNU/Linux. Yast es un acrónimo del inglés *yet another setup tool* lo que se podría traducir como, otra herramienta de configuración (Yast, 2015).

Su interfaz gráfica está dividida en dos partes, en la izquierda se muestran ocho sesiones, software, hardware, sistema, servicio de red, seguridad y usuarios, virtualización, asistencia y miscelánea. Y a la derecha los módulos de la sesión seleccionada, ver ilustración 5.

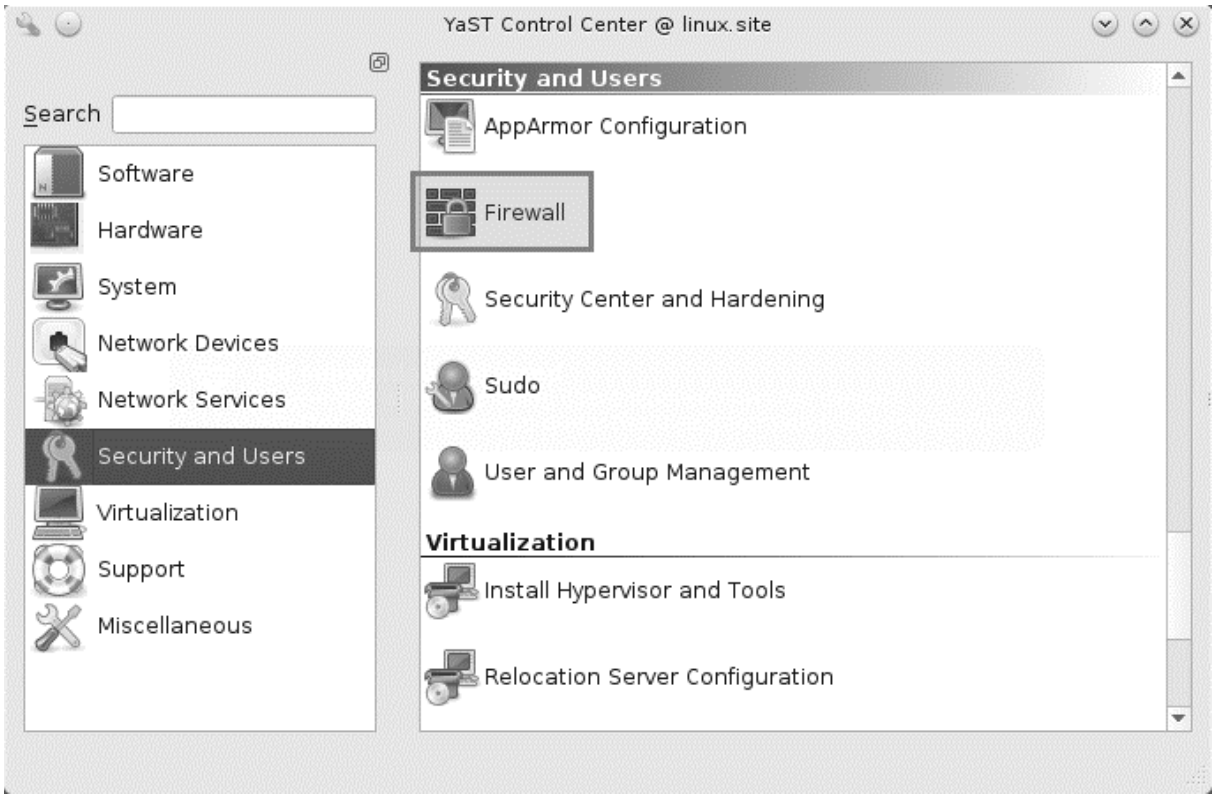


Ilustración 5: Centro de control Yast. Ilustración tomada de la distribución openSUSE.

Dentro de la sesión seguridad y usuarios se encuentran la opción (*firewall*) que permite al administrador del sistema configurar las reglas de un cortafuegos. Este cortafuegos permite al usuario trabajar con diferentes funcionalidades en sus interfaz como son: inicio del servicio, interfaces, servicios autorizados, enmascaramiento, configuración de difusión, nivel de registro, reglas personalizadas (red de origen, protocolo, puerto destino y puerto de origen).

1.6.4 Centro de control de GNOME

El proyecto GNOME provee un entorno de escritorio intuitivo y atractivo, poniendo énfasis en la simplicidad, usabilidad y eficiencia. El proyecto fue iniciado por los mexicanos Miguel de Icaza y Federico Mena como una alternativa a KDE, lanzando su primera versión en marzo de 1999. El entorno de escritorio GNOME es bastante configurable: se pueden configurar los menús, los iconos, las tipografías, el fondo, el protector de pantalla, el tema, el administrador de ventanas, sonido, la interacción con las ventanas y muchos otros detalles de acuerdo al gusto del usuario (Rodriguez, 2012).

El centro de control de GNOME, conocido como el paquete gnome-control-center, es la aplicación instalada por defecto en el entorno de escritorio de los en sistemas operativos como Ubuntu y las distribuciones de Nova Escritorio, su objetivo es administrar la configuración centralizada del entorno de GNOME.

Centro de control de GNOME 3.20.1

GNOME-control-center 3.20.1 es el administrador de la configuración del entorno de escritorio de GNOME 3.20.1. Está dividido en tres sesiones, con una arquitectura modular, y cada módulo dentro de su respectiva

sesión. El mismo está disponible para el trabajo en la configuración de la distribución, ver ilustración 6. Su código fuente está implementado en el lenguaje C y extensible markup language (XML) en las interfaces de usuario.

Ilustración 6: Centro de control de GNOME. Ilustración tomada del entorno de escritorio de GNOME.

Como se puede apreciar dentro del centro de control de GNOME no existe un módulo que permita configurar las reglas de un cortafuegos. El módulo de privacidad solo es para establecer contraseñas y el de red para configurar las direcciones de redes.

1.6.5 Centro de control de *Windows*

El panel de control de *Windows* aunque no es una solución por ser privativo, el autor considera que se debe estudiar porque pueden aportar elementos para la solución, ver la ilustración 7. El panel de control ha sido una parte inherente del sistema operativo de *Microsoft Windows* desde su lanzamiento (*Windows* 1.0), presentado en noviembre de 1985. El panel de control de la interfaz de usuario de *Windows* permite a los usuarios que vean y manipulen ajustes y controles del sistema básico, tales como agregar nuevo hardware, agregar o quitar programas, cuentas de usuario y opciones de accesibilidad entre otras opciones de sonidos

y pantalla.

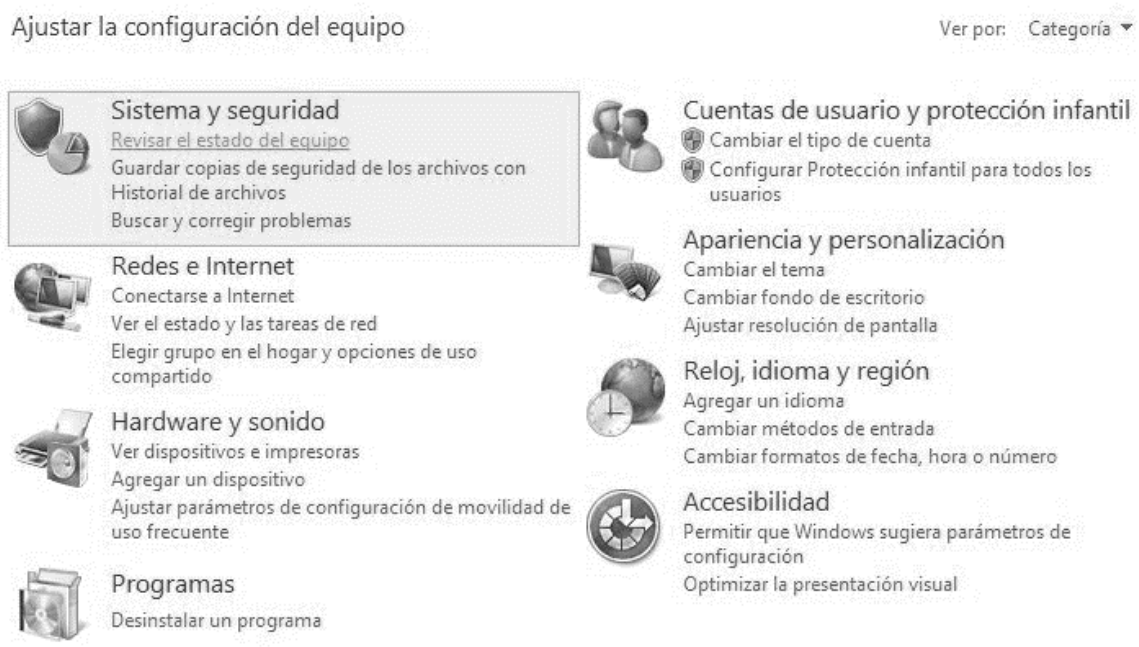


Ilustración 7: Panel de control de Windows. Ilustración tomada de la interfaz de escritorio de Windows.

Como se puede apreciar dentro del panel de control de *Windows,* en sistemas y seguridad, existe una opción que permite configurar las reglas de un cortafuegos. El cortafuegos de *Windows* puede apagarse o encenderse según el usuario lo considere, dejando reglas por defecto. Además, brinda la posibilidad de agregar, eliminar o editar reglas. Pero le falta un gestor de perfiles, para cuando se quiera tener reglas predeterminadas por perfiles, ver ilustración 8.

Ilustración 8: Panel de control de Windows. Ilustración tomada de la interfaz de escritorio de Windows.

1.6.6 Centro de control de *Mac OS X*

El centro de control de Mac OS X se llama “Preferencias del sistema”, ver la ilustración 9. El mismo está compuesto por cuatro sesiones en las que se pueden realizar todas las configuraciones necesarias para empezar a usar este sistema operativo.

Ilustración 9: Panel de control de Mac OS X. Ilustración tomada de la interfaz de escritorio de Mac OS X.

En el centro de control de *Mac OS X* tiene la funcionalidad de ‘‘seguridad y privacidad’’ en la cual el usuario puede configurar reglas de un cortafuegos y gestionar los permisos a las aplicaciones que tengan acceso a la red, ver ilustración 10.

Ilustración 10: Panel de control de Mac OS X. Ilustración tomada de la interfaz de escritorio de Mac OS X.

Se realizó una comparativa entre los centros de control estudiados, con el objetivo de identificar si en los mismos existe una funcionalidad para la configuración de las reglas de un cortafuegos en los sistemas operativos estudiados. Ver tabla 1.

Tabla 1: Comparativa entre centros de control de sistemas operativos libres y privativos. Elaboración propia.

Sistema operativo	**Centro de control**	**Funcionalidad de cortafuegos**
Distribución GNU/Linux Kubuntu	KControl	no
Distribución GNU/Linux Ubuntu	Unity-control-center	no
Distribución GNU/Linux openSUSE	Yast	si
Distribución GNU/Linux	Gnome-control-center	no

Ubuntu-GNOME		
Microsoft Windows	Panel de control	si
Max OS X	Preferencias del sistema	si

1.7 Resultados del estudio de los centros de control

Como resultado del estudio de los diferentes centros de control, se obtuvo que solo en el centro de control Yast de la distribución GNU/Linux openSUSE, existe un módulo que permite la configuración de las reglas de un cortafuegos. Se tomaron ideas de este centro de control, asociadas a la personalización de las reglas para el manejo de la configuración desde una interfaz de usuario. No se utiliza como opción ya que la propuesta de solución es desarrollar un módulo en el centro de control gnome-control-center para GNOME 3.20.1 que permita la configuración de las reglas de un cortafuegos, y no utilizar otro centro de control como solución. El estudio de los centros de control de *Windows* y *Mac OS X* permitió analizar otras variantes de presentación de la configuración de las reglas de un cortafuego desde una interfaz de usuario.

En esta investigación, (por la extensión y profundidad propia de la misma), se refiere el hecho de los propósitos económicos que determinan el tipo de mercado, diferentes en uno y otro caso (el software privativo y el libre). En el primero la tendencia indica a crear la mayor cantidad de condiciones óptimas (entre las cuales destacamos centros de control, habilitados con mayor cantidad de módulos entre ellos los cortafuegos), para la expansión de su uso y por ende consumo, de ahí que el lenguaje propiciado entre máquina-usuario, impliquen, muy poca necesidad de nivel de especialización técnica.

El desarrollo de los software libre que desde su nacimiento parte de una concepción de resistencia y no desde el *marketing*[6], entiéndase que, el software privativo compite dentro del mercado por imponer el uso de sus propuestas en cada caso, lo que implica un continuo perfeccionamiento de sus características para tales fines. Las ventajas que genera el uso del software libre, básicamente son más perceptible por el usuario especializado en las tecnologías de las informáticas y las comunicaciones, quien por sus conocimientos pudiese explotar las potencialidades que genera la versatilidad de este tipo de software. El usuario común ante la intención de migrar del uso del software privativo a los libres continúa necesitando un proceso de capacitación especializado para la óptima explotación de sus potencialidades.

Todo esto implica y refuerza lo planteado anteriormente como objetivo general de la investigación en cuestión, donde desarrollar el módulo del cortafuegos en el centro de control es parte de un proceso estratégico mucho más amplio, para estimular y viabilizar la migración de usuarios de software privativo a software libre. Se asume que un margen importante en las estadísticas de migración se daría en los usuarios comunes, aquellos que utilizan la tecnología para la ejecución de tareas indirectas al tema y no fundamentalmente en

[6] *Es el proceso social y administrativo en el cual la organización identifica los objetivos, necesidades y deseos del mercado.*

usuarios especializados, que usan la tecnología como herramienta y fin en sí mismas. Para los usuarios especializados las diferentes interfaces solo cumplen el objetivo de agilizar los procesos y no necesariamente de accesibilidad, porque cuentan con recursos para hacer lo mismo por vía de líneas de comandos.

1.8 Interfaces gráficas para la configuración del cortafuegos

En las distribuciones GNU/Linux y específicamente en los centros de control no existe un módulo para configurar las reglas de un cortafuegos, pero si existen otras formas de poder configurarlo. Con la ayuda de herramientas gráficas si es posible, pero el usuario debe tener conocimiento de estas para poder descargarlas.

La GUI (por sus siglas en inglés *graphical user interface*) son aquellas que incluyen elementos como menús, ventanas, contenido gráfico, cursor y algunos otros sonidos que la computadora hace, y en general, todos aquellos canales por los cuales se permite la comunicación entre el ser humano y la computadora. Su principal uso consiste en proporcionar un entorno gráfico sencillo para permitir la comunicación con el sistema operativo de un ordenador (Royo, 2004).

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.

1.8.1 Modos de configuración de cortafuegos

Existen dos modos de configuración de cortafuegos:

Basado en ficheros de texto: es la de uso más extendido en lo que respecta a los cortafuegos de elaboración propia. Esta forma de configurar el cortafuegos permite al administrador editar un archivo específico donde puede introducir parámetros de configuración. Se trata de la interfaz más común para los administradores de los sistemas Unix tradicionales. 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 (Ziegler, 2001).

Basado en menú: presenta al usuario un menú basado en texto, iconos, imágenes y botones permitiendo que el usuario sepa con que está interactuando. Esta interfaz reduce la probabilidad de producirse errores; pero proporciona menor capacidad de control. Sin embargo, la posibilidad de error no queda totalmente excluida, dado que el usuario no siempre puede ver el efecto de algunos cambios. Esta interfaz es con la que se va a trabajar a la hora de desarrollar el módulo, para que un usuario común pueda interactuar con el módulo (Ziegler, 2001).

1.9 Herramientas para la configuración del cortafuegos

En este acápite se analizarán varias herramientas para la administración de cortafuegos con el objetivo de seleccionar las características de las mismas que puedan contribuir con el desarrollo de la solución al

problema de investigación. Para ayudar este proceso se hará una breve comparación de las mismas, en función de obtener información que pueda ayudar a implementar el módulo en el panel de control, basándose en la interfaz gráfica y las principales características de cada una de ellas.

Iptables

Iptables es una herramienta de cortafuegos disponible en el núcleo Linux que permite interceptar y manipular paquetes de red. No solamente permite filtrar paquetes, sino también realizar traducción de direcciones de red (NAT[7]) a IPv4[8] y mantener registros. Además, iptables permite al administrador, a través de reglas de configuración, definir políticas de seguridad para el tráfico que circula por la red. En todas las distribuciones de GNU/Linux mediante iptables se puede configurar las reglas de un cortafuegos, pero es necesario tener conocimiento avanzado en este tema ya que es muy difícil establecer las reglas. Su código fuente está escrito en C[9].

Iptables cuenta con una gama amplia de usuarios debido a que este, con respecto a su antecesor ipchains[10] 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 agregó el concepto de tablas. Otro aspecto a tener en cuenta es las configuraciones de las reglas de iptables es que, a pesar de ser difíciles de establecer, un usuario avanzado en el tema, puede llegar a crear una sólida barrera protectora (Vicedo, 2015). Iptables utiliza cuatro tablas distintas que almacenan las reglas que regulan tres tipos de operaciones sobre los paquetes:

- Filter se refiere a las reglas de filtrado (aceptar, rechazar o ignorar un paquete);
- NAT se refiere a la traducción de las direcciones de origen o destino y puertos de los paquetes;
- Mangle se refiere a otros cambios en los paquetes IP (incluyendo el campo ToS — tipo de servicio: «Type of Service» — y opciones);
- RAW permite otras modificaciones manuales en los paquetes antes de que lleguen al sistema de seguimiento de conexiones.

Shorewall

Es una excelente herramienta para realizar complejas configuraciones para la red, regula los paquetes de entrada y salida de las computadoras que viajan a través de la red, también se define como un cortafuegos y a su vez como una puerta de enlace con sus respectivos requisitos de las entradas y salidas de paquetes.

---

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

8IPv4: (internet Protocol versión 4 o Protocolo de internet versión 4): es la cuarta versión del protocolo Internet Protocol (IP), y la primera en ser implementada a gran escala.

[9] S*e trata de un lenguaje de programación de tipos de datos estáticos, de medio nivel, pero con muchas características de bajo nivel.*

[10] *Ipchains es un cortafuegos libre para Linux.*

(Shorewall, 2016).

- **Público objetivo:** 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, *Nat*, filtrado de paquetes.
- **Licencia:** GPLv2 Licencia pública general versión 2.
- **Código fuente:** Perl.
- **Interfaz gráfica:** No posee interfaz gráfica.

IPCop

Es para usuarios de pequeñas oficinas. Esta es una herramienta firewall, que requiere una PC independiente de baja potencia para ejecutar el software (IPCop, 2016).

- **Público objetivo:** Usuario común.
- **Principal funcionalidad:** Gestiona el simple filtrado de paquetes hasta la asignación de ancho de banda fijo a cada puesto de trabajo o la configuración de redes virtuales VPN.
- **Licencia:** Varias licencias públicas.
- **Código fuente:** Perl.
- **Interfaz gráfica:** Proporciona una simple interfaz web.

Forefront TMG

Microsoft forefront threat management gateway (TMG) es un completo *gateway*[11] de seguridad web desarrollado por Microsoft que ayuda a proteger a las empresas de las amenazas que existen actualmente en internet (Share, 2017).

- **Público objetivo:** Usuario común.
- **Principal funcionalidad:** Conjunto de herramientas de seguridad perimetral: *firewall*, VPN, análisis anti-malware, proxy, proxy inverso.
- **Licencia:** Privativa.
- **Código fuente:** C++.
- **Interfaz gráfica:** Interfaz gráfica intuitiva.

[11] *Es un protocolo mediante el cual se intercambia información de encaminamiento o ruteo entre sistemas autónomos.*

Gufw

Gufw (por sus siglas en inglés *graphic uncomplicated firewall*) es una interfaz gráfica para ufw (***u**ncomplicated **f**ire**w**all*) que simplifica complicados comandos de iptables para facilitarle el trabajo al administrador. Gufw proporciona un marco para la gestión de iptables, así como una interfaz de línea de comandos para manipular el servidor de seguridad. Además tiene como objetivo proporcionar una interfaz fácil de usar (GUFW, 2016).

- **Público objetivo:** Usuario común.
- **Principal funcionalidad:** Gestiona el tráfico entrante y saliente de paquetes, así como configuraciones de rangos de IP y de puertos.
- **Licencia:** Licencia pública general.
- **Código fuente:** Python[12].
- **Interfaz gráfica:** Interfaz gráfica intuitiva.

Con gufw se puede prescindir de la consola, usando el motor de ufw. Es más sencillo activar/desactivar el cortafuegos, añadir reglas que nieguen, permitan, limiten puertos y direcciones IP, además de borrar conjuntos de reglas creadas. Permite configurar el nivel de registro de ufw, y mantiene su propio registro de operaciones realizadas. Es ideal para personas sin conocimientos del funcionamiento de un cortafuegos.

Ufw es un cortafuegos diseñado para ser de fácil uso y por defecto viene instalado en todas las distribuciones de GNU/Linux, pero desactivado. Utiliza la línea de comandos para configurar iptables y crear reglas básicas, usando un pequeño número de comandos simples. Un ejemplo de esto véase la tabla 1.

Tabla 2: Comandos de línea usados en UFW. Elaboración propia. (Ubunto, 2017).

Función	Comandos	Comentario
Enciende el firewall	sudo ufw enable	
Apaga el firewall	sudo ufw disable	
Muestra estatus	sudo ufw status	
Muestra estatus detallado	sudo ufw status verbose	
Muestra lista de reglas numerada	sudo ufw status numbered	
Bloquea todo el tráfico de entrada	sudo ufw default deny incoming	Esto detendrá por omisión todo el tráfico entrante LAN desde internet a menos que se especifique lo contrario en una regla
Permite todo el tráfico de entrada	sudo ufw default allow incoming	Esto permitirá por omisión todo el

[12] *Es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.*

		tráfico entrante LAN a menos que se especifique lo contrario en una regla
Bloquea todo el tráfico saliente	sudo ufw default deny outgoing	Esto detendrá por omisión todo el tráfico saliente hacia internet LAN a menos que se especifique lo contrario en una regla
Permite el puerto 53 en el UDP	sudo ufw allow port 53	Esto permitirá a cualquiera en internet hacer transacciones DNS sobre dispositivos en el LAN
Permite un rango de puertos TCP	sudo ufw allow 2500:5000/tcp	Esto permitirá a cualquiera en internet comunicarse a cualquier puerto en el rango 2500 - 5000 en dispositivos en el LAN
Elimina un número de regla	sudo ufw delete 1	Esto borrará la primera regla en la lista de reglas

Con estos comandos se puede crear una simple regla del cortafuegos. En la interfaz de usuario se le pide al usuario que introduzca una serie de datos y mediante un método de programación se capturan esos datos, para después enviárselos a ufw y que entonces se configure la regla.

Tabla 3: Comparativa entre herramientas de cortafuegos. Elaboración propia.

Herramientas de cortafuegos	**Interfaz gráfica**	**Público objetivo**	**Licencia**	**Orientada a computadora personal**	**Compatibilidad con el centro de control de GNOME**
Iptables	no	no	pública	si	si
Shorewall	no	no	pública	no	no
Gufw	si	si	pública	si	si
Ipcop	si	si	pública	no	no
Forefront TMG	si	si	privativa	no	no
Ufw	no	no	pública	si	si

1.9.1 Resultados obtenidos de las herramientas estudiadas

Luego del establecido análisis de las herramientas para la administración de cortafuegos buscando que la aplicación final sea adaptable al centro de control de GNOME 3.20.1, configurable para que se pueda seguir perfeccionando y tenga una interfaz de usuario basada en menú para que el usuario tenga una mejor experiencia; se concluye que:

- Es necesario una herramienta que sea compatible con GNOME 3.20.1 y su integración no cause problemas de bloqueo o reconfiguraciones indebidas al sistema. Por lo que una herramienta muy eficaz seria ufw ya que es un motor para la configuración de reglas de cortafuegos y muy utilizado por otras herramientas de cortafuegos. Ufw utiliza líneas de comandos para configurar las reglas de cualquier sistema GNU/Linux mediante iptables.
- Gufw es una herramienta muy intuitiva, por lo tanto, las ideas sobre la interfaz, sobre todo la presentación, botones, mensajes y su manera de interactuar con el usuario se pueden tomar para la realización de la interfaz de usuario. No se tomó en cuenta a la hora de integrar su interfaz al panel de control ya que esta aplicación está desarrollada en Python y el centro de control solo carga interfaces que estén desarrolladas en C.

1.10 Tecnologías y metodología

Las metodologías de desarrollo se pueden definir como guías para la construcción de un producto. Tienen como objetivo aumentar la calidad del producto y definen un conjunto de etapas por las cuales debe transitar el proceso de desarrollo. Estas definen el qué, cómo y cuándo debe desarrollarse cada una de las actividades (Ramos, y otros).

1.10.1 Metodología a utilizar

Todo proceso de desarrollo de software debe estar guiado por una metodología de desarrollo de software.
Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.), se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI.
La metodología AUP-UCI surge con el objetivo de aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas. Estas prácticas se centran en el desarrollo de productos y servicios de calidad (UCI, 2015).
Para el desarrollo de la solución se utilizará la variante de la metodología AUP propuesta por la UCI, ya que es una de las estrategias de los centros productivos de la universidad para ganar en uniformidad en el desarrollo de todas las soluciones. Además, las características de la misma permiten ajustarse a las necesidades de la presente investigación.
AUP-UCI establece tres fases que transcurren de manera consecutiva.

1. Inicio: El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.

2. Ejecución: El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.

3. Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

1.10.2 Lenguajes de programación y herramientas

En el proceso de desarrollo de funcionalidades en el módulo del cortafuegos del centro de control de GNOME 3.20.1, se utiliza el lenguaje de programación C, por ser este su lenguaje natural de implementación. Como navegador de ayuda para la documentación de interfaz de programación de aplicaciones (API) se utiliza a Devhelp y como biblioteca grafica GTK. Se desarrolla haciendo uso del IDE de programación Builder, la herramienta Glade para el diseño de interfaz y Pencil como herramienta para crear prototipos del módulo a desarrollar. A continuación, se describen las herramientas mencionadas:

Builder

Builder en su versión 3.20.4, la cual incluye funcionalidades y características adaptables a un proyecto desarrollado con el entorno de trabajo GTK. Builder tiene soporte para la integración con *GNU/Make* un conjunto de archivos de configuración que aseguran que el proyecto pueda ser distribuido e instalado en la mayoría de sistemas tipo Unix especificando las dependencias a bibliotecas compartidas, recursos y un compilador de código para compilar la aplicación (Ubunto, 2017).

C

En su versión 2.23 se trata de un lenguaje de programación de tipos de datos estáticos, débilmente tipificado, de medio nivel, pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel, pero a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos (Santos Espino, 2009).

Glade

Glade es una herramienta que permite rápidamente y eficientemente el diseño de interfaces gráficas de usuario. Las interfaces de usuario se guardan como un archivo XML que describe la estructura del *widget*[13] y sus propiedades. Estos son asociados con cada uno de los manejadores de señales. El paquete LibGlade

[13] *Es una pequeña aplicación o programa que su principal objetivo está en dar acceso a funciones que son frecuentemente usadas y proveer de información visual.*

puede cargar el archivo de interfaz de usuario con el fin de construir dinámicamente las aplicaciones, lo que permite modificar la interfaz de usuario estéticamente sin necesidad de recompilar la aplicación. Glade se utiliza para diseñar la interfaz de usuario de una aplicación, configurar las señales que serán asociados con funciones de retro llamada implementadas en el código y cuidar las propiedades comunes del *widget*. Sin embargo, Glade no es un editor de código o un entorno de desarrollo integrado, imprime los archivos que deben ser cargados por su aplicación, y debe implementar todas las funciones de devolución de llamada en el código. Glade sólo está destinado a simplificar el proceso de inicialización de interfaz gráfica de usuario de la aplicación y la conexión de las señales (Krause, 2007).

Pencil

Pencil en su versión 3.0.3 está diseñado como una herramienta de creación de prototipos de interfaz de usuario, libre y de código abierto que la gente puede instalar y usar fácilmente para crear simulaciones en plataformas de escritorio populares.

Devhelp

En su versión 3.2 es un navegador GTK+/GNOME para la documentación de la API; funciona de forma nativa con gtk-doc (que es el formato de referencia de la API para la documentación GTK+/GNOME). Se integra con herramientas de desarrollo de GNOME como Glade, y es una aplicación oficial del proyecto GNOME. Devhelp utiliza GTK y WedKit para la representación html de la documentación (Gnome, 2016).

Herramienta de modelado

Las herramientas *CASE* (*computer aided software engineering*, ingeniería de software asistida por ordenador) son las aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y dinero. Estas herramientas contribuyen de manera directa en todos los aspectos del ciclo de vida de desarrollo del software en tareas como la realización de un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (Larman, 2006).

Visual paradigm (v8.0)

En su versión 8.0 fue la herramienta CASE seleccionada, es una herramienta para el modelado UML profesional que soporta el ciclo de vida completo de desarrollo del software: análisis y diseño, construcción, pruebas y despliegue (E. Hernández Orallo, 2009).

1.10.3 Bibliotecas utilizadas

Las principales bibliotecas de las distribuciones de GNU/Linux están escritas en el lenguaje de programación C, puesto que es uno de los lenguajes ideales para el desarrollo de bibliotecas gráficas, de multimedia y para el trabajo con sistemas de bajo nivel, no así para sistemas complejos ya que se hace difícil y propenso a

errores en el desarrollo. Las bibliotecas son de mucha ayuda en el manejo de widget, interfaces de usuarios, llamadas de funcionalidades en widget, el tratamiento de clases y widget como objetos, manejo directamente de memoria y brindan funcionalidades ya implementadas. A continuación, se describen las necesarias para la solución:

GTK

En su versión 3.20 es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI) (Dennis Ritchie, 1998).

GtkWidget

GtkWidget es la clase base de todos los widget en GTK, más sus derivados. Gestiona el ciclo de vida del widget, su estado y estilo (Gnome, 2016). Se hace uso de esta biblioteca para la creación de widget como los botones y ventanas.

Gsettings

Es el API de alto nivel para configuración de aplicaciones. La clase Gsettings proporciona una API conveniente para almacenar y recuperar ajustes de la aplicación. La lectura y escritura pueden ser consideradas no bloqueante. La lectura con Gsetting es extremadamente rápida, aproximadamente el mismo orden de magnitud (pero más lento) que una búsqueda (Gnome, 2016).

Gtkbuilder

Es el constructor de interfaz de usuario (UI). GtkBuilder es un objeto auxiliar que lee descripciones textuales de una UI y la instancia de los objetos descritos. GtkBuilder analiza descripciones textuales de UI que se especifican en un formato XML (Gnome, 2016).

Valoraciones finales

El estudio de los diferentes centros de control asociados a los entornos de escritorio para software libre y los que proponen los sistemas operativos privativos, permitió determinar que:

- Existen pocos centros de control en los sistemas operativos libres que tengan integrada la configuración de las reglas del cortafuegos y por tanto significa una desventaja en cuanto a los sistemas privativos que ofrecen esta posibilidad de manera fácil e intuitiva.
- Dentro del módulo se integrará ufw para la administración de las reglas de un cortafuegos.
- La selección de las herramientas y tecnologías necesarias para la implementación de un módulo para la configuración de las reglas de un cortafuegos contribuye a la calidad de la solución.

Capítulo 2. Análisis y diseño del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.2.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0

En el presente capítulo se establecen las principales características con que contará la solución ya sean como requisitos funcionales y no funcionales. Se generan los diagramas correspondientes a las etapas de planificación y diseño que propone la metodología de desarrollo AUP-UCI. Además, se expone el diagrama de componentes y la arquitectura que se utiliza.

2.1 Modelo de dominio

El modelo de domino es una representación gráfica de objetos reales relacionados con el problema y las relaciones entre ellos. Se comprende la interacción del sistema con los usuarios, facilita la captura de requisitos, pues ya se tiene una noción de lo que se va a realizar.

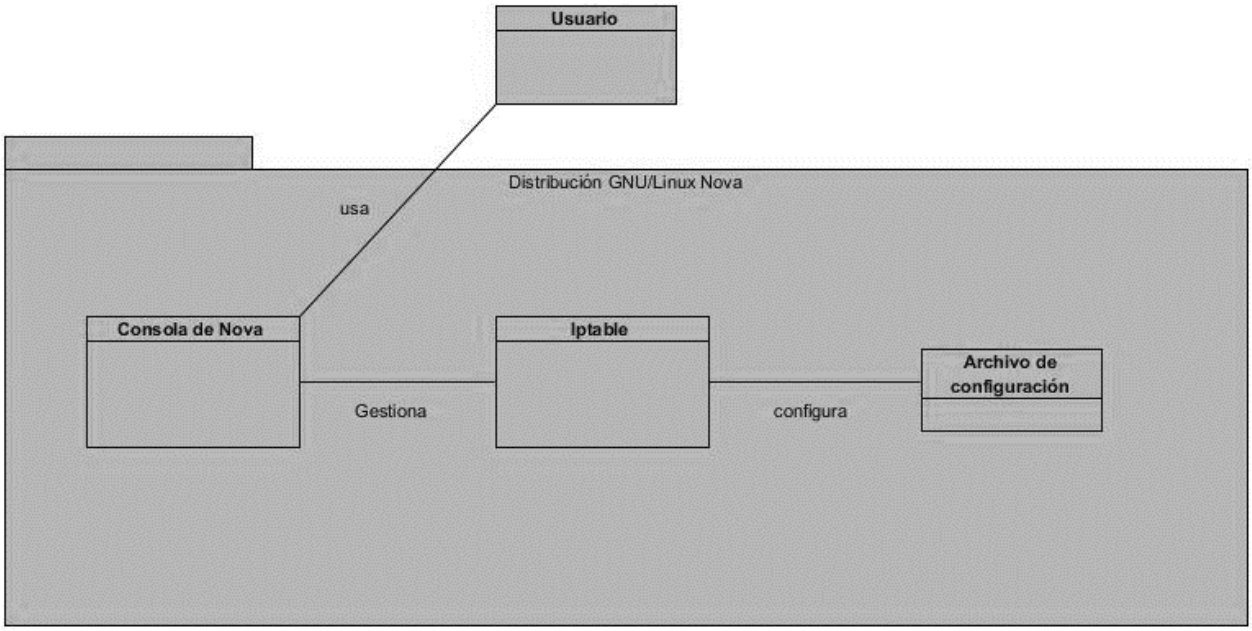


Ilustración 11: Modelo del dominio. Elaboración propia.

A continuación, se describirá cada uno de los elementos representados en la Ilustración 11.

Usuario: Es la persona que usa el cortafuegos.

Distribución GNU/Linux Nova: Es el sistema donde está ejecutándose el cortafuegos.

Consola de Nova: Es por donde se ejecutan las reglas del cortafuegos.

Iptables: Es el cortafuegos.

Archivo de configuración: Es donde se guardan las reglas configuradas.

2.2 Propuesta de solución

El módulo permitirá a los usuarios de la distribución cubana GNU/Linux Nova Escritorio 6.0 configurar las

reglas de un cortafuegos desde el centro de control de GNOME 3.20.1 mediante una interfaz de usuario. Los usuarios podrán adicionar, eliminar y modificar los perfiles y reglas creados. Este módulo eliminará la dependencia de software de terceros para la configuración de las reglas de un cortafuegos. En la Ilustración 12, se muestra cómo quedaría la estructura de la solución propuesta para el desarrollo del módulo.

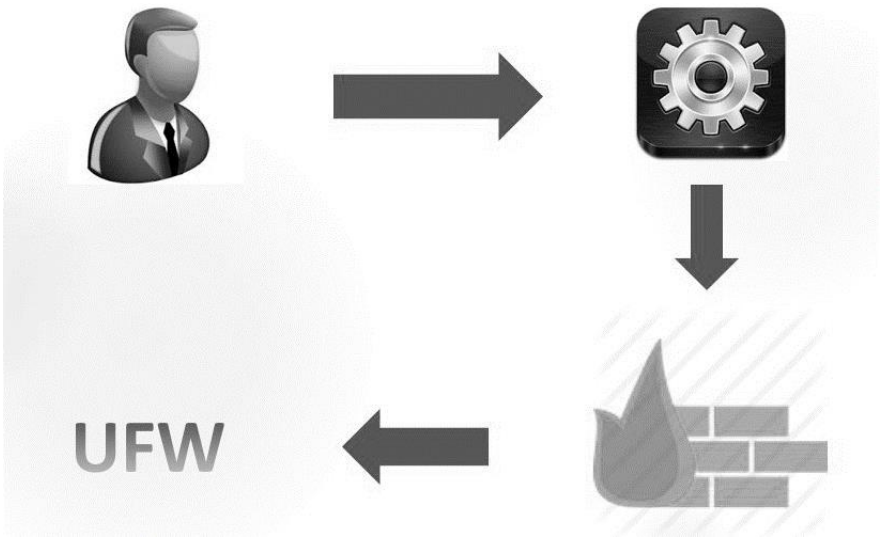


Ilustración 12: Propuesta de solución para el módulo de un cortafuegos. Elaboración propia.

2.3 Requisitos funcionales y no funcionales

Un requisito funcional describe 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 (Sommerville, 2006).

Un requisito no funcional: 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 (Sommerville, 2006).

Listado de requisitos funcionales (RF)

Tabla 4: Requisitos. Elaboración propia.

Requisitos funcionales (RF)		
Asignado a	Ítem	Descripción
Prioridad Alta		
John Claro Andrade	1	Crear perfil.
John Claro Andrade	2	Eliminar perfil.
John Claro Andrade	3	Modificar perfil.
John Claro Andrade	4	Mostrar perfil.
John Claro Andrade	5	Crear regla.
John Claro Andrade	6	Eliminar regla.
John Claro Andrade	7	Modificar regla.
John Claro Andrade	8	Mostrar regla.
Prioridad Media		
John Claro Andrade	9	Iniciar el cortafuegos.
John Claro Andrade	10	Detener el cortafuegos.
Prioridad Baja		
John Claro Andrade	11	Salvar configuraciones existentes.
John Claro Andrade	12	Cargar configuraciones existentes.
Requisitos no funcionales		
Software		
1	Es necesario tener instalada la distribución cubana GNU/Linux Nova Escritorio 6.0 para que el cortafuegos funcione correctamente sobre este sistema operativo.	
Restricciones del diseño		
2	Utilizar como lenguaje de programación C.	
Usabilidad		
4	El módulo debe poseer una interfaz intuitiva, que posibilite a los usuarios sin experiencia una rápida adaptación.	
5	El módulo debe ser extensible, por lo que con el tiempo se le pueden adicionar nuevas funcionalidades.	
Interfaz		
6	Interfaz ajustada a los estándares de diseño del panel de control del proyecto GNOME.	

2.4 Historias de usuario (HU)

La HU es la forma que tiene la metodología de desarrollo AUP-UCI en el escenario cuatro para especificar

los requisitos del software. Estas son escritas en el lenguaje del cliente, en las cuales se describe brevemente las características que el sistema tiene que tener. Las HU deben ser lo suficientemente comprensibles y delimitadas como para que el programador las implemente en unas pocas semanas. A continuación, se definen las diferentes historias de usuarios correspondientes a los requisitos definidos.

Tabla 5: HU crear perfil. Elaboración propia.

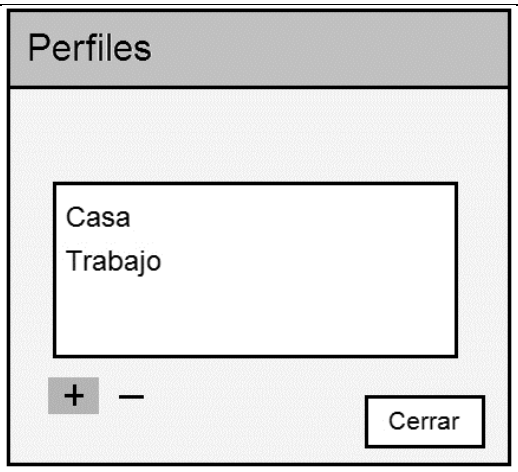
Historias de usuarios	
Número: HU-01	**Nombre HU:** Crear perfil.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Este escenario está relacionado con la gestión de los perfiles existentes. En este caso se refiere al proceso de crear un perfil.	
Observación: Para crear un perfil el usuario selecciona el botón preferencia y a continuación selecciona la opción crear (+). Después se creará un fichero en la dirección */etc/gnome-firewall/* donde se guardarán todos los datos de este perfil: nombre, tipo de regla y cantidad de reglas. Para poder cargar después el perfil con todos sus datos.	
Prototipo de interfaz: 	

Tabla 6: HU Eliminar perfil. Elaboración propia.

Historias de usuarios	
Número: HU-02	**Nombre HU:** Eliminar perfil.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1

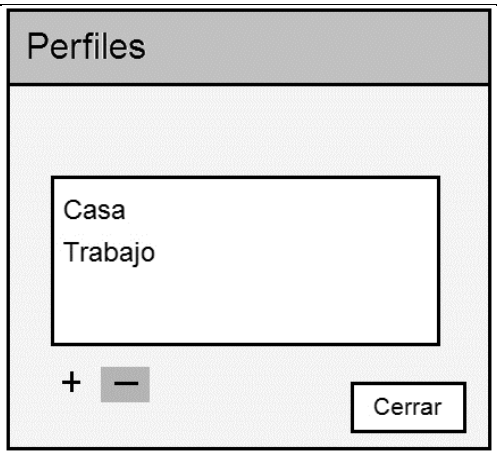
Prioridad en negocio: Media	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Este escenario está relacionado con la gestión de los perfiles existentes. En este caso se refiere al proceso de eliminar un perfil.	
Observación: Para eliminar un perfil el usuario selecciona el botón de preferencia, después selecciona el perfil a eliminar y posteriormente se selecciona la opción eliminar (-). Después se eliminará el fichero del perfil seleccionado en la dirección */etc/gnome-firewall/.*	
Prototipo de interfaz: 	

Tabla 7: HU Editar perfil. Elaboración propia.

Historias de usuarios	
Número: HU-03	**Nombre HU:** Editar perfil.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Este escenario está relacionado con la gestión de los perfiles existentes. En este caso se refiere al proceso de editar un perfil.	
Observación: Para editar un perfil el usuario, primero debe buscar el perfil a editar, dar doble clic y después editarlo. Luego se modificarán los datos en el fichero del perfil seleccionado.	

Prototipo de interfaz:

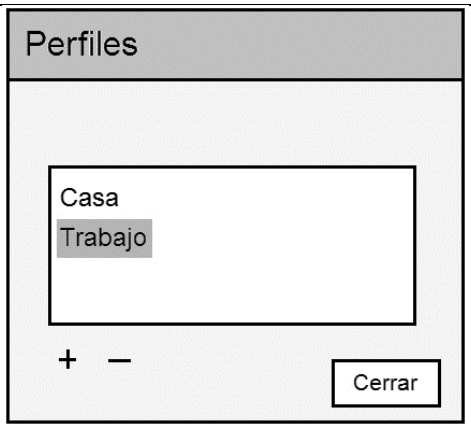


Tabla 8: HU Mostrar perfil. Elaboración propia.

Historias de usuarios	
Número: HU-04	**Nombre HU:** Mostrar perfil.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Media	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Este escenario está relacionado con la gestión de los perfiles existentes. En este caso se refiere al proceso de mostrar un perfil.	
Observación: Se selecciona el botón preferencia, después aparecerá una vista con los perfiles creados.	

Prototipo de interfaz:

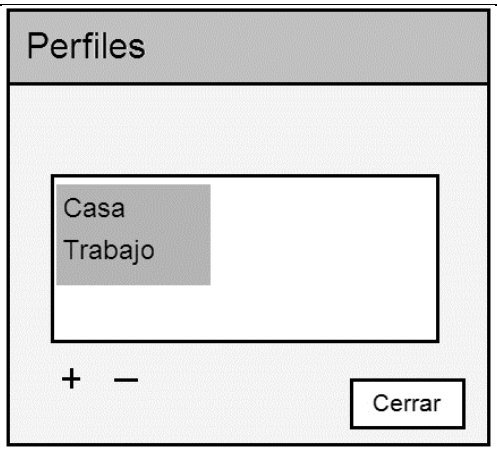


Tabla 9: HU crear regla. Elaboración propia.

Historias de usuarios	
Número: HU-05	**Nombre HU:** Crear regla.
Modificación de HU: Ninguna	

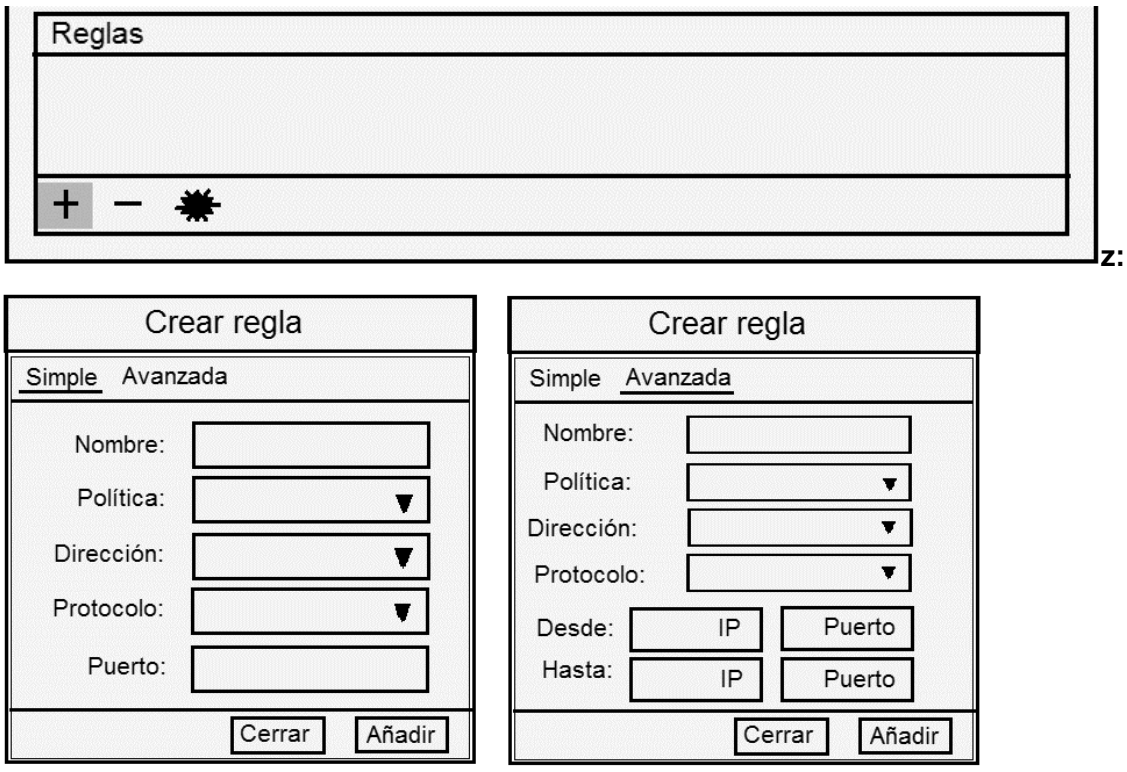
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Medio	**Puntos reales:** 1 semana
Descripción: Permitirá crear una regla.	
Observación: Para crear una regla el usuario selecciona la opción crear (+). Luego aparecerá una ventana donde podrá tener dos opciones: **Opción 1:** Podrá configurar una regla simple donde podrá poner el nombre, política, dirección, protocolo, puerto y añadir o cerrar la regla. **Opción 2:** Podrá configurar una regla avanzada donde podrá poner el nombre, política, dirección, protocolo, IP origen e IP destino y sus respectivos puertos, añadir o cerrar la regla. Luego de añadirse la regla, se ejecutará un método donde según los parámetros introducidos por el usuario, se ejecutará UFW permitiendo crear la regla en el sistema. Después se guardará la regla en el fichero del perfil activo.	
Prototipo de interfaz:  	

Tabla 10: HU eliminar regla. Elaboración propia.

Historias de usuarios	
Número: HU-06	**Nombre HU:** Eliminar regla.

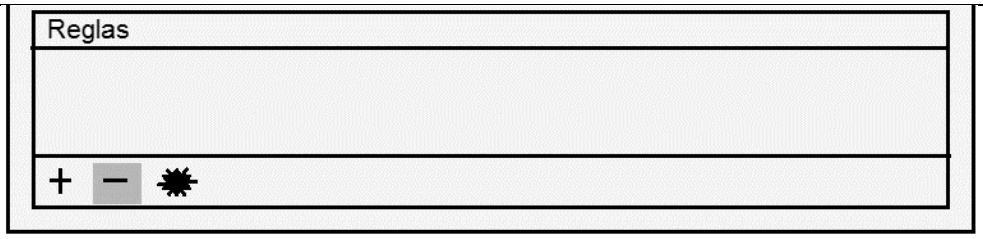
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Medio	**Puntos reales:** 1 semana
Descripción: Permitirá eliminar una regla.	
Observación: Para eliminar una regla el usuario selecciona la regla que quiera eliminar del menú de reglas y luego da clic a la opción eliminar. Luego se eliminará del fichero del perfil activo.	
Prototipo de interfaz: 	

Tabla 11: HU editar reglas. Elaboración propia.

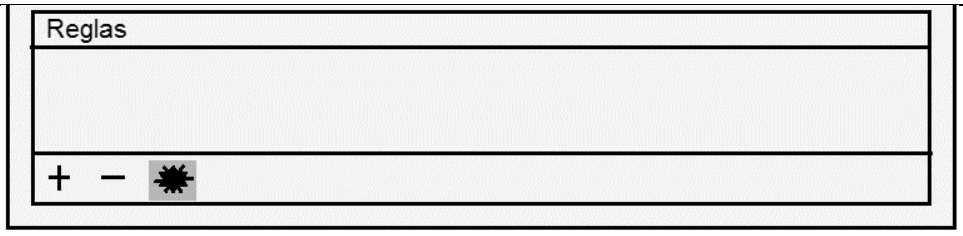
Historias de usuarios	
Número: HU-07	**Nombre HU:** Editar reglas.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Medio	**Puntos reales:** 1 semana
Descripción: Permitirá editar una regla.	
Observación: El usuario selecciona la regla que quiera editar del menú de reglas y luego da clic a la opción editar. Posteriormente saldrá una ventana donde podrá cambiar el nombre de la regla, política, dirección, protocolo, el puerto si es una regla simple y añadir o cancelar la edición. Si es una regla avanzada entonces podrá cambiar el nombre de la regla, política, dirección, protocolo, el IP origen e IP destino con sus respectivos puertos y añadir o cancelar la edición. Luego se editará del fichero del perfil activo.	
Prototipo de interfaz: 	

Tabla 12: HU Mostrar listas de reglas configuradas. Elaboración propia.

Historias de usuarios

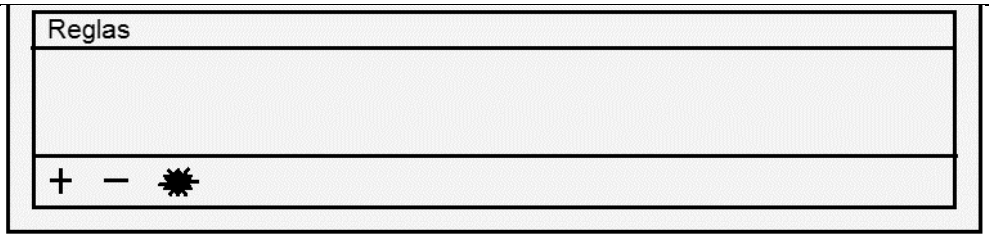
Número: HU-08	**Nombre HU:** Mostrar listas de reglas configuradas.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 1
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Medio	**Puntos reales:** 1 semana
Descripción: Permitirá mostrar las reglas de iptables que estén insertadas.	
Observación: La aplicación permitirá que las reglas configuradas se muestren.	
Prototipo de interfaz: 	

Tabla 13: HU Gestionar estado del cortafuegos. Elaboración propia.

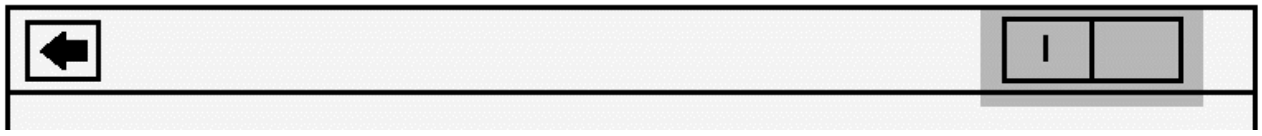
Historias de usuarios	
Número: HU-09	**Nombre HU:** Gestionar estado del cortafuegos.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 2
Prioridad en negocio: Media	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Permitirá activar o desactivar el cortafuegos.	
Observación: **Estado 1:** 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. **Estado 2:** 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.	
Prototipo de interfaz: 	

Tabla 14: HU Gestionar configuraciones existentes. Elaboración propia.

Historias de usuarios

Número: HU-010	**Nombre HU:** Gestionar configuraciones existentes.
Modificación de HU: Ninguna	
Usuario: John Claro Andrade	**Iteración asignada:** 4
Prioridad en negocio: Baja	**Puntos estimados:** 2 semana
Riesgo en desarrollo: Baja	**Puntos reales:** 2 semana
Descripción: Este escenario está relacionado con la gestión de las configuraciones existentes. En este caso se refiere al proceso de cargar y guardar las configuraciones realizadas.	
Observación: **Estado 1:** La aplicación en este caso podrá cargar las configuraciones de las reglas que se encuentran en el fichero donde se guarda el perfil activo. **Estado 2:** La aplicación podrá guardar las configuraciones de las reglas que se encuentren activas. El módulo creará un fichero donde se guardará el perfil activo.	

2.4.1 Tareas de ingeniería

A continuación se relacionan algunas de las tareas de ingenierías correspondientes a las historias de usuario, basándose en la prioridad que tienen. Debido a que el proceso es cambiante estas deben ir adecuándose a las nuevas funcionalidades propuestas, por lo que esta es solo una planificación inicial.

Tabla 15: Tarea de ingeniería programación de la funcionalidad, crear perfil. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 1	Nombre de la historia de usuario: HU-01
Nombre tarea: Crear perfil	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en añadir un perfil en los siguientes pasos: 1. Crear un fichero con el nombre del perfil. 2. Copiar la información dentro del fichero relacionada al perfil.	

Tabla 16: Tarea de ingeniería programación de la funcionalidad, eliminar perfil. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 2	Nombre de la historia de usuario: HU-02
Nombre tarea: Eliminar perfil	
Tipo de tarea: Desarrollo.	Puntos estimados: 5

Programador responsable: John Claro Andrade
Descripción: Esta funcionalidad consiste en eliminar un perfil en los siguientes pasos: 1. Eliminar el fichero de la ruta donde se guardan.

Tabla 17: Tarea de ingeniería programación de la funcionalidad, editar perfil. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 3	Nombre de la historia de usuario: HU-03
Nombre tarea: Editar perfil	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en editar un perfil en los siguientes pasos: 1. Editar la información del fichero a editar.	

Tabla 18: Tarea de ingeniería programación de la funcionalidad, mostrar perfil. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 4	Nombre de la historia de usuario: HU-04
Nombre tarea: Mostrar perfil	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en mostrar perfil en los siguientes pasos: 1. Mostrar todos los perfiles que se encuentran creados en la dirección donde se guardan.	

Tabla 19: Tarea de ingeniería programación de la funcionalidad, crear regla. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 5	Nombre de la historia de usuario: HU-05
Nombre tarea: crear regla.	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en crear regla en los siguientes pasos: 1. Copiar las reglas creadas en el fichero del perfil activo.	

Tabla 20: Tarea de ingeniería programación de la funcionalidad, eliminar regla. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 6	Nombre de la historia de usuario: HU-06
Nombre tarea: Eliminar regla.	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en eliminar regla en los siguientes pasos: 1. Eliminar las reglas copiadas en el fichero del perfil activo.	

Tabla 21: Tarea de ingeniería programación de la funcionalidad, editar regla. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 7	Nombre de la historia de usuario: HU-07
Nombre tarea: Editar regla.	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en editar regla en los siguientes pasos: 1. Editar las reglas copiadas en el fichero del perfil activo.	

Tabla 22: Tarea de ingeniería programación de la funcionalidad, mostrar regla. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 8	Nombre de la historia de usuario: HU-08
Nombre tarea: Mostrar regla.	
Tipo de tarea: Desarrollo.	Puntos estimados: 5
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en mostrar regla en los siguientes pasos: 1. Mostar las reglas copiadas en el fichero del perfil activo.	

Tabla 23: Tarea de ingeniería programación de la funcionalidad, gestionar estado del cortafuegos. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 9	Nombre de la historia de usuario: HU-09
Nombre tarea: Gestionar estado del cortafuegos.	

Tipo de tarea: Desarrollo.	Puntos estimados: 1
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en activar o desactivar el cortafuegos.	

Tabla 24: Tarea de ingeniería programación de la funcionalidad, gestionar configuraciones existentes. Elaboración propia.

Tarea de ingeniería	
Número de la tarea: 9	Nombre de la historia de usuario: HU-09
Nombre tarea: Gestionar configuraciones existentes.	
Tipo de tarea: Desarrollo.	Puntos estimados: 1
Programador responsable: John Claro Andrade	
Descripción: Esta funcionalidad consiste en cargar o guardar las configuraciones del cortafuegos.	

2.4.2 Plan de entrega

A continuación, se presentará una tabla donde se encuentra la especificación de las entregas al cliente, la misma es el resultado del artefacto Plan de iteraciones. Dicho plan contiene las HU por iteraciones, especificando cuáles de ellas serán implementadas en cada iteración del proceso. El desarrollo se realiza en tres iteraciones las cuales se muestran a continuación:

Iteración 1:

El principal objetivo de esta iteración es dar cumplimiento a las historias de usuario donde la prioridad sea alta (01, 02, 03, 04, 05, 06, 07, 08) ya que estas son fundamentales para el correcto funcionamiento del cortafuegos.

Iteración 2:

Esta segunda iteración tiene como fin dar cumplimiento a las historias de usuario donde la prioridad sea media (09), estas brindan al usuario el modo de activar o desactivar el cortafuegos.

Iteración 3:

Esta tercera y última iteración tiene como meta dar cumplimiento a las historias de usuario donde la prioridad es baja (10). El desarrollo de estas HU finalizará las funcionalidades expuestas anteriormente, lo que posibilitará que el módulo podrá estar listo para hacer uso de ella.

Tabla 25: Plan de entrega. Elaboración propia.

Iteración	Descripción	Orden	Duración total de la iteración
1	Desarrollo de las	HU-01, HU-02, HU-03,	5 semanas

	Historias de Usuario de prioridad alta.	HU-04, HU-05, HU-06, HU-07, HU-08	
2	Desarrollo de las Historias de Usuario de prioridad media.	HU-09	1 semanas
3	Desarrollo de las Historias de Usuario de prioridad baja.	HU-10	1 semanas

2.5 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes o bibliotecas cargadas dinámicamente. Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente (Sommerville, 2006).

Este diagrama es un esbozo inicial del sistema sin entrar en especificaciones ni detalles y proporciona ventajas como: confeccionar un diseño inicial y sencillo del sistema que sirve como base para la definición de una futura arquitectura. En el siguiente diagrama se muestra la relación que posee el componente cc-firewall-panel.c con el resto de los componentes, para mostrar la interfaz a los usuarios y llevar el proceso de configurar las reglas de un cortafuegos. Para integrar el módulo al centro de control se utiliza el componente cc-panel-loader.c y para conectar la interfaz gráfica con el motor de configurar las reglas (UFW) se utiliza ufwadapter.c.

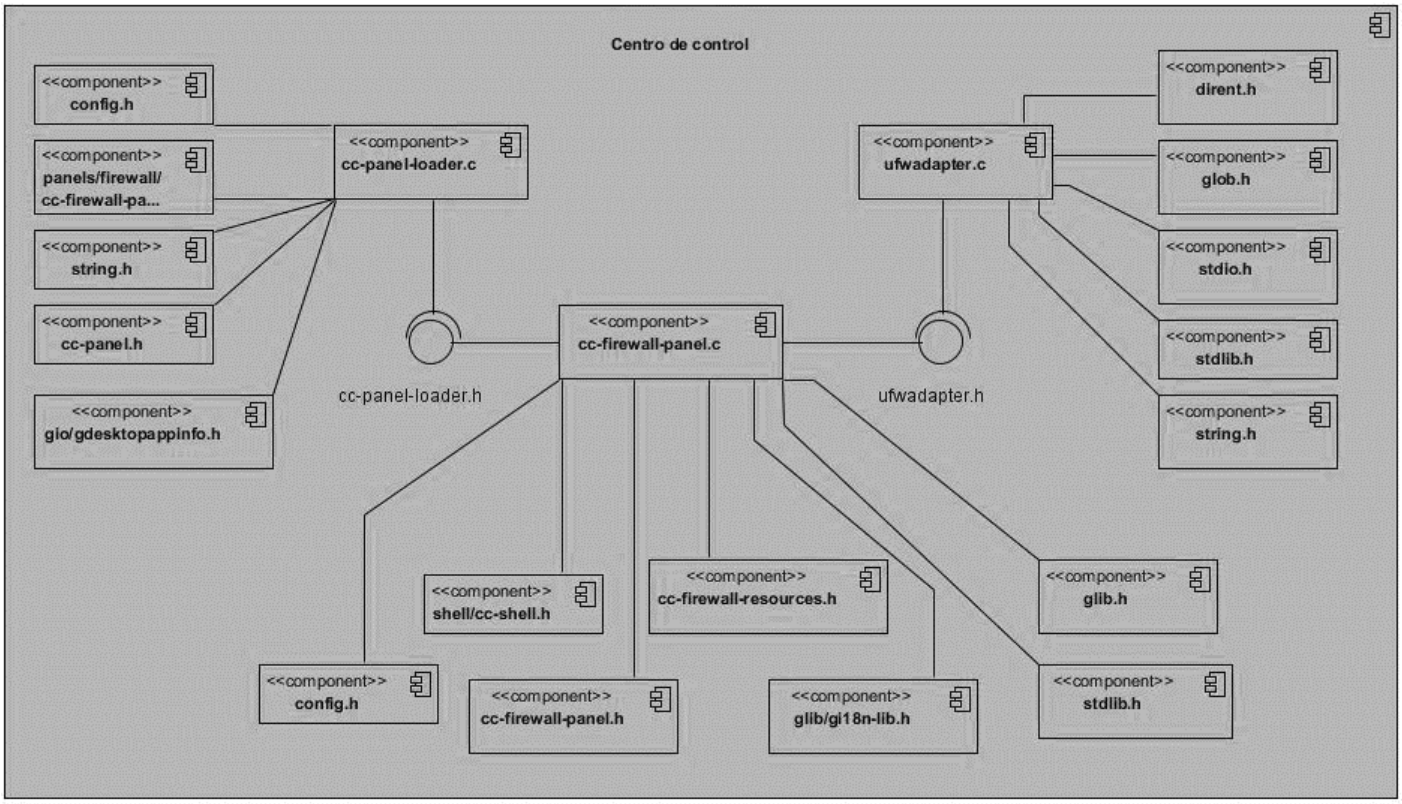


Ilustración 13: Diagrama de componente. Elaboración propia.

2.6 Diseño arquitectónico

Mediante el estudio del centro de control de GNOME 3.20.1 y a través del análisis, se muestra gráficamente un esbozo de la arquitectura del centro de control, ver la Ilustración 10.

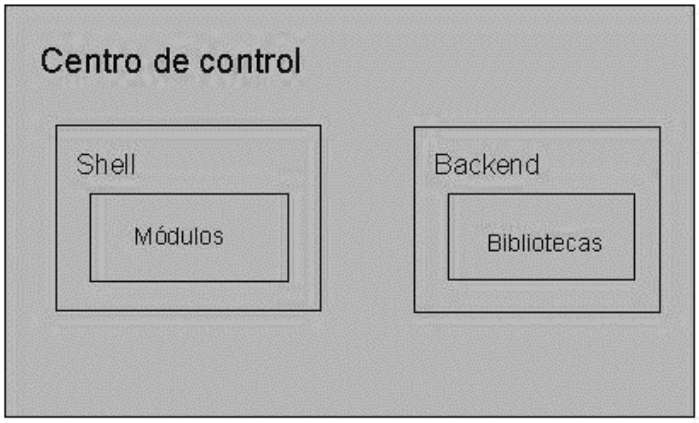


Ilustración 14: Arquitectura del centro de control de GNOME 3.20.1. Elaboración propia.

Donde shell es la capa que permite mostrar los módulos que contiene el panel, posibilitando el cumplimiento de sus funciones haciendo uso de las bibliotecas incluidas en el backend.

El desarrollo dentro de los módulos del centro de control de GNOME 3.20.1, deberá regirse por la arquitectura definida en el mismo, aun así, se hace evidente el uso de un patrón arquitectónico en el centro de control. A

través del estudio de diferentes estilos arquitectónicos, se llega a la conclusión que dentro del centro de control de GNOME 3.20.1 se evidencia el uso de la arquitectura por **descomposición modular**, donde el software se estructura en grupos funcionales muy acoplados. No existe una distinción rígida entre la organización del sistema y la descomposición modular, los componentes de los módulos son normalmente más pequeños que los subsistemas, lo cual permite usar estilos alternativos de descomposición. El diseño modular propone dividir el sistema en partes diferenciadas y definir sus interfaces. Una descomposición modular posee cualidades como independencia funcional, acoplamiento, cohesión y adaptabilidad (Sommerville, 2005).

2.7 Estándar de codificación en C

Un estándar de codificación en informática son reglas que se rigen la escritura del código fuente; con el fin de homogeneizar la manera de codificar para que otros puedan entenderlo sin muchas dificultades. Parte esencial en cada estándar lo establecen las formas de escribir clases, métodos y variables. La propuesta de solución a ser implementada debe regirse por las normas de codificación del proyecto GNOME, ya que será implementando un módulo para la configuración de las reglas de un cortafuegos. Estas indicaciones de codificación son inspiradas por los documentos *GTK's CODING-STYLE*, *The Linux Kernel's CodingStyle*, y el GNU *Coding Standards*. Cada uno de ellos es una variación de otros documentos con modificaciones particulares para cada proyecto y cultura. Una codificación basada en estándares permite el soporte a largo plazo del proyecto (Santos Espino, 2009).

La regla más importante: Cuando se escribe el código se debe preservar el estilo del código original.

Usar líneas de código con 80 y 120 caracteres de longitud para permitir su visualización en la mayoría de los monitores con una fuente de texto legible.

Sangría del código: Estilo GNU. Cada nuevo nivel se le añade de sangría 2 espacios y las líneas que están en el mismo nivel comparten la misma sangría.

Funciones: El tipo de dato de retorno de la función debe ser puesto en una línea antes.

Parámetros en funciones: Se deben escribir separados por coma cada uno en una nueva línea.

Condicionales: No se debe comparar las variable booleanas con valores true o false usar siempre if (variable_condicional). En caso de varias variables en la condición poner cada una seguida de la operación lógica en líneas nuevas.

Paréntesis: Siempre que se escriba una llave adicionar un espacio antes nunca después.

2.8 Pasos para integrar el módulo del cortafuegos al centro de control de GNOME 3.20.1

Dentro de una carpeta llamada *panels* estarán todos los módulos del centro de control. Ahí se crea una carpeta con el nombre de firewall que tendrá los archivos para la configuración del módulo del cortafuegos. Estos archivos son:

1. *cc-firewall-panel.c y cc-firewall-panel.h* para el trabajo con la interfaz del cortafuegos.

2. *ufwadapter.c y ufwadapter.h* para el enlace de la interfaz con ufw.
3. *firewall.gresource.xml* para referenciar las interfaces al módulo.

Después se modificó el archivo *cc-panel-loader.c* que se encuentra dentro de la carpeta *shell*. En este archivo se le agregó en la estructura siguiente una línea de código para que cargara el módulo del cortafuegos en el centro de control:

```
/* Extension points */
extern GType cc_background_panel_get_type (void);
#ifdef BUILD_BLUETOOTH
extern GType cc_bluetooth_panel_get_type (void);
#endif /* BUILD_BLUETOOTH */
extern GType cc_color_panel_get_type (void);
extern GType cc_date_time_panel_get_type (void);
extern GType cc_display_panel_get_type (void);
extern GType cc_firewall_panel_get_type (void);
extern GType cc_info_panel_get_type (void);
extern GType cc_keyboard_panel_get_type (void);
extern GType cc_mouse_panel_get_type (void);
#ifdef BUILD_NETWORK
extern GType cc_network_panel_get_type (void);
#endif /* BUILD_NETWORK */
extern GType cc_notifications_panel_get_type (void);
extern GType cc_goa_panel_get_type (void);
extern GType cc_power_panel_get_type (void);
#ifdef BUILD_PRINTERS
extern GType cc_printers_panel_get_type (void);
#endif /* BUILD_PRINTERS */
extern GType cc_privacy_panel_get_type (void);
extern GType cc_region_panel_get_type (void);
extern GType cc_search_panel_get_type (void);
extern GType cc_sharing_panel_get_type (void);
extern GType cc_sound_panel_get_type (void);
```

extern GType cc_ua_panel_get_type (void);

extern GType cc_user_panel_get_type (void);

También se le agregó en la estructura siguiente:

all_panels[] = {

PANEL_TYPE("background", cc_background_panel_get_type),

#ifdef BUILD_BLUETOOTH

PANEL_TYPE("bluetooth", cc_bluetooth_panel_get_type),

#endif

PANEL_TYPE("color", cc_color_panel_get_type),

PANEL_TYPE("datetime", cc_date_time_panel_get_type),

PANEL_TYPE("display", cc_display_panel_get_type),

PANEL_TYPE("firewall", cc_firewall_panel_get_type),

PANEL_TYPE("info", cc_info_panel_get_type),

PANEL_TYPE("keyboard", cc_keyboard_panel_get_type),

PANEL_TYPE("mouse", cc_mouse_panel_get_type),

#ifdef BUILD_NETWORK

PANEL_TYPE("network", cc_network_panel_get_type),

#endif

Además se tuvo que modificar el archivo ***Makefile.in*** en la estructura siguiente:

*gnome_control_center_DEPENDENCIES = libshell.la $(am__DEPENDENCIES_1) *

*$(am__DEPENDENCIES_1) $(top_builddir)/libgd/libgd.la *
*$(top_builddir)/panels/common/liblanguage.la *
*$(top_builddir)/panels/common/libdevice.la *
*$(top_builddir)/panels/background/libbackground.la *
*$(top_builddir)/panels/color/libcolor.la *
*$(top_builddir)/panels/datetime/libdate_time.la *
*$(top_builddir)/panels/display/libdisplay.la *
*$(top_builddir)/panels/firewall/libfirewall.la *
*$(top_builddir)/panels/info/libinfo.la *
*$(top_builddir)/panels/keyboard/libkeyboard.la *
*$(top_builddir)/panels/mouse/libmouse-properties.la *

$(top_builddir)/panels/notifications/libnotifications.la \

$(top_builddir)/panels/online-accounts/libonline-accounts.la

Valoraciones finales

El análisis del módulo permitió identificar las principales funcionalidades que se le adicionarán. Teniendo en cuenta las especificaciones de la metodología de desarrollo seleccionada, se detallaron los artefactos generados en el desarrollo como:

- Los requisitos funcionales y las HU, las cuales posibilitaron la descripción de los requisitos de manera clara y legible.
- La planificación de iteraciones se hizo teniendo en cuenta la prioridad para el negocio de cada HU.
- Finalmente se realizaron los diagramas de componentes y el modelo de dominio lo cual permitió dar una mejor visión del proyecto.

Capítulo 3. Validación del módulo para la configuración de las reglas del cortafuegos en el centro de control de GNOME 3.20.1 para la distribución cubana GNU/Linux Nova Escritorio 6.0.

En el presente capítulo se describirán los casos de pruebas de aceptación recogidos como parte de la metodología de desarrollo de software seleccionada, con el objetivo de comprobar que la aplicación cumple exitosamente con todas las funcionalidades expuestas en el capítulo anterior, permitiendo así que el usuario determine su grado de aceptación con el módulo desarrollado.

3.1 Pruebas de software

Las pruebas de software son un conjunto de herramientas, técnicas y métodos que evalúan la excelencia y el desempeño de un software, involucra las operaciones del sistema bajo condiciones controladas y evaluando los resultados. Las técnicas para encontrar problemas en un programa son variadas y van desde el uso del ingenio por parte del personal de prueba hasta herramientas automatizadas que ayudan a aliviar el peso y el costo de tiempo de esta actividad (Pressman, 2002).

3.1.1 Niveles de prueba

Cuando se le van a aplicar pruebas a un software, se tienen en cuenta una serie de objetivos en diferentes escenarios y niveles de trabajo, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo (Pressman, 2002).

Pruebas de aceptación

Se escoge la prueba de aceptación ya que es necesario, para la validación de la solución, que el cliente esté de acuerdo con el funcionamiento del módulo desarrollado y así pueda emitir la carta de aceptación que demuestra la conformidad del cliente con la solución. Como técnica se escoge las pruebas alfa, estas se llevan a cabo por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario.

Prueba de usabilidad

Además de las pruebas de aceptación, se realizan las pruebas de usabilidad para determinar si el usuario podrá usar y entender la aplicación. Identifica las áreas de diseño que hacen al sistema de difícil uso para el usuario. La prueba de usabilidad detecta problemas relacionados con la conveniencia y practicidad del sistema desde el punto de vista del usuario. Se intenta que la aplicación sea fácil de entender, aprender y de usar, provea utilidad, funcionalidad y cumpla con las tareas para las cuales fue de desarrollada la aplicación (Rubin, 1994).

3.1.2 Método de prueba

Se escoge el método de caja negra ya que los niveles de pruebas escogidos pretenden probar el

funcionamiento de la interfaz y de las funcionalidades del sistema, para esto es necesario este método ya que permite comprobar el comportamiento del software a través de los requisitos funcionales (Pressman, 2002), obviando el funcionamiento interno y la estructura del programa.

Las pruebas de caja negra pretenden encontrar estos tipos de errores:

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores de comportamiento o desempeño.
- Errores de inicialización y de terminación.

3.2 Casos de pruebas de aceptación

El objetivo de los casos de prueba de aceptación en la metodología AUP-UCI es validar que el sistema cumple con el funcionamiento esperado. Estos son comprobados por el usuario de forma tal que sea este el que determine el grado de aceptación que tiene con respecto a las funcionalidades y el rendimiento del producto. Para la validación del módulo del centro de control de la distribución cubana de GNU/Linux Nova en la versión 6.0 se definieron los casos de prueba para las diferentes historias de usuario.

Tabla 26: Caso de prueba 01. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 01	**Nombre de la HU:** HU crear perfil.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad crear perfil, seleccionando el botón crear perfil(+).	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Selecciona el menú preferencias. 2. Seleccionar el menú crear perfil (+). 3. Escribir el nombre del perfil. 4. Se presiona el botón cerrar para salir de la ventana.	
Resultado esperado: Debe aparecer el perfil creado.	
Evaluación de la prueba: Satisfactoria.	

Tabla 27: Caso de prueba 02. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 02	**Nombre de la HU:** HU Eliminar perfil.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad eliminar perfil, seleccionando el botón eliminar perfil.	

Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.
Pasos de ejecución: 1. Selecciona el menú preferencias. 2. Debe estar seleccionado el perfil a eliminar. 3. Seleccionar la opción eliminar perfil (-). 4. Se presiona el botón cerrar para salir de la ventana.
Resultado esperado: Debe desaparecer el perfil.
Evaluación de la prueba: Satisfactoria.

Tabla 28: Caso de prueba 03. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 03	**Nombre de la HU:** HU Editar perfil.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad editar perfil, seleccionando el botón editar perfil.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Selecciona el menú preferencias. 2. Selecciona el perfil a editar. 3. Escribir el nombre nuevo. 4. Se presiona el botón cerrar para salir de la ventana.	
Resultado esperado: Debe aparecer el perfil ya editado.	
Evaluación de la prueba: Satisfactoria.	

Tabla 29: Caso de prueba 04. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 04	**Nombre de la HU:** HU Mostrar perfil.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad mostrar el perfil.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Selecciona el menú preferencias. 2. El perfil se muestra en un menú de perfiles.	
Resultado esperado: Debe aparecer los perfiles creados.	

Evaluación de la prueba: Satisfactoria.

Tabla 30: Caso de prueba 05. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 05	**Nombre de la HU:** HU crear regla.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad crear regla, seleccionando el botón crear regla (+).	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Seleccionar el menú crear regla (+). 2. Escribir los parámetros establecidos para crear una regla simple o avanzada. 3. Se presiona el botón crear.	
Resultado esperado: Debe aparecer la regla creada en el campo de mostrar las reglas.	
Observaciones: Al seleccionar el menú crear regla (+), saldrá una ventana para introducir datos en los campos asignados. En el campo ‘‘puerto’’ solo podrá introducir números de hasta cuatro dígitos. Si se le introducen mal los datos el sistema enviará un mensaje de error. Si es una regla avanzada tendrá que poner bien los IP y los puertos ya que si se introducen mal el sistema enviará un mensaje de error. Si no introduce ningún dato y presiona el menú añadir, el sistema no creará ninguna regla.	
Evaluación de la prueba: Satisfactoria.	

Tabla 31: Caso de prueba 06. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 06	**Nombre de la HU:** HU Eliminar regla.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad eliminar regla, seleccionando el botón eliminar regla.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Debe estar seleccionada la regla a eliminar. 2. Seleccionar el menú eliminar regla (-). 3. Se presiona el botón aceptar para eliminar la regla.	
Resultado esperado: Debe desaparecer la regla del campo de mostrar las reglas.	
Observaciones: Para poder eliminar una regla el usuario deberá primero tener una seleccionada, de lo	

contrario el sistema dará una alerta.
Evaluación de la prueba: Satisfactoria.

Tabla 32: Caso de prueba 07. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 07	**Nombre de la HU:** HU Editar regla.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad editar regla, seleccionando el botón editar regla.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. Seleccionar la regla a editar. 2. Seleccionar el menú editar regla. 3. Escribir los parámetros deseados. 4. Se presiona el botón aceptar para editar la regla.	
Resultado esperado: Debe aparecer la regla editada en el campo de mostrar las reglas.	
Observaciones: Para poder editar una regla el usuario deberá primero tener una seleccionada, de lo contrario el sistema dará una alerta.	
Evaluación de la prueba: Satisfactoria.	

Tabla 33: Caso de prueba 08. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 08	**Nombre de la HU:** HU Mostrar regla.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad mostrar regla.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel y tener una regla creada.	
Pasos de ejecución: 1. Las reglas se muestran en el módulo agregado.	
Resultado esperado: Debe aparecer las reglas creadas en el campo de mostrar las reglas.	
Evaluación de la prueba: Satisfactoria.	

Tabla 34: Caso de prueba 09. Elaboración propia.

Casos de pruebas de aceptación

Código caso de prueba: 09	**Nombre de la HU:** HU Gestionar estado del cortafuegos.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad estado del cortafuegos.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. El cortafuegos se activará o desactivará, dependiendo del estado en que se haya quedado.	
Resultado esperado: Debe activarse o desactivarse el cortafuegos.	
Evaluación de la prueba: Satisfactoria.	

Tabla 35: Caso de prueba 10. Elaboración propia.

Casos de pruebas de aceptación	
Código caso de prueba: 10	**Nombre de la HU:** HU Configuraciones existentes.
Nombre de la persona que realiza la prueba: John Claro Andrade	
Descripción de la prueba: Prueba a la funcionalidad configuraciones existentes.	
Condiciones de ejecución: Acceder al módulo del cortafuegos en el panel.	
Pasos de ejecución: 1. El cortafuegos guarda las configuraciones en un archivo del sistema, que podrá ser cargado cuando este se inicie. 2. Al reiniciar el sistema el cortafuegos cargará las configuraciones existentes del archivo que se haya guardado.	
Resultado esperado: El cortafuegos debe cargar y guardar las configuraciones existentes.	
Evaluación de la prueba: Satisfactoria.	

3.2.1 Resultados obtenidos en las pruebas de aceptación.

Las pruebas se centraron en el cumplimiento de las funcionalidades de todas las historias de usuario implementadas. Las no conformidades (NC) se clasificaron en alta (A), media (M) o baja (B) en dependencia del impacto que tuvieran, generalmente las altas responden a errores técnicos relacionados directamente con la funcionalidad interna de la HU y las bajas tienden a ser errores ortográficos, validaciones entre otros errores de bajo impacto.

Se llevaron a cabo 3 iteraciones donde se obtuvieron varias NC, en la primera iteración se detectaron 10 NC de las cuales 9 fueron de prioridad alta y 1 de prioridad media, las cuales fueron eliminadas; para la segunda iteración se detectaron 6 NC distribuidas en 4 de prioridad alta y 2 de prioridad baja; en la iteración final ya no se encontraron NC, por lo tanto las pruebas de funcionalidad en el sistema culminaron satisfactoriamente.

Las mismas permitieron garantizar que el módulo cumple con las funcionalidades definidas. Para mayor entendimiento de este proceso a continuación se muestra un gráfico de barras que ilustra las NC por cada iteración, ver ilustración 15.

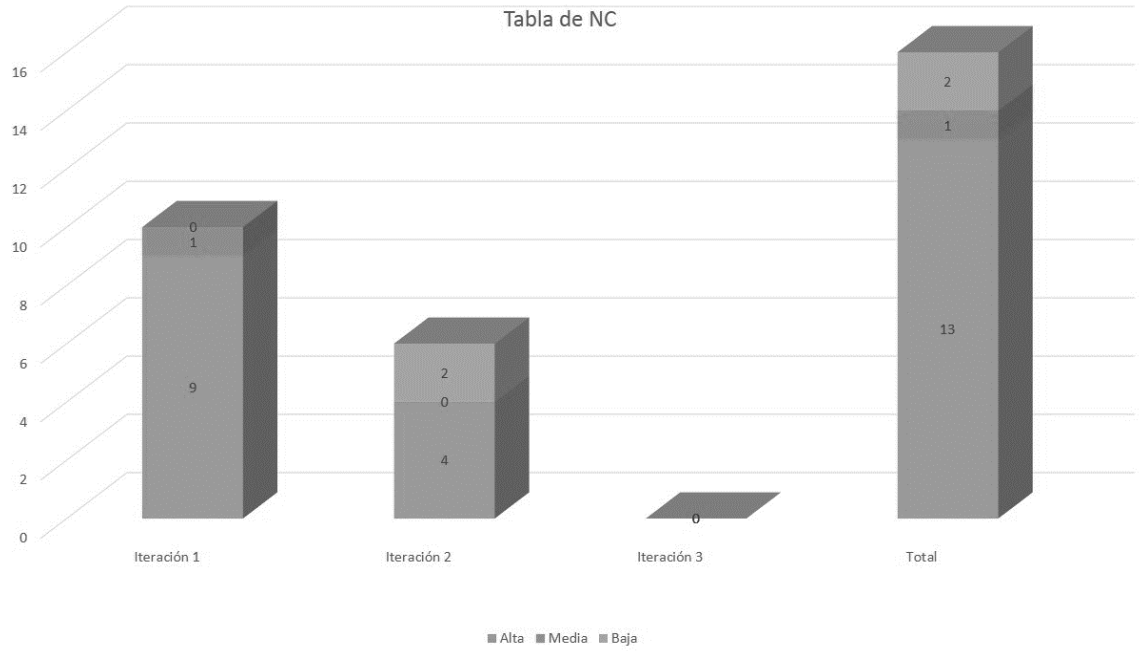


Ilustración 15: Resultado de las no conformidades por cada una de las iteraciones. Elaboración propia.

3.3 Pruebas de usabilidad.

Se evaluó la usabilidad del módulo a partir de lo establecido en el modelo de calidad para el Aseguramiento de Calidad en el Desarrollo de Software Libre CENDITEL, 2013. Para las pruebas de usabilidad se contó con el apoyo de un profesional con 2 años de experiencia, en las pruebas de usabilidad en el campo de software libre, que evaluó la interfaz gráfica a partir de un conjunto de preguntas definidas para este fin (Ver anexo 2). A continuación, se presentan los aspectos positivos y negativos encontrados en la primera iteración de las pruebas.

Aspectos positivos del módulo:

1. La interfaz escribe mensajes de ayuda al usuario para facilitar la comprensión de sus elementos.
2. Resulta fácil encontrar las funcionalidades en el software.
3. La interfaz hace uso de identificadores que representa claramente su significado. Ejemplo: títulos e iconos.
4. La interfaz informa a los usuarios sobre la función de los botones, iconos y ventanas de diálogo al posicionar el cursor del ratón sobre ellos.
5. Las pantallas utilizan los tipos y tamaños de letra para facilitar la visualización de los campos de entrada de datos.
6. La interfaz mantiene una estandarización en relación al formato de los iconos.

Aspectos negativos del módulo:

1. Las pantallas presentaron medianamente una distribución uniforme de su contenido según los espacios disponibles.
2. La interfaz no mantenía una estandarización en relación al idioma para las personas a las cuales va destinado el software.
3. La interfaz no mantenía una estandarización en relación a la posición de los botones y existía ambigüedad en las funciones de los mismos.
4. El sistema no proporcionaba la documentación necesaria para que el usuario realice sus tareas.

Para este tipo de prueba se llevaron a cabo 2 iteraciones donde se detectaron en la primera iteración 6 aspectos positivos y 4 negativos. Todas las dificultades fueron corregidas para la segunda iteración donde la cantidad de aspectos negativos encontrados fue nula. En la gráfica siguiente se puede entender mejor este proceso, ver ilustración 16.

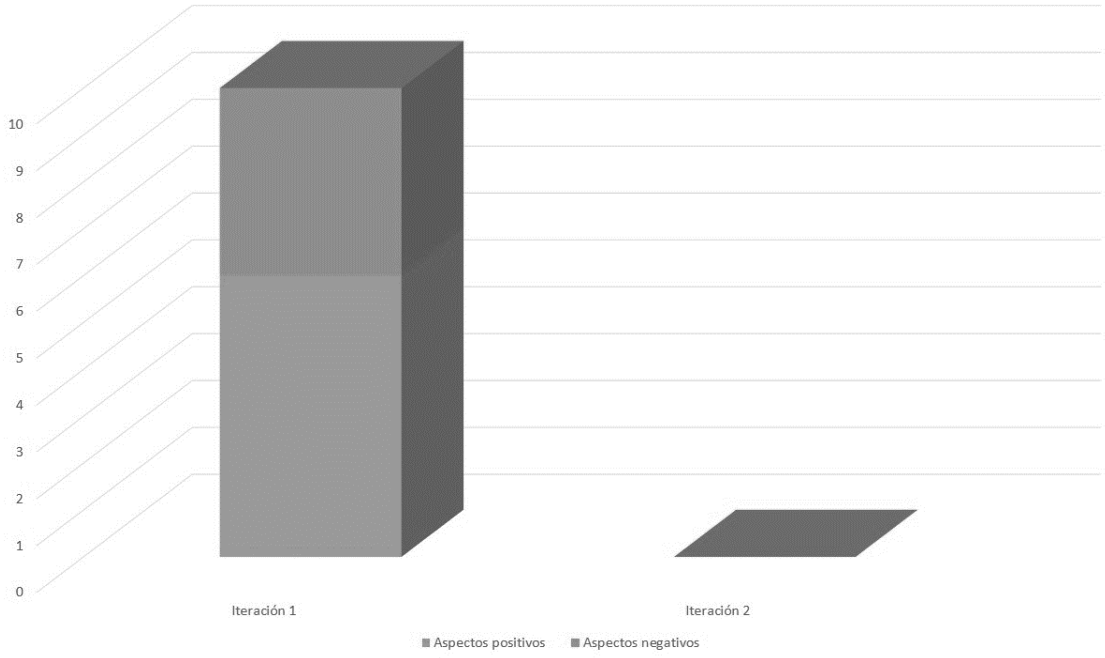


Ilustración 16: Tabla de usabilidad. Elaboración propia.

Valoraciones finales

Para evaluar el funcionamiento del módulo de cortafuegos se realizaron un conjunto de pruebas de aceptación utilizando el método de caja negra. Las pruebas permitieron corregir los errores no detectados durante la implementación, además de garantizar que la solución es completamente funcional. También se le aplicaron pruebas de usabilidad para dar solución a problemas que podía contener la interfaz, en función de contribuir a que el usuario se sienta cómodo con la aplicación y le sea fácil de usar (Ver anexo 1).

Conclusiones generales

Al desarrollar un módulo que permite configurar las reglas de un cortafuegos en el centro de control de la distribución GNU/Linux Nova Escritorio 6.0, se da cumplimiento al objetivo general planteado. Para llegar a este resultado se concluye lo siguiente:

- El estudio realizado de los centros de control de sistemas GNU/Linux y de las herramientas que se utilizan para configurar las reglas de un cortafuegos permitió identificar que existen muy pocos centros de control que permiten la configuración de un cortafuegos. Además, permitió seleccionar a ufw como parte de la solución al problema de la investigación e identificar características de la interfaz del módulo para la configuración de las reglas de un cortafuegos en el centro de control de GNOME 3.20.1 para la distribución GNU/Linux Nova Escritorio 6.0.
- La selección de las herramientas y tecnologías necesarias en la implementación del módulo para la configuración de las reglas de un cortafuegos permitió la compatibilidad de la solución con el centro de control de GNOME 3.20.1.
- El diseño y la implementación de un módulo en el centro de control permitió centralizar las configuraciones de un cortafuegos mediante una interfaz de usuario en el centro de control de GNOME 3.20.1 para la distribución GNU/Linux Nova Escritorio 6.0.
- La realización de las pruebas de aceptación y usabilidad evidenció el correcto funcionamiento del módulo añadido al centro de control de GNOME 3.20.1 para la distribución GNU/Linux Nova Escritorio 6.0, ya que la solución permite al usuario gestionar sus perfiles y reglas básicas para contribuir a la seguridad de su sistema operativo.

Recomendaciones

El módulo implementado cuenta con las funcionalidades básicas, por lo que se le debe dar un seguimiento a la siguiente recomendación:

- Añadir funcionalidades que permitan gestionar las reglas en una tabla NAT y Mangle.

Referencias bibliográficas

Campo, Gustavo Damián. 2009. http://www.ucasal.edu.ar/. *http://www.ucasal.edu.ar/.* [En línea] 2009. [Citado el: 10 de 3 de 2017.] http://www.ucasal.edu.ar/htm/ingenieria/cuadernos/archivos/4-p101-Campo.pdf.

Dennis Ritchie, Brian Kernighan. 1998. *El lenguaje de programación C.* 1998.

E. Hernández Orallo. 2009. El Lenguaje Unificado de Modelado. *(UML).* [En línea] 2009. [Citado el: 2017 de 2 de 20.] http://www.disca.upv.es/enheror/pdf/ActaUML.PDF..

Fuentes, Allan Pierra. 2011. Nova, distribución cubana de GNU/Linux. Reestructuración estratégica de. *Nova, distribución cubana de GNU/Linux.* 2011.

Gnome. 2016. GNOME. *Centro del desarrollador de GNOME.* [En línea] 2016. [Citado el: 10 de 12 de 2016.] https://developer.gnome.org/.

Hafner, Katie. 1998. *Where Wizards Stay Up Late: The Origins Of The Internet.* s.l. : Simon & Schuster, 1998. 0-684-83267-4.

Hauben, Ronda. 2007. *TCP Digest (UUCP).* 2007.

—. 2007. *TCP Digest (UUCP).* 2007.

KDE. 2016. KDE Documentation. [En línea] 2016. [Citado el: 5 de 12 de 2016.] https://docs.kde.org/.

Koch, Stefan. 2005. *Free/Open Source Software Development.* s.l. : Chicago: IGI Global, 2005.

Krause, Andrew. 2007. Build sophisticated graphical applications using one of the world's most . *Foundations of GTK+ Development.* [En línea] 2007. [Citado el: 26 de 11 de 2016.] http://sunshine.prod.uci.cu/gridfs/sunshine/books/Foundations_of_GTK_Development.pdf..

Larman, Craig. 2006. *UML y Patrones.* 2006. pág. 505.

—. 2006. *UML y Patrones.* 2006.

—. 1999. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al .* s.l. : Prentice Hall, 1999.

MarcusGoncalves. 2016. *ManualdeFirewalls.* s.l. : EditorialMcGraw-Hill, 2016.

Metaphor. 2017. Desktop Metaphor. *Csdl.tamu.edu.* [En línea] 2017. [Citado el: 2 de 3 de 2017.] http://www.csdl.tamu.edu/~l0f0954/academic/cpsc610/hw2-3.htm.

OSI. 2016. Oficina de Seguridad de Internauta. *OSI.* [En línea] Diciembre de 2016. [Citado el: 8 de Diciembre de 2016.] cortafuegos de escritorios.

Paradigm, Visual. Visual Paradigm. *Visual Paradigm.* [En línea] [Citado el: 10 de 1 de 2017.] http://www.visual-paradigm.com/product/vpuml/..

Perpiñan, Antonio. 2003. *GNU/Linux Básicamente. Fundación de código libre.* 2003.

Pressman, Roger. 2002. *Ingeniería del Software, un enfoque práctico.* s.l. : Oc-Graw Hill, 2002.

Ramos, Saily Llechú y Torrez., Dainet Torrez. *Proceso de ingeniería de requisitos en el desarrollo.* s.l. : Universidad de las Ciencias.

RODRÍGUEZ, A. y FÍRVIDA, A. 2009. *Guano, entorno de escritorio cubano, libre y de código abierto.* La

Habana : Universidad de las Ciencias Informáticas, 2009.

Rodriguez, Juan Manuel Fuentes. 2012. Entorno de Escritorio de Nova 4.0. *Entorno de Escritorio de Nova 4.0.* Habana : UCI, 2012.

Rubin, Jeffrey. 1994. *Handbook of usability testing.* 1994. 978-0-470-18548-3.

Santos Espino, J. 2009. *Introducción al lenguaje C.* 2009.

Scribd. 2017. Requerimientos funcionales y no funcionales. [En línea] 2017. [Citado el: 29 de 3 de 2017.] https://www.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales.

Share, Slide. 2017. Slide Share. *Slide Share.* [En línea] 18 de 2 de 2017. https://es.slideshare.net/JanCit0/microsoft-forefront-tmg.

Sommerville, Ian. 2005. *Ingeniería del software.* s.l. : Top printer plus, 2005. 84-7829-074-5.

—. 2006. *SoftwareEngineering, 8th ed.* 2006. p. 127. ISBN 0-321-31379-8.

Tanenbaum, Andrew S. y Wetherall, David J. 2012. *Redes de computadoras.* s.l. : Pearson Educación, 2012. Vol. IV.

Ubunto. 2017. UbuntuFirewall. [En línea] 2017. [Citado el: 25 de 3 de 2017.] https://wiki.ubuntu.com/UbuntuFirewall.

UCI. 2015. *PROGRAMA DE MEJORA, Metodología de desarrollo para la Actividad productiva de la UCI.* Habana : s.n., 2015.

Unity. 2010. Unity and Ubunto. *Unity and Ubunto.* [En línea] 2010. [Citado el: 15 de 12 de 2016.] www.markshuttleworth.com/archive/383.

Yast. 2015. Portal YaST. *YaST - openSUSE.* [En línea] 2015. [Citado el: 20 de 12 de 2016.] https://es.opensuse.org/Portal:YaST.

Zimmerman, Hubert. 2015. *OSI Reference Model.* s.l. : IEEE Transactions on Communications, 2015.

Stallings, William. 2006. *Comunicaciones y Redes de Computadoras.* 2006.

Vicedo, Daniel Serra. 2015. *Iptables. Manual y ejemplos.* 2015.

Ziegler, Robert L. 2001. *Firewalls Linux.* 1ra. s.l. : Prentice Hall PTR, 2001. ISBN 8420529494.

Cuauhtémoc Calzada de Luna, Américo. 2009. *Desarrollo de software de aplicación en modo trans parente con.* 2009.

GUFW. 2016. GUFW. *Desarrolladores Gufw.* [En línea] 2016. [Citado el: 19 de 11 de 2016.] http://gufw.org.

IPCop. 2016. IPCop. [En línea] 2016. [Citado el: 20 de 11 de 2016.] www.ipcop.org.

RedIris. 2016. RedIris. *Cortafuegos: Conceptos teóricos.* [En línea] 9 de 11 de 2016. http://www.rediris.es/cert/doc/unixsec/node23.html.

Rodríguez, Diego Lendoiro. 2005. *Maual de Iptables.* s.l. : Networking basics, 2005.

Royo, Javier. 2004. *Diseño Digital.* s.l. : Ediciones Paidós Ibérica, 2004.

Russell, Rusty. 2000. *Linux netfilter Hacking.* 2000.

Shorewall. 2016. Documentation. *Shorewall.* [En línea] 2016. [Citado el: 15 de 11 de 2016.]

http://shorewall.net/Documentation_Index.html.

Anexos

Anexo 1: Imágenes del módulo desarrollado:

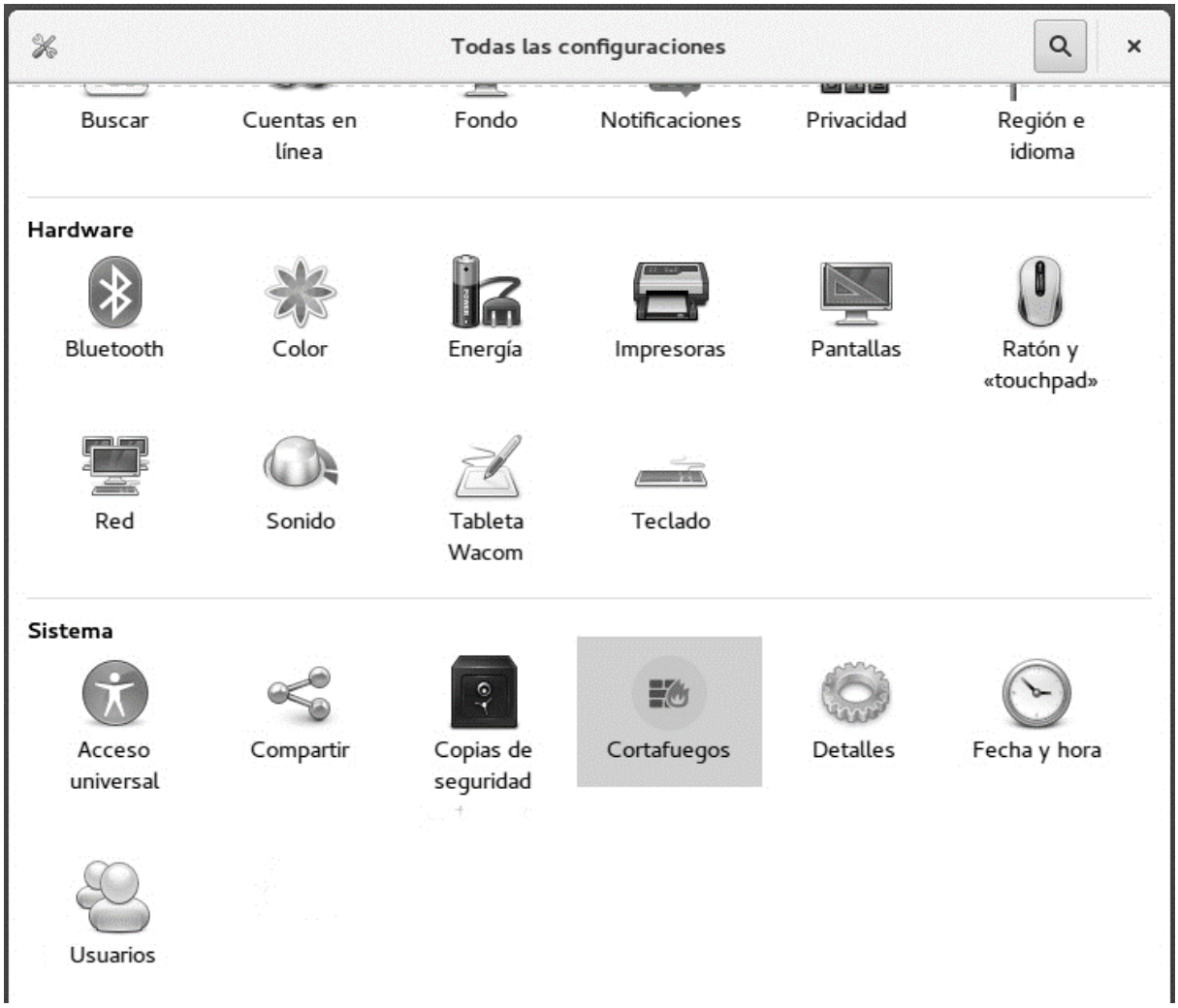


Ilustración 17: Imagen de módulo añadido al centro de control de GNOME 3.20.1. Ilustración tomada del entorno de escritorio de GNOME.

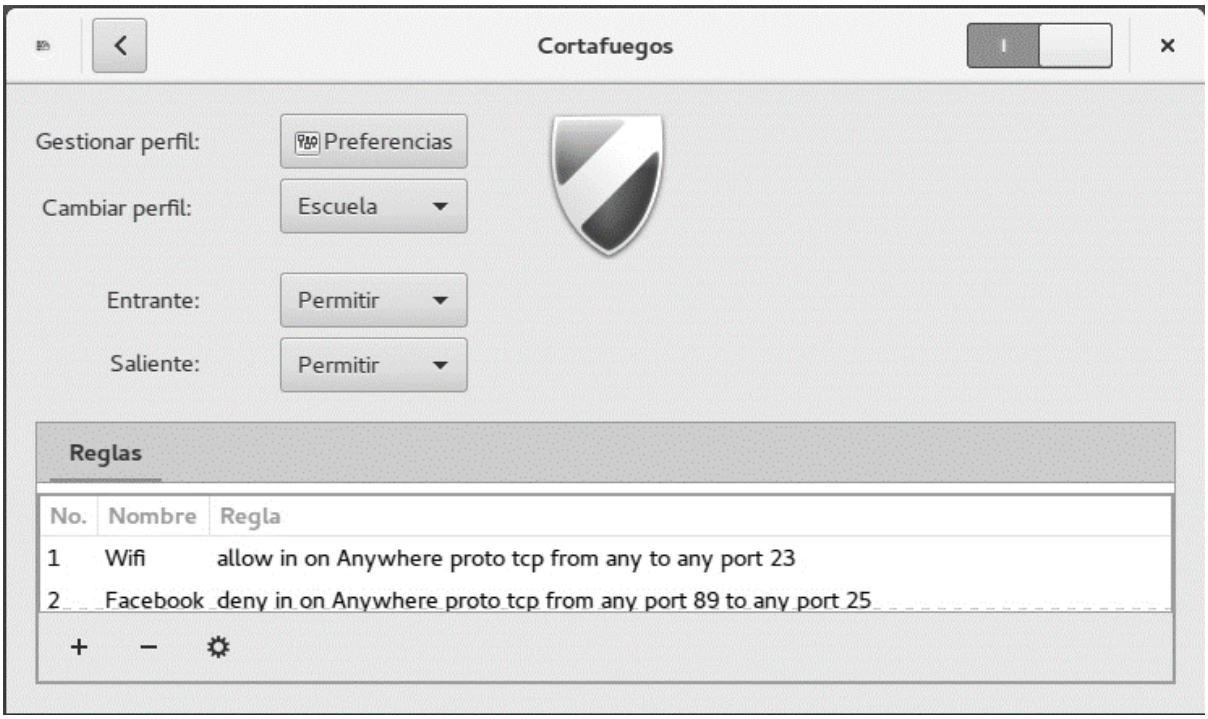


Ilustración 18: Módulo del cortafuegos. Ilustración tomada del entorno de escritorio de GNOME.

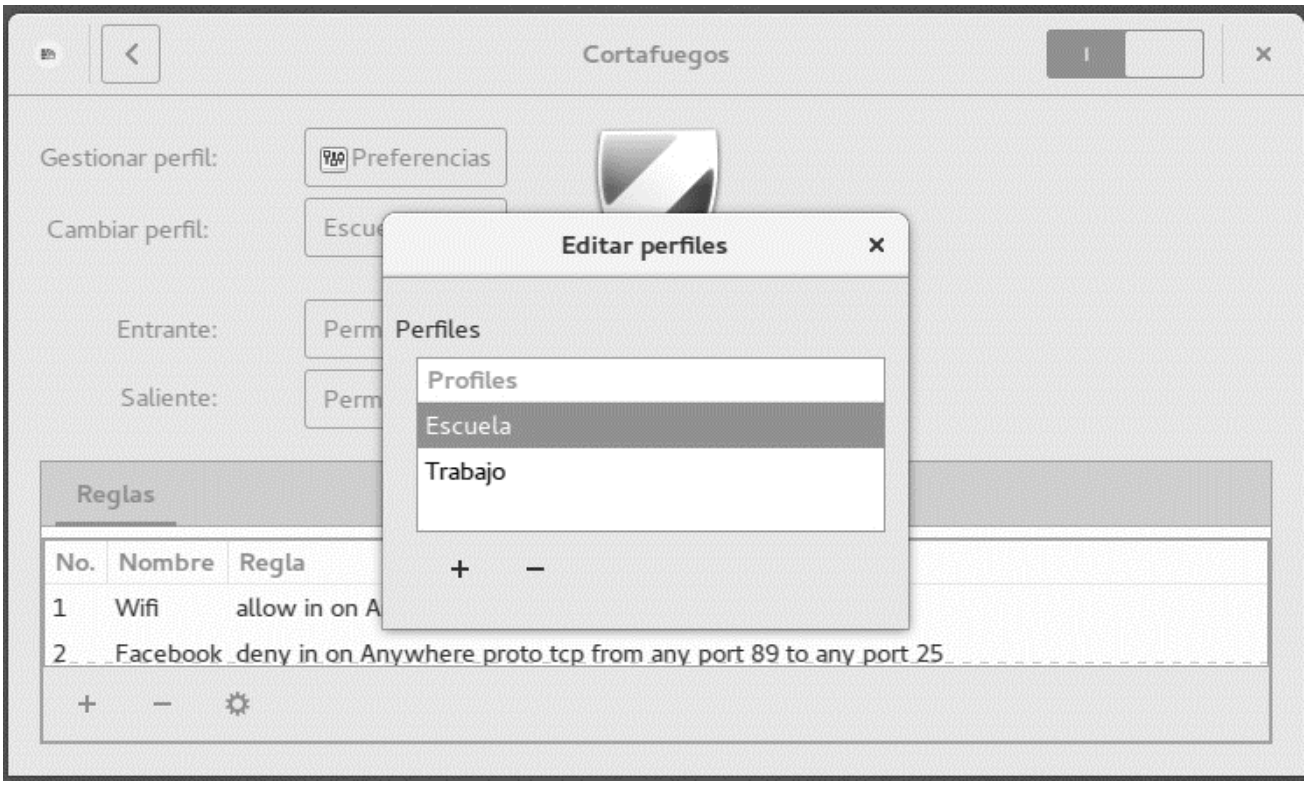


Ilustración 19: Gestionar perfiles. Ilustración tomada del entorno de escritorio de GNOME.

Ilustración 20: Crear regla simple. Ilustración tomada del entorno de escritorio de GNOME.

Ilustración 21: Crear regla avanzada. Ilustración tomada del entorno de escritorio de GNOME.

Anexo 2: Modelo de calidad para las pruebas de usabilidad *(CENDITEL, 2013)*.

Componente	Métricas asociadas al componente	Preguntas para evaluar el componente	Opciones de respuesta	Responsable
Interfaz de usuario	Usabilidad	¿Se mantiene un estándar visual en la interfaz de las operaciones del software?	Si No	Revisor de Interfaz
		¿Resulta fácil encontrar las funcionalidades en el software?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz está organizada en grupos de acuerdo a una forma lógica comprendida por el usuario?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz hace uso de identificadores que representa claramente su significado. Ejemplo: títulos, iconos, entre otros?	Si No	
		¿La interfaz informa a los usuarios sobre la función de un botón, menú, icono o ventana de diálogo al posicionar el cursor del ratón sobre el?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz permite deshabilitar algunos diálogos y presentaciones iniciales?	Si No	
		¿Las pantallas presentan una distribución uniforme de su contenido, teniendo en cuenta los espacios disponibles?	Nada Poco Medianamente Ampliamente Completamente	
		¿Las pantallas tienen áreas de selección de elementos de menú?	Si No	

Componente	Métricas asociadas al componente	Preguntas para evaluar el componente	Opciones de respuesta	Responsable
		¿Las pantallas utilizan los tipos y tamaños de letra para facilitar la visualización de los campos de entrada de datos y sus formatos?	Nada Poco Medianamente Ampliamente Completamente	
		¿Las pantallas tienen colores contrastantes, por lo que es fácil de leer?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz presenta errores gramaticales?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz presenta errores ortográficos?	Nada Poco Medianamente Ampliamente Completamente	
		¿En la interfaz se destacan las palabras en otro idioma. Por ejemplo, entre comillas, negritas, cursivas, etc?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz mantiene una estandarización en relación al idioma de las personas que usarán el software?	Nada Poco Medianamente Ampliamente Completamente	

Componente	Métricas asociadas al componente	Preguntas para evaluar el componente	Opciones de respuesta	Responsable
		¿La interfaz mantiene una estandarización en relación a la configuración de las ventanas?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz mantiene una estandarización en relación al formato de los iconos?	Nada Poco Medianamente Ampliamente Completamente	
		¿La interfaz mantiene una estandarización en relación a la posición de un determinado botón que ejecuta una misma función en ventanas de diálogo distintas (Ejemplo: botón "Cancelar", "Imprimir", "Aceptar", entre otros)?	Nada Poco Medianamente Ampliamente Completamente	