UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 1

MÓDULO PARA LA ADMINISTRACIÓN DE SAVUNIX EN NOVA PARA SERVIDORES

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

Autores:

Rosniel Arian Montanet Baños

Marisé Bonilla Marzá

Tutores:

Ing. Abel Alfonso Fírvida Donéstevez

Ing. Eugenio Rosales Rosa

La Habana

2013

"Desde el momento en que un hombre no sabe leer ni escribir, es un hombre que ha renunciado a esa herencia, es un hombre desposeído por completo de esa riqueza espiritual que la humanidad ha producido."

Fidel Castro Ruz.

Declaración de autoría

Declaramos que somos los únicos autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Rosniel Arian Montanet Baños

_____

Firma del Autor

Marisé Bonilla Marzá

_____

Firma del Autor

Ing. Abel A. Fírvida Donéstevez

_____

Firma del Tutor

Ing. Eugenio Rosales Rosa

_____

Firma del Tutor

A mi abuelo Alberto por ser mi guía y protector en todos los momentos de mi vida a pesar de no poder contar con su presencia física.

A mi abuela Delfina por darme todo el apoyo del mundo y haber permitido que sea lo que hoy soy.

A mis padres y mi hermana por estar presentes en mi vida.

A mi esposo Rosniel, por haber sido la luz en mis momentos de oscuridad y ser el tesoro que encontré durante mi andanza por la vida.

De ustedes Marisé.

A mis padres por ser lo más grande que tengo en la vida, por ser mi faro eterno, porque hoy soy el resultado del amor incondicional que han venido cultivando durante estos 24 años.

A mis hermanos Rankiel, Reinier y Yuniesky por ser una parte importante de mi vida y por permitirme ser parte de la de ellos, sobre todo mi hermano menor por ser mi compañero siempre y mi mejor amigo.

A mi maravillosa familia, a los que están y a los que lamentablemente no están físicamente a nuestro lado pero que siempre tendrán un espacio en nuestras vidas.

A mi esposa Marisé por permitirme entrar en su vida y alegrar cada momento de la mía con su presencia, amor y dedicación.

A mi Comandante en Jefe Fidel Castro Ruz, por darme la oportunidad de convertirme en el joven que soñó Martí.

Por siempre de ustedes, Rosniel.

Queremos hacer constar nuestros más profundos agradecimientos a:

A nuestra familia por su infinito apoyo, comprensión, cariño y admiración.

A nuestros suegros y cuñados por habernos dado la oportunidad de ser parte de la familia y por su apoyo constante.

A Abel Fírvida, por haber sido nuestro hermano mayor en la universidad y porque ha sido un excelente tutor con el cual estaremos siempre en deuda por el apoyo que nos ha dado en varios momentos difíciles durante nuestra carrera.

A Eugenio Rosales, por ser un excelente amigo y compañero, por haber sido un maravilloso tutor al cual le estaremos siempre agradecidos por su aporte en la realización de esta meta.

A Miguel Albalat, Jorge Luis Machín, Daniel Hernández, Dariem Pérez y Anielkis Herrera por habernos acogido como miembros del Proyecto Nova desde el 1er año de nuestra vida universitaria y habernos permito ser parte de esa gran familia que es Nova.

Al colectivo del Proyecto Nova en general por prepararnos para ser buenos profesionales.

A Eduardo Macias por ser más que un amigo, por ser la excelente persona que es y por ayudarnos en los momentos más difíciles (estamos en deuda eterna).

A la Revolución y a nuestro Comandante en Jefe Fidel Castro, por darnos la posibilidad de estudiar en la mejor universidad del país y contribuir en la formación de profesionales competentes.

A todas nuestras amistades, a las personas que nos alegraron la vida y las que no también porque nos enseñaron a no rendirnos.

Sinceramente, muchas gracias.

Resumen

Desde el comienzo de la era de la información los programas malignos han sido un tema recurrente. Estos han ido evolucionando hasta convertirse en poderosas armas en manos de las grandes potencias mundiales. Cuba, país bloqueado durante más de 50 años y asediado constantemente por provocaciones e injusticias, se ha planteado no retroceder en el proceso de la informatización de la sociedad, velando siempre por lograr la soberanía tecnológica y seguridad en las Tecnologías de la Información y las Comunicaciones (TIC). Para ello se redactó en el 2007 el artículo 50 de la Resolución 127 del Ministerio de la Informática y las Comunicaciones (MIC), el cual delega la protección contra malware a programas antivirus de producción nacional u otros autorizados oficialmente para su uso en el país.

Nova para Servidores en su versión actual utiliza a Zentyal como la plataforma para la administración de los servicios y esta herramienta no provee un módulo capaz de administrar un antivirus que satisfaga lo planteado en el artículo 50.

La presente investigación constituye un paso de avance en aras de contrarrestar la situación existente, al proponer como objetivo el desarrollo de un módulo que permita la administración en Zentyal de un antivirus que cumpla con el artículo 50 de la Resolución 127 del 2007 del MIC. Para ello se realiza un estudio de los antivirus que cumplen con el artículo y se describe el proceso de desarrollo del módulo.

Palabras Claves: Antivirus, Nova para Servidores, SAVUnix, SAVUnix Milter, Zentyal.

Índice

Resumen.....VI
Introducción.....1
Capítulo 1: Fundamentación Teórica.....4
Introducción.....4
1.1 Conceptos y definiciones fundamentales.....4
1.1.1 Antivirus de ficheros.....4
1.1.2 Antivirus de correos.....4
1.1.3 Funcionamiento de los antivirus.....5
1.1.4 Motores de búsqueda.....5
1.2 ¿Por qué es necesario proteger los sistemas informáticos?.....6
1.3 Principios para el desarrollo, uso y aplicación de las TIC en los OACE.....9
1.4 Análisis de la Resolución 127 del 2007 del MIC.....10
1.5 Antivirus que cumplen con la Resolución 127 del 2007 del MIC.....12
1.5.1 SAVUnix.....12
1.5.2 SAVUnix Milter.....13
1.5.3 Kaspersky para servidores Linux.....13
1.5.4 Kaspersky para servidores de correos Linux.....15
1.6 Comparación de SAVUnix y Kaspersky.....15
1.7 Plataforma Zentyal.....17
1.8 Metodología de desarrollo.....17
1.8.1 Fases.....18
1.9 Herramientas a utilizar.....18
1.9.1 Visual Paradigm 18
1.9.2 Geany.....18
1.9.3 Lenguaje de programación Perl.....19
Conclusiones del capítulo.....19
Capítulo 2: Análisis y diseño de la aplicación.....20
Introducción.....20
2.1 Actor del sistema.....20
2.2 Requisitos del sistema.....20
2.2.1 Requisitos funcionales (RF) agrupados por Casos de Usos (CU).....20
2.2.2 Requisitos no funcionales (RNF).....21
2.3 Diagrama de Casos de Uso del Sistema.....23
2.4 Descripción de los casos de uso.....23
2.5 Arquitectura de la aplicación 32
2.7 Diagrama de Clases del Diseño.....33
2.8 Diagrama de Secuencia 34
2.9 Patrones de diseño 34
2.9.1 Patrones GRASP 34
Conclusiones del capítulo.....36
Capítulo 3: Implementación y realización de pruebas 37
Introducción.....37

3.1 Diagrama de Componentes 37
3.2 Diagrama de Despliegue 38
3.3 Resultados obtenidos 39
3.3.1 Usando una máquina virtual con Zentyal 39
3.3.2 Usando la propia máquina 39
3.3.3 Implementación del módulo antivirus cubano para Zentyal.....40
3.3.4 Preparando el entorno de trabajo.....40
3.3.5 Estructura de un módulo de Zentyal.....41
3.3.6 Construir e instalar el paquete 42
3.4 Pruebas de software.....45
Conclusiones del capítulo.....52
Conclusiones.....54
Recomendaciones.....55
Bibliografía.....56
Glosario de términos.....59
Anexos.....60
Anexo 1: Instalar el módulo Antivirus SAVUnix.....60
Anexo 2: Diagrama de secuencia del RF Instalar el módulo Antivirus SAVUnix.....60
Anexo 3: Desinstalar el módulo Antivirus SAVUnix.....61
Anexo 4: Diagrama de Secuencia del RF Desinstalar el módulo Antivirus SAVUnix.....61
Anexo 5: Actualizar antivirus SAVUnix.....62
Anexo 6: Diagrama de Secuencia del RF Actualizar antivirus SAVUnix.....62
Anexo 7: Cargar licencia SAVUnix.....63
Anexo 8: Diagrama de Secuencia del RF Cargar licencia SAVUnix.....63
Anexo 9: Escanear directorios.....64
Anexo 10: Diagrama de Secuencia del RF Eliminar Directorio.....65
Anexo 11: Diagrama de Secuencia del RF Revisar Directorio.....65
Anexo 12: Acta de liberación del producto Nova para Servidores 2013.....66

Índice de tablas

Tabla 1: Computadoras zombies en Cuba....8
Tabla 2: Comparación entre los antivirus Kaspersky y SAVUnix....17
Tabla 3: Comparación entre los antivirus Kaspersky y SAVUnix según las 4S....18
Tabla 4: Requisitos Funcionales agrupados por Caso de Uso....23
Tabla 5: Caso de Uso “Administrar el módulo Antivirus SAVUnix”....27
Tabla 6: Caso de Uso “Actualizar antivirus SAVUnix”....29
Tabla 7: Caso de Uso “Cargar licencia del antivirus SAVUnix”....30
Tabla 8: Caso de Uso “Cargar licencia antivirus SAVUnix Milter”....31
Tabla 9: Caso de Uso “Mostrar reportes SAVUnix”....32
Tabla 10: Caso de Uso “Mostrar reportes SAVUnix Milter”....32
Tabla 11: Caso de Uso “Configurar antivirus SAVUnix Milter”....33
Tabla 12: Caso de Uso “Escanear directorio”....34
Tabla 13: Caso de Uso “Desinfectar directorio”....36
Tabla 14: Caso de Uso “Personalizar opciones de escaneo”....37
Tabla 15: Caso de Prueba “Administrar el módulo Antivirus SAVUnix”....53
Tabla 16: Caso de Prueba “Actualizar antivirus SAVUnix”....54
Tabla 17: Caso de Prueba “Cargar licencia del antivirus SAVUnix”....55
Tabla 18: Caso de Prueba “Cargar licencia antivirus SAVUnix Milter”....56
Tabla 19: Caso de Prueba “Mostrar reportes SAVUnix”....57
Tabla 20: Caso de Prueba “Mostrar reportes SAVUnix Milter”....57
Tabla 21: Caso de Prueba “Configurar antivirus SAVUnix Milter”....57
Tabla 22: Caso de Prueba “Escanear directorio”....58
Tabla 23: Caso de Prueba “Desinfectar directorio”....60
Tabla 24: Caso de Prueba “Personalizar opciones de escaneo”....60

Índice de figuras

Figura 1: Diagrama de Casos de Uso del Sistema.....26
Figura 2: Arquitectura Modelo-Vista-Controlador.....38
Figura 3: Diagrama de Clases del Diseño.....39
Figura 4: Diagrama de Secuencia del RF Adicionar Directorio.....40
Figura 5: Diagrama de Componentes.....44
Figura 6: Diagrama de Despliegue.....45
Figura 7: Estructura general del módulo Antivirus.....49

Introducción

Las primeras referencias de programas malignos datan del año 1970 (Pfleeger, 2006). Algunos de estos primeros programas eran elaborados como experimentos o bromas, no para causar graves daños (ESET, 2010) en las computadoras. Sin embargo, debido al aumento de usuarios de Internet y la acelerada revolución de la información, actualmente son usados como armas dentro de un eficaz modelo de agresión: la ciberguerra; con el fin de sabotear, robar o destruir los sistemas informáticos del enemigo.

Durante los últimos años estos ataques han aumentado considerablemente en número y envergadura, demostrando que existen organizaciones que crean programas malignos “personalizados”, para lo cual invierten mucho en recursos humanos y conocimiento.

En Cuba, atendiendo a las vulnerabilidades y debilidades propias de los sistemas informáticos y a las dificultades y limitaciones que se presentan para detectar y neutralizar oportunamente las posibles acciones del enemigo en esta esfera, se redacta en el año 2007 la Resolución 127 del Ministerio de la Informática y las Comunicaciones (MIC). En el artículo 50 de la misma se establece que: “(...) Para la protección contra virus se utilizarán los programas *antivirus de producción nacional* u *otros autorizados oficialmente* para su uso en el país, debidamente actualizados” (MIC, 2007). Este pretende minimizar los posibles daños a los sistemas y a la información que se maneja en las computadoras de los centros estatales.

Otro paso en la eliminación de estas amenazas es la migración a software libre, debido a que los sistemas operativos basados en Unix (entre ellos las distribuciones de GNU/Linux) son menos propensos a la infección por malware. Esto se atribuye al escaso número de usuarios que utilizan Linux como sistema operativo de escritorio, a que comúnmente el malware no puede actuar con permisos de administración y a las rápidas correcciones de las vulnerabilidades del sistema (Ray, 2005). Pero a pesar de esto en un servidor que comparte ficheros y servicios a distintos ordenadores pueden encontrarse numerosas amenazas sin que el sistema operativo del servidor se vea afectado.

Nova para Servidores[1] utilizando la plataforma Zentyal[2] provee un módulo que administra la revisión y/o desinfección de ficheros y servicios utilizando el antivirus Clamav. Sin embargo este no cumple con lo

1 Sistema operativo GNU/Linux desarrollado en la Universidad de las Ciencias Informáticas (UCI), que tiene por objetivo ofrecerle a los administradores de las pequeñas y medianas empresas cubanas una interfaz cómoda y eficaz para la gestión de los servicios telemáticos y cumplir con los principios para el desarrollo, uso y aplicación de las TIC en los Organismos de la Administración Central del Estado.

2 Herramienta de administración de servicios para pequeñas y medianas empresas.

regulado en el artículo 50 de la Resolución 127 del MIC.

Descrita la situación existente, se plantea como **problema científico:** ¿Cómo posibilitar la administración de un antivirus que cumpla con el artículo 50 de la Resolución 127 del 2007 del MIC en Zentyal?

El **objeto de estudio** comprende la administración de los antivirus en servidores.

El **campo de acción** se enfoca en la administración de los antivirus en servidores que cumplen con el artículo 50 de la Resolución 127 del año 2007 del MIC.

Para el desarrollo del trabajo de diploma se trazó como **objetivo general:** Desarrollar un módulo que permita la administración desde Zentyal de un antivirus que cumpla con el artículo 50 de la Resolución 127 del 2007 del MIC.

Para lograr este objetivo se plantean los siguientes **objetivos específicos**:

- Realizar un estudio del estado del arte de los antivirus que cumplen con el artículo 50.
- Implementar el módulo para la administración desde la plataforma Zentyal de un antivirus que cumpla con el artículo 50.
- Probar el módulo desarrollado.

La **idea que se defiende** es que si se desarrolla un módulo que permita la administración desde Zentyal de un antivirus que cumpla con el artículo 50, se minimizan los posibles daños a los servicios y a la información que se maneja en un servidor que utiliza Nova para Servidores.

El proceso de investigación está dirigido por las siguientes **tareas de investigación**:

- Identificación de los antivirus que cumplen con el artículo 50.
- Análisis del proceso de administración de los antivirus identificados.
- Levantamiento de los requisitos necesarios para la construcción del módulo.
- Realización del análisis y el diseño del módulo.
- Implementación de los requerimientos a partir del análisis y el diseño antes desarrollado.
- Realización de las pruebas de funcionalidad de la herramienta para comprobar su correcto funcionamiento.

El **resultado esperado** de este trabajo es proveer un módulo para la administración de un antivirus que cumpla con el artículo 50 de la Resolución 127 del 2007 del MIC en la plataforma Zentyal.

Métodos Utilizados

Con el objetivo de obtener los conocimientos a partir de los datos de la práctica y de la teoría precedente y elaborar una estrategia general para enfrentar el problema que se investiga se utilizan los siguientes métodos del diseño metodológico de la investigación científica.

Método Teórico:

- Histórico-Lógico: Para analizar la trayectoria y evolución de los antivirus en servidores y determinar las razones que fundamentan la necesidad de escoger un antivirus cubano.

Método Empírico:

- Observación: Este método posibilita revelar las relaciones esenciales y las características fundamentales de los antivirus en servidores. Permitirá obtener un registro visual del comportamiento de algunas herramientas que auxilian el desarrollo de esta aplicación.

El presente trabajo se encuentra estructurado en tres capítulos y varios anexos, a fin de exponer detalladamente todo el trabajo investigativo y práctico realizado:

- **Capítulo 1. Fundamentación Teórica:** Se procede a realizar un estudio del estado del arte sobre los antivirus que cumplen con el artículo 50 de la Resolución 127 del 2007, así como la metodología de desarrollo a seguir y las herramientas necesarias para el desarrollo del sistema propuesto.
- **Capítulo 2. Análisis y diseño de la aplicación:** Este capítulo estará enfocado a la solución del problema en cuestión, en el mismo se realizará una descripción de la propuesta del módulo. Se incluyen los diagramas que se generan según la metodología empleada, permitiendo una mejor comprensión del proceso de análisis y diseño.
- **Capítulo 3. Implementación y realización de pruebas:** A partir de los resultados obtenidos en el capítulo anterior, se procede a la implementación de los mismos, además, se utilizan las pruebas de caja negra para verificar el cumplimiento de los objetivos trazados.

Capítulo 1: Fundamentación Teórica

Introducción

El presente capítulo tiene como objetivo, presentar la fundamentación teórica de la investigación comenzando por describir conceptos relacionados con los antivirus en servidores. Se analiza la Resolución 127 del 2007 del MIC. Además, se realiza un estudio sobre los antivirus que cumplen con el artículo 50 de la Resolución 127, así como la metodología a seguir y las herramientas necesarias para el desarrollo del sistema propuesto.

1.1 Conceptos y definiciones fundamentales

1.1.1 Antivirus de ficheros

La característica primordial de cualquier antivirus actual es detectar la mayor cantidad de amenazas informáticas que puedan afectar un ordenador (ESET, 2010) y bloquearlas antes de que la misma pueda infectar un equipo, o poder eliminarla tras la infección.

1.1.2 Antivirus de correos

El correo electrónico es el método de propagación más utilizado por los virus (Álvarez, 2010), por lo que protegerlo es imperativo. La protección antivirus a nivel de servidores de correo electrónico puede ser utilizada en distintos niveles, pero siempre con el mismo objetivo: detener los virus antes de que ingresen a la red y lleguen a los usuarios. Por ello, es importante que un antivirus para servidores de correo electrónico cuente con una protección activa capaz de detectar amenazas desconocidas, ya que de esta manera provee mayor seguridad.

El correo electrónico es una aplicación informática que permite a los usuarios enviar y recibir mensajes a través de cuentas especificadas en el servidor. Los servidores de correo pueden reducirse en dos principales categorías: servidores de correo salientes y servidores de correo entrantes (Pfleeger, 2006). Los servidores de correo saliente son conocidos como SMTP (Simple Mail Transfer Protocol).

Los servidores de correo entrantes se separan en dos variedades:

- POP (Post Office Protocol) que permite almacenar los mensajes salientes y entrantes en el disco duro de la computadora.
- IMAP (Internet Message Access Protocol) que guarda copias de los mensajes.

Postfix: Es un servidor de correo de software libre que permite el enrutamiento y envío de correos electrónicos (Pfleeger, 2006). Fue creado con la intención de que fuera una alternativa más rápida, fácil de administrar y segura.

- *Zentyal-mail:* Este módulo añade el servidor de correos Postfix al Zentyal, lo que permite que la configuración del servidor sea de forma visual y más sencilla.

1.1.3 Funcionamiento de los antivirus

La información que está en el "sistema origen" debe llegar al "sistema destino" (Tori, 2008). El sistema origen podría ser un disquete y el sistema destino el disco duro del ordenador.

El funcionamiento del mecanismo de interceptación de la información varía en función de su implantación en sistemas operativos, en aplicaciones o bien de la necesidad de mecanismos especiales.

Una vez analizada la información, por el método que sea, si se ha detectado cualquier peligro, se llevan a cabo dos acciones:

1. Devolver la información limpia al mecanismo de interceptación que, a su vez, la devolverá al sistema para que siga su curso hasta el destino final.

2. Emitir una alarma a la interfaz del usuario (Corletti, 2011). En un antivirus para una estación de trabajo puede ser un mensaje mostrado por pantalla, pero en una solución para servidores la alarma puede consistir en un mensaje de correo electrónico, un mensaje a la red interna, una entrada en un informe de actividad o una comunicación de algún tipo a la herramienta de gestión del antivirus.

1.1.4 Motores de búsqueda

Independientemente de cómo se haya conseguido la información a analizar, entra en acción la parte más importante de un antivirus: el motor de búsqueda. Este motor se encarga de buscar virus en la información que ha sido interceptada y, si procede, desinfectarla.

Esta búsqueda de información se lleva a cabo de dos maneras. Una consiste en comparar la información recibida con una base de datos de virus (las llamadas "firmas de virus") (ESET Global, 2010). Si coincide la información con los patrones previamente conocidos mediante las firmas, se concluye que el fichero está infectado por un virus.

La otra manera es "averiguar" si lo que se está analizando puede ser peligroso sin saber previamente si

es un virus o no, esto se llama método heurístico. Para ello se analiza cómo se comporta la información (Pfleeger, 2006) y se compara con una lista de patrones de comportamientos peligrosos.

Por ejemplo, si se encuentra que un fichero tiene capacidad de formatear un disco duro el antivirus puede avisar al usuario. Quizá no sea un virus, sino un nuevo sistema de formateo que el usuario está instalando en el sistema; sin embargo, la acción de por sí, es peligrosa. Es el usuario, ante la alerta que le da el antivirus el que debe decidir si elimina el peligro o no.

1.2 ¿Por qué es necesario proteger los sistemas informáticos?

¿Cómo serán las guerras del futuro? Muchos autores opinan que la tercera guerra mundial se librará en campos de batalla virtuales.

En la actualidad la ciberguerra es toda una realidad. Durante el transcurso de la última década hasta el presente, han sucedido una serie de acontecimientos que han dado “motivos” para armar ejércitos de hackers, espías y grupos preparados para sabotear, robar o destruir sistemas informáticos enemigos.

En junio del año 2010 la empresa de seguridad VirusBlokada con sede en Bielorrusia descubre un ejemplar de lo que pasaría a ser el malware más sofisticado de la historia (Keizer, 2010), pasándose a llamar Stuxnet.

Stuxnet es un programa informático que se propaga mediante dispositivos de almacenamiento USB, aprovechando vulnerabilidades hasta entonces desconocidas en Windows, una vez infectado el sistema, infectará cualquier memoria USB que sea conectada a ese ordenador. Luego, identifica los ordenadores que tiene en su red local, e intenta atacarlos usando otras vulnerabilidades.

La compañía europea de seguridad digital Kaspersky Lab describía a Stuxnet como "un prototipo funcional y aterrador de un arma cibernética que conducirá a la creación de una nueva carrera armamentística mundial" (Kaspersky, 2012). Kevin Hogan, un ejecutivo de Symantec, advirtió que el 60% de los ordenadores contaminados por el gusano se encuentran en Irán (Symantec, 2012), sugiriendo que sus instalaciones industriales podrían ser su objetivo. Kaspersky concluye que los ataques sólo pudieron producirse "con el apoyo de una nación soberana" (Kaspersky, 2012). Tiempo después salía a la luz que sus creadores fueron Estados Unidos e Israel, convirtiendo a Irán en el primer objetivo de una guerra cibernética real.

Este gusano lleva un código especialmente diseñado para atacar sistemas SCADA. Los sistemas SCADA son sistemas digitales que monitorean y controlar procesos industriales como potabilización de agua, o centrifugación de uranio (McMillan, 2010). Stuxnet utilizaba vulnerabilidades desconocidas hasta la fecha

para detectar si el sistema que infecta tiene conectado un sistema SCADA y comprometer su seguridad, tomar el control del SCADA y acelerar y frenar de forma brusca los centrifugadores de uranio conectadas al SCADA.

Stuxnet hizo gran daño al proceso de centrifugado de Irán, y cientos de centrifugadoras, fueron reemplazadas de emergencia, ralentizando el programa nuclear iraní (Matrosov, 2012), sin tirar una sola bala.

Stuxnet demostró que:

- Existen “organizaciones” que crean programas malignos “personalizados” con fines de espionaje, sabotaje, robo, etc.
- La inversión en recursos humanos y conocimiento parece muy grande para limitarla a un solo programa maligno.
- Los países desarrollados poseen los recursos para construir programas malignos muy complejos. El costo de un ”desarrollo” similar es muy alto y pocos podrían pagarlo.
- Se pueden obtener y explotar vulnerabilidades no publicadas.

La bomba en el mundo de la ciberguerra la constituye el virus Flame. Este programa fue usado para llevar a cabo ataques de ciberespionaje en países del Oriente Medio. Fue descubierto en mayo de 2012 por MAHER (el Equipo de Respuesta ante Emergencias Informáticas de Irán), Karspersky Lab y el CrySyS Lab, de la Universidad de Tecnologías y Economía de Budapest.

Karspersky reveló que Flame es el software de espionaje más complejo descubierto hasta la fecha y que llevaba operativo al menos cinco años (Kaspersky, 2012).

Flame puede propagarse a otros sistemas a través de la red de área local (LAN) y mediante memorias USB. Puede grabar audio, capturas de pantalla, pulsaciones de teclado y tráfico de red. El programa también graba conversaciones de Skype[3] y puede controlar el Bluetooth para intentar obtener información de los dispositivos bluetooth cercanos (Gostev, 2012). Estos datos, junto con los documentos almacenados en el ordenador, son enviados a uno o varios servidores dispersos alrededor del mundo. Cuando termina, el programa se mantiene a la espera hasta que recibe nuevas instrucciones de esos servidores.

De acuerdo a las estimaciones de Kaspersky, Flame ha infectado aproximadamente 1000 máquinas

3 Software privativo que permite comunicaciones de texto, voz y vídeo sobre Internet.

(Kaspersky, 2012). Entre las víctimas se encuentran organizaciones gubernamentales, instituciones educativas y usuarios privados. En mayo de 2012, los países más afectados son Irán, Israel, Sudán, Siria, Líbano, Arabia Saudí y Egipto (Symantec, 2012).

En los esfuerzos de creación de Flame participaron la Agencia de Seguridad Nacional (NSA), la Agencia Central de Inteligencia (CIA) y el Ejército Israelí (Bidot, 2012).

Al final, todos estos sucesos y una infinidad más, solo terminan por demostrar que la ciberguerra es una herramienta utilizada por los gobiernos para mermar o destruir sistemas o servicios del enemigo, con la esperanza de ganar ventajas competitivas o dañar la capacidad de defensa o abastecimiento del enemigo. La ciberguerra solo es una forma más que disponen los países para atacar a sus enemigos, de hecho, prueba de esto es que Estados Unidos ha reconocido oficialmente la ciberguerra como teatro de operaciones[4].

La realidad es que luchar en esta guerra sin un sistema operativo y software soberanos, es un suicidio. Por esto Cuba ha apostado por lograr la soberanía tecnológica e incrementar la seguridad de los sistemas informáticos. Con este fin se crea Segurmática, empresa cubana que tiene como línea principal la comercialización de antivirus para distintos fines. Es precisamente esta entidad la que descubre la existencia de computadoras zombies[5] ubicadas en puntos estratégicos del país.

Tabla 1: Computadoras zombies en Cuba.

IP Fuente	Entidad
200.55.151.2	Transimport
200.55.155.165	Cubarte
200.55.166.62	Ministerio de Trabajo y Seguridad Social Habana del Este
200.55.144.46	Hotel Isla del Sur
200.14.55.138	Centro Universitario de Guantánamo
200.55.186.181	Agencia de Viajes Cubanacan
190.6.65.122	Comercializadora Instituto Tecnológico Hotelero División Territorial Varadero
200.55.139.148	Agencia de Viajes ECOTUR.SA

4 Área geográfica específica en la cual se desarrolla un conflicto armado. Una guerra tiene que desarrollarse en una proporción considerable del globo terráqueo para que se la considere suficientemente grande para tener teatros de operaciones: una guerra tiene dos o más teatros.

5 Computadoras personales que tras haber sido infectados por algún tipo de malware, pueden ser usadas por una tercera persona para ejecutar actividades hostiles.

200.55.155.165	Ministerio de Cultura
200.55.179.146	Ministerio de Trabajo y Seguridad Social Villa Clara

Fuente: Bidot, J. (2012). *Programas malignos y otras amenazas a la Seguridad Informática. Acciones para su enfrentamiento.*

1.3 Principios para el desarrollo, uso y aplicación de las TIC en los OACE

Desde el año 2010 el desarrollo de todos los productos de Nova están dirigidos al cumplimiento de los principios para el desarrollo, uso y aplicación de las Tecnologías de la Información y las Comunicaciones (TIC) en los Organismos de la Administración Central del Estado (OACE), también llamados las 4S. Estos principios son utilizados en la práctica diaria para el análisis y selección de distintas tecnologías, teniendo en cuenta que en caso de tener varias alternativas siempre se opta por las que más afines sean con las 4S.

Soberanía tecnológica

La Soberanía Tecnológica es la capacidad de un país para desarrollarse tecnológicamente de forma autónoma. No supone autarquía (independencia absoluta) sino capacidad decisional sobre su uso y desarrollo (Pierra, 2011).

La soberanía tecnológica es un aspecto indispensable para garantizar la seguridad nacional (Pierra, 2011). Sin la presencia de soberanía tecnológica no se puede hablar de seguridad total, ya que seguiríamos desarmado en medio de la ciberguerra por el hecho de no tener poder decisional sobre las tecnologías. Con el transcurso del tiempo más países lo entienden y se suman a la lucha por lograr la soberanía tecnológica, y no solo países, sino también regiones de todas las partes del mundo.

Seguridad

Para un país que ha sufrido más de 50 años de bloqueo y agresiones la seguridad adquiere una importancia significativa en los procesos industriales y económicos. Con el propósito de garantizará sistemas seguros de ataques y sin puertas traseras se debe adoptar el modelo de desarrollo colaborativo que nos propone el movimiento de software libre, el acceso al código fuente y el exhaustivo proceso de revisión y auditoría de código.

Socio-Adaptabilidad

El bloqueo económico impuesto a Cuba impide la adquisición de equipamientos y programas informáticos desde compañías norteamericanas y empresas de otras nacionalidades. Por eso es necesario que las bases tecnológicas para la informatización de Cuba, sean hechas por cubanos y para los cubanos

(Pierra, 2011), logrando inigualable adaptabilidad a las condiciones de nuestro País y garantizando la soberanía tecnológica.

Sostenibilidad

La constante asimilación e investigación de las nuevas tecnologías, la planificación, los modelos novedosos de comercialización y el uso racional de los recursos humanos, materiales y naturales, garantizará vigencia y sostenibilidad a largo plazo (Pierra, 2011). Es importante tener en cuenta que no basta con el análisis del ahorro que constituye la implantación de alguna aplicación informática basándose solamente en la sustitución de importaciones o mejoras de algún proceso productivo o administrativo. Se deberá además buscar mecanismos de comercialización que permitan ingresar divisas al país.

1.4 Análisis de la Resolución 127 del 2007 del MIC

Los avances alcanzados en los últimos años en la informatización de la sociedad y el impulso orientado por la dirección del país al desarrollo acelerado de programas que multipliquen dichos logros, requieren la adopción de medidas que garanticen un adecuado nivel de seguridad para su protección y ordenamiento.

En Cuba la Resolución 127, del año 2007, aprobada por el Ministerio de la Informática y las Comunicaciones, puso en vigor el Reglamento para las tecnologías de la información, donde a través de 100 artículos regula el funcionamiento del sistema informático nacional.

La Resolución 127 del 2007 tiene como objetivo establecer los requerimientos que rigen la seguridad de las tecnologías de la información y garantizar un respaldo legal que responda a las condiciones y necesidades del proceso de informatización del país (MIC, 2007).

En el capítulo 2 titulado *“Del Sistema de Seguridad Informática”*, se plantea en el artículo 12 que los usuarios de las tecnologías de información en órganos, organismos y entidades tienen varias obligaciones específicamente en su inciso (*g)* que se debe cumplir las reglas establecidas para el empleo de las contraseñas.

El capítulo 3 que lleva como nombre *“Empleo conveniente y seguro de las tecnologías de la información”* está compuesto por 8 secciones las cuales tienen sus particularidades, la quinta sección trata sobre *“Identificación, autenticación y control de accesos”*.

Dentro de esta sección se exponen los requisitos que se deben cumplir para hacer uso de las tan usadas contraseñas, las cuales se exponen a continuación:

- Serán privadas e intransferibles.

- Su estructura, fortaleza y frecuencia de cambio estarán en correspondencia con el riesgo estimado para el acceso que protegen.
- Combinarán en todos los casos letras y números sin un significado evidente, con una longitud mínima de 6 caracteres.
- No pueden ser visualizadas en pantalla mientras se teclean.
- No pueden ser almacenadas en texto claro (sin cifrar) en ningún tipo de tecnologías de información.

La sección 6 trata sobre la *"Seguridad ante programas malignos"*. Brindarle especial atención en esta sección al artículo 50 que plantea: "En cada entidad se implementarán los controles y procedimientos para protegerse contra virus y otros programas dañinos que puedan afectar los sistemas en explotación, así como para impedir su generalización. Para la protección contra virus se utilizarán los *programas antivirus de producción nacional* u *otros autorizados oficialmente* para su uso en el país, debidamente actualizados."

En el capítulo 4 se trata la *"Gestión de los incidentes de seguridad"*. En el capítulo 5 se le brinda atención a la *"Prestación de servicios de seguridad informática a terceros"*.

La Resolución 127 cuenta con el capítulo 6 que se titula *"De la inspección a la seguridad de las tecnologías de la información*".

Por último, el capítulo 7 aborda el tema *"De los incumplimientos"*. En este capítulo se destaca el artículo 99 que plantea:

Toda persona natural o jurídica que incumpla lo dispuesto en la presente Resolución y en las disposiciones legales vigentes en la materia, estará sujeta a la aplicación de las siguientes medidas:

> a) Invalidación temporal o definitiva de las autorizaciones administrativamente concedidas por el Ministerio de la Informática y las Comunicaciones al infractor, entre ellas, cancelación de licencias, permisos, autorizaciones, desconexión parcial o total de las redes privadas de datos y otras;
>
> b) Suspensión y/o cancelación, temporal o definitiva, de los servicios de informática y comunicaciones que hayan suscrito con empresas debidamente reconocidas y autorizadas por el Estado cubano;
>
> d) Ocupación cautelar de los medios, instrumentos, equipamientos y otros utilizados para cometer la infracción, con la finalidad de disponer posteriormente el decomiso de los mismos, según

proceda.

e) La aplicación de las medidas que correspondan, de conformidad con lo legalmente establecido.

1.5 Antivirus que cumplen con la Resolución 127 del 2007 del MIC

1.5.1 SAVUnix

La empresa cubana de seguridad informática Segurmática creada en 1995 que tiene como línea principal la comercialización de productos antivirus para Linux y Microsoft Windows (Segurmática, 2012) ofrece un grupo de servicios relacionados con la seguridad informática y otros productos de software como analizadores de registros, así como sociedad con la compañía de seguridad informática Kaspersky.

Entre sus principales productos se encuentra SAVUnix el cual es la solución antivirus cubana para la protección de sistemas de código abierto. Este puede utilizarse para el escaneo a demanda en servidores Linux. Este solo cuenta con una interfaz CLI[6] lo cual dificulta su utilización por los usuarios finales, sin embargo la configuración de tareas programadas de revisión, desinfección y actualización es muy eficiente.

Administración

Para comenzar el proceso de administración del antivirus SAVUnix es necesario haberlo instalado con anterioridad.

- **Configuración de SAVUnix**

Para realizar la configuración del antivirus se debe ejecutar el comando /opt/sav/bin/config.sh cuando se termina la instalación.

- **Búsqueda**

Se realiza de forma manual utilizando el comando *sscan*, que permite:

- Identificar: sscan <camino> -i.
- Descontaminar: sscan <camino> -d.

Al finalizar se muestra un resumen con los resultados de la búsqueda.

- **Actualización**

Se realiza ejecutando el comando /opt/sav/bin/update.sh. Se utiliza la configuración especificada al instalar el producto. Si ocurre algún fallo al iniciar con las nuevas actualizaciones se restauran las anteriores de manera automática.

6 Acrónimo de línea de comandos (Command Line Interface).

1.5.2 SAVUnix Milter

SAVUnix Milter integra el antivirus SAVUnix con servidores de correo que soporten el protocolo Milter (Segurmática, 2012). La versión 1.0 se puede obtener en las extensiones RPM y DEB lo cual permite ser instalado en las distribuciones que usen estas tecnologías. Para su instalación y correcto funcionamiento el programa requiere tener instalado y actualizado el antivirus SAVUnix.

Este antivirus analiza en tiempo real los mensajes que procesan los servidores de correo electrónico. Siendo capaz de detectar programas malignos que se ejecuten en Windows y Linux.

Administración

Para comenzar el proceso de administración del antivirus SAVUnix Milter es necesario instalar primero el antivirus SAVUnix y luego el antivirus SAVUnix Milter, este provee una configuración automática para Sendmail y Postfix.

- **Herramienta de administración**

El comando *savunixmilter-admin* permite modificar la configuración existente. Este comando provee las siguientes opciones:

  - Mostrar la configuración: savunixmilter-admin –c.
  - Desvincular del servidor de correos: savunixmilter-admin –u.
  - Vincular al servidor de correos: savunixmilter-admin –i.

- **Registros**

Los registros de los mensajes analizados y los programas malignos detectados se almacenan en /opt/savunixmilter/savunixmilter.log.

1.5.3 Kaspersky para servidores Linux

Kaspersky para servidores Linux permiten la protección de estaciones de trabajo y servidores de archivos para redes heterogéneas.

Características

- *Protección optimizada para servidores de archivos:* La arquitectura de Kaspersky ofrece protección de varias capas para servidores de archivos en Linux y redes heterogéneas que funcionan simultáneamente en el nivel del sistema de archivos y en el nivel del protocolo de transferencia de datos.
- *Consola de administración web de Kaspersky:* Muestra datos sobre el estado de la aplicación en

tiempo real y ayuda a configurar y administrar su funcionamiento.

- *Cuarentena:* Los objetos infectados, sospechosos y dañados que se detectan en el servidor del sistema de archivos pueden ser movidos a una carpeta de cuarentena, donde pueden pasar por más acciones como desinfección, eliminación, etc.

- *Alto rendimiento:* El motor del antivirus proporciona equilibrio de carga de los recursos del servidor, una tecnología optimizada de análisis antivirus y la facilidad de excluir del análisis los procesos de confianza. Estas funciones aumentan el rendimiento del producto y reducen la cantidad de recursos del sistema necesarios para realizar los análisis antivirus.

Administración

- *Instalación y administración centralizadas:* Los administradores del sistema pueden utilizar Kaspersky Security Center (un sistema de administración centralizado) para configurar y administrar de forma remota la aplicación en varios servidores al mismo tiempo.

- *Fácil instalación:* La instalación del producto demora unos pocos minutos y requiere la instalación de un solo paquete.

- *Flexible configuración del análisis:* La aplicación ofrece una amplia gama de valores de configuración, y permite a los administradores:

    - Ajustar el nivel de protección antivirus.
    - Asignar diferentes valores de configuración a diferentes usuarios que tienen acceso a objetos protegidos en el servidor de archivos.
    - Especificar excepciones de análisis.
    - Asignar acciones específicas para objetos sospechosos o infectados, incluido el tipo de amenaza.
    - Iniciar análisis de acuerdo con el programa más conveniente.

  Esta amplia gama de configuraciones permite la optimización de las cargas del servidor y garantiza una administración flexible de la seguridad de la red corporativa.

- *Sistema de elaboración de informes:* Los administradores pueden controlar la aplicación mediante informes gráficos a través de la consola web en formato PDF o XLS, o mediante Kaspersky Security Center. Mediante la línea de comandos, pueden ver los informes en formato HTML o CSV

para componentes específicos.

- *Notificaciones sobre los eventos de seguridad:* La aplicación viene con una extensa lista de eventos sobre los que puede notificarse al administrador mediante servicios de mensajes cortos (SMS), mensajería instantánea y SMTP, o mediante Kaspersky Security Center.

1.5.4 Kaspersky para servidores de correos Linux

Ofrece funciones de seguridad esenciales para el correo electrónico, entre las que se incluyen antimalware, antispam y filtrado de contenido, todo en un paquete único fácil de gestionar.

Características

- *Potente motor antispam:* Proporciona la última versión del motor antispam de Kaspersky, que incluye servicio forzado de actualizaciones antispam y filtrado de reputación basado en la nube.
- *Filtrado de archivos adjuntos:* La nueva función Format Recogniser puede filtrar archivos adjuntos usando información sobre el tipo de archivos, el nombre y el tamaño del mensaje.
- *Integración con los agentes de transferencia de correo más populares (Postfix, Sendmail, Exim, qmail y CommunigatePro):* Permite seleccionar el método de integración en función del agente de transferencia de correo, para que puedas integrarlo como filtro o usando el protocolo Milter.

1.6 Comparación de SAVUnix y Kaspersky

Tabla 2: Comparación entre los antivirus Kaspersky y SAVUnix.

Características	Kaspersky	SAVUnix
Valoración		
Calidad	Muy Bien	Muy Bien
Calidad Precio	Bien	Muy Bien
Puntuación		
Detección en análisis	Muy Bien	Muy Bien
Detección en tiempo real	Muy Bien	-
Velocidad análisis	Muy Bien	Bien
Falsas amenazas detectadas	Excelente	-
Rendimiento del equipo	Bien	Muy Bien
Interfaz	Muy Bien	-
Certificados independientes		
AV COMPARATIVES	Sí	-

VIRUS BULLETIN	No	-
ICSA LABORATORIES	Sí	-
WEST COAST LABS ANTIVIRUS	Sí	-
Protección detallada		
Anti-virus	Sí	Sí
Anti-troyanos	Sí	Sí
Anti-spyware	Sí	Sí
Anti-gusanos	Sí	Sí
Anti-phishing	Sí	Sí
Anti-rootkits	Sí	Sí
Protección Email/Pop3	Sí	Sí
Protección P2P	Sí	Sí
Protección mensajería instantánea	Sí	Sí
Compatibilidad		
Windows 8	Sí	Sí
Windows 7 Vista	Sí	Sí
Windows XP	Sí	Sí
MAC OS	Sí	No
Linux	Sí	Sí
Actualizaciones		
Automática	Sí	Sí
Manual	Sí	Sí
Frecuencia de actualizaciones	24 horas	24 horas
Protección del registro de sistema	Sí	Sí
Auto-limpieza de infecciones	Sí	Sí
Cuarentena para infecciones	Sí	-
Modo juego	Sí	-
Autodetección USB	Sí	Sí
Navegación segura	Sí	-

Después del análisis realizado de los antivirus SAVUnix y Kaspersky, en cuanto a sus principales características y funcionalidades más importantes para garantizar un nivel de seguridad apropiado, se ha llegado a la conclusión de que ambos son excelentes antivirus.

Con el propósito de que la Distribución Cubana de GNU/Linux Nova para Servidores cumpla con los principios para el desarrollo, uso y aplicación de las TIC en los OACE se decide usar el antivirus SAVUnix, ya que no es aconsejable poner en riesgo la soberanía tecnológica y la seguridad de las empresas

cubanas usando un antivirus de producción extranjera, cuando existe uno de producción nacional que satisface las necesidades del país y es más veloz en la publicación de actualizaciones de virus encontrados en Cuba. En la Tabla 3 se muestra una comparación de ambos antivirus teniendo en cuenta el cumplimiento de las 4S.

Tabla 3: Comparación entre los antivirus Kaspersky y SAVUnix según las 4S.

Características	Kaspersky	SAVUnix
Principios para el desarrollo, uso y aplicación de las TIC en los OACE		
Soberanía	No	Sí
Socio-Adaptabilidad	No	Sí
Seguridad	No	Sí
Sostenibilidad	Sí	Sí

1.7 Plataforma Zentyal

Zentyal se desarrolló con el objetivo de acercar Linux a las pymes[7] (Zentyal, 2010) y permitirles aprovechar todo su potencial como servidor de empresa. Está basado en la popular distribución Ubuntu y permite administrar todos los servicios (Zentyal, 2012) de una red informática, tales como el acceso a Internet, la seguridad de la red, la compartición de recursos, la infraestructura de la red o las comunicaciones, de forma sencilla y a través de una única plataforma.

Esta plataforma posee una interfaz intuitiva que incluye únicamente aquellas funcionalidades de uso más frecuente, aunque también dispone de los medios necesarios para realizar toda clase de configuraciones avanzadas. Otra de las características importantes de Zentyal es que todas sus funcionalidades están estrechamente integradas entre sí, automatizando la mayoría de las tareas (Zentyal, 2011) y ahorrando tiempo en la administración de sistemas.

1.8 Metodología de desarrollo

Las metodologías de desarrollo de software definen una serie de procedimientos, técnicas y herramientas (Pressman, 2001) para la realización de un producto de software. Para el desarrollo de la solución propuesta se toma **OpenUp**[8]. Esta metodología de desarrollo es un proceso unificado (de aplicación general) y ágil (se centra en el desarrollo rápido de sistemas) (Open UP, 2012) que involucra un conjunto mínimo de prácticas que ayudan a los equipos de trabajo a ser más efectivos en el desarrollo de sistemas

7 Pequeña y mediana empresa (pyme) es una empresa con características distintivas, y tiene dimensiones con ciertos límites ocupacionales y financieros prefijados por los Estados o regiones.

8 Open Unified Process en español Proceso Unificado Abierto.

software.

1.8.1 Fases

Fase Inicio: El propósito en esta fase es lograr concurrencia entre todos los stakeholders sobre los objetivos del ciclo de vida para el proyecto (Open UP, 2012). Se realiza la estimación inicial del coste del proyecto y su planificación. Se identifican los casos de uso críticos.

Fase Elaboración: En esta fase se realizan tareas de análisis del dominio y definición de la arquitectura del sistema. Si se decide continuar con el proyecto se debe elaborar un plan de proyecto en esta fase, para lo cual se deben establecer unos requisitos y arquitectura estables (Open UP, 2012). Al concluir esta fase se debe haber definido los casos de uso, los actores, la arquitectura del sistema y un prototipo ejecutable de la misma.

Fase Construcción: Todos los componentes y funcionalidades del sistema que falten por implementar son realizados, testeados e integrados en esta fase. Los resultados obtenidos en forma de incrementos ejecutables deben ser desarrollados de la forma más rápida posible sin dejar de lado la calidad de lo desarrollado (Open UP, 2012) El propósito de esta fase es completar el desarrollo del sistema basado en la arquitectura.

Fase Transición: El propósito de esta fase es asegurar que el producto software está listo (Open UP, 2012) para ser distribuido a los usuarios.

1.9 Herramientas a utilizar

1.9.1 Visual Paradigm

Esta herramienta que soporta el Lenguaje Unificado de Modelado (UML) y permite generar artefactos del ciclo de vida del desarrollo de software (Visual Paradigm, 2012), ayuda a una rápida construcción de aplicaciones de calidad; permite elaborar todo tipo de diagramas de clases, código inverso, generar códigos desde diagramas y generar documentación.

1.9.2 Geany

La herramienta seleccionada para el desarrollar el módulo para la administración de SAVUnix usando la plataforma Zentyal es el Geany. Este es un editor de texto pequeño y ligero con características básicas que utiliza librerías GTK (Geany, 2012) para su funcionamiento. Las principales características del Geany son: resaltado de sintaxis, autocompletamiento, fácil gestión de proyectos, soporte para extensiones y

muchos tipos de archivos, tales como: C, Java, PHP, Python, Perl, entre otros.

1.9.3 Lenguaje de programación Perl

Perl es un lenguaje de programación que toma características principalmente del lenguaje C. Es un lenguaje imperativo, con variables, expresiones, asignaciones, bloques de código delimitados por llaves, estructuras de control y subrutinas (Perl, 2012). Perl también toma características de la programación shell. Todas las variables son marcadas con un el signo “$” este identifica inequívocamente los nombres de las variables, permitiendo tener una amplia sintaxis. Este lenguaje tiene muchas funciones integradas para tareas comunes y para acceder a los recursos del sistema. Perl toma las listas del Lisp y hash (memoria asociativa) del AWK (Merelo, 2009). Todo esto simplifica y facilita todas las formas del análisis sintáctico, manejo de texto y tareas de gestión de datos.

Conclusiones del capítulo

- La ciberguerra se perpetra a través de sabotajes y espionaje, utilizando las posibilidades de Internet. Esta situación no hace que Cuba renuncie al uso de las nuevas tecnologías, al contrario, continua ampliando su potencial, teniendo en cuenta la soberanía tecnológica y la seguridad con el uso de software libres y de código abierto en sustitución del código propietario.
- Los programas malignos forman parte de las armas utilizadas para realizar ataques y dañar los sistemas “enemigos”. Para evitar que estos ataquen el ordenador surgen los programas antivirus que no son más que sistemas que analizan información de diferentes maneras dependiendo de dónde provengan y, en caso de que se encuentre infectada, proceden a su desinfección.
- Se analizó la Resolución 127 del 2007 del MIC y se estudiaron los antivirus SAVUnix y Kaspersky por ser los únicos que cumplen con el artículo 50 de la Resolución.
- Se decidió que Nova para Servidores utilizará el antivirus cubano SAVUnix, ya que además de cumplir con el artículo 50, cumple con los principios de soberanía tecnológica y seguridad, satisface las necesidades del país y es más veloz en la publicación de actualizaciones de virus encontrados en Cuba.
- Se utilizará la metodología de desarrollo OpenUP para estructurar, planificar y controlar el proceso de desarrollo del módulo, la herramienta Visual Paradigm para generar los artefactos del ciclo de vida del desarrollo del módulo y el editor de texto Geany para la programación del módulo usando el lenguaje Perl.

Capítulo 2: Análisis y diseño de la aplicación

Introducción

Teniendo en cuenta los elementos estudiados en el capítulo anterior, en el presente se recogen los elementos del análisis y el diseño utilizando la metodología OpenUP, haciendo posible el desarrollo de una aplicación robusta. Para ello se definen los requisitos funcionales agrupados por caso de uso y los requisitos no funcionales del módulo, luego se realiza la descripción de cada caso de uso. Se define las clases que serán utilizadas en el sistema y las relaciones entre ellas, así como la arquitectura de la herramienta y los patrones diseño utilizados para la asignación de responsabilidades.

2.1 Actor del sistema

- **Administrador:** Es la persona encargada de hacer toda la gestión de la información que brindará el sistema.

2.2 Requisitos del sistema

La solución propuesta es un módulo para la plataforma de desarrollo Zentyal que permita la administración del antivirus SAVUnix. El mismo deberá cumplir con los requisitos que a continuación se relacionan:

2.2.1 Requisitos funcionales (RF) agrupados por Casos de Usos (CU)

Son capacidades o condiciones que el sistema debe cumplir.

Tabla 4: Requisitos Funcionales agrupados por Caso de Uso.

Caso de Uso	Requisitos
Administrar el módulo Antivirus SAVUnix.	Instalar el módulo Antivirus SAVUnix.
	Desinstalar el módulo antivirus SAVUnix.
Actualizar antivirus SAVUnix.	Actualizar antivirus SAVUnix.
Cargar licencia SAVUnix.	Cargar licencia del antivirus SAVUnix.
Cargar licencia SAVUnix Milter.	Cargar licencia del antivirus SAVUnix Milter.
Mostrar reportes SAVUnix.	Mostrar reportes SAVUnix.
Mostrar reportes SAVUnix Milter.	Mostrar reportes SAVUnix Milter.
Configurar antivirus SAVUnix Milter.	Configurar antivirus SAVUnix Milter.
Escanear directorio.	Adicionar directorio.
	Eliminar directorio.
	Modificar directorio.

	Revisar directorio.
Desinfectar directorio.	Desinfectar directorio.
Personalizar opciones de escaneo.	Personalizar opciones de escaneo.

2.2.2 Requisitos no funcionales (RNF)

Son las propiedades o cualidades que el producto debe tener.

Apariencia o interfaz externa

RNF1: La aplicación deberá presentar y mantener una interfaz amigable, interactiva, intuitiva y entendible por el usuario.

RNF2: Los mensajes mostrados al usuario deben seguir los patrones definidos por la plataforma Zentyal.

Usabilidad

RNF3: Se debe utilizar el idioma que predomina en el entorno donde será utilizado.

RNF4: El administrador permanecerá en el sistema el tiempo que así lo considere.

RNF5: Para el trabajo con el sistema se requerirán conocimientos mínimos.

Accesibilidad

RNF6: La información y las funcionalidades estarán disponibles y el administrador podrá acceder a ella en todo momento.

Disponibilidad

RNF7: El sistema siempre estará disponible para el administrador, dependiendo solamente de la operatividad de la plataforma Zentyal.

Rendimiento

RNF8: El tiempo de respuesta y procesamiento de la información dependerán de las condiciones del hardware y de la cantidad de tareas a realizar.

Legales

RNF9: Las herramientas seleccionadas para el desarrollo del producto están respaldadas por licencias libres, bajo las condiciones de software libre. En el caso de la herramienta Visual Paradigm como no es libre se utiliza la licencia que posee la universidad.

RNF10: La aplicación y toda la documentación generada pertenecen al grupo de proyecto Gestión Documental y Archivística y a la Universidad de las Ciencias Informáticas.

Software

RNF11: Las PC clientes deben tener instalado un navegador web.

RNF12: La PC servidor debe contar con el servidor web Apache.

RNF13: La PC servidor debe tener instalada la plataforma Zentyal.

Hardware

RNF14: El servidor de aplicaciones requiere una RAM de 2 GB o superior para garantizar un mejor funcionamiento.

RNF15: El servidor de aplicaciones requiere de al menos 10GB disponible para su correcto funcionamiento.

Soporte

RNF16: Realizar pruebas y mantenimiento necesarios para lograr el mejoramiento y evolución en el tiempo.

Confidencialidad

RNF17: La información manejada por el sistema está protegida de acceso no autorizado.

Restricciones de diseño e implementación

1. El servidor debe tener instalado el servidor web Apache.
2. El editor de texto Geany para la implementación del módulo, utilizando como lenguaje de programación Perl.
3. Para la modelación del sistema se utilizará la herramienta Visual Paradigm.
4. La metodología de desarrollo de software empleada será OpenUp, haciendo uso del Lenguaje de Modelación Unificado (UML).

2.3 Diagrama de Casos de Uso del Sistema

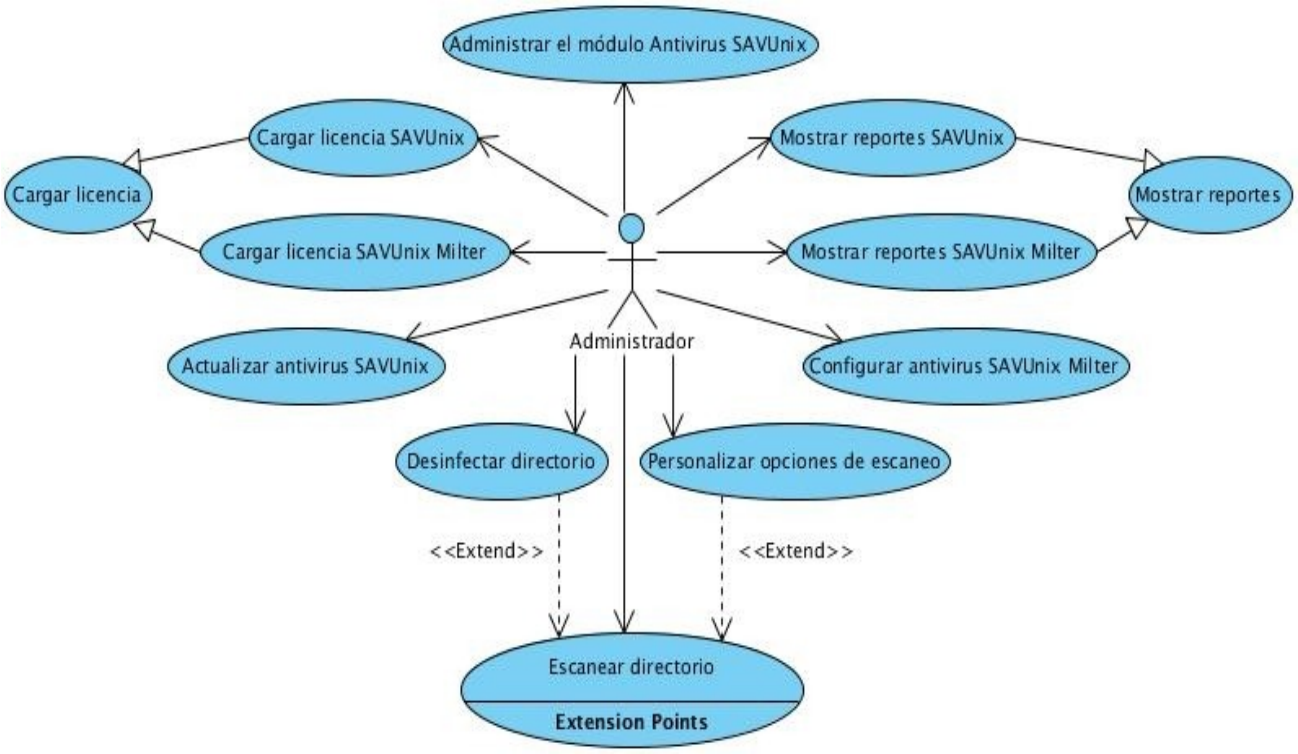


Figura 1: Diagrama de Casos de Uso del Sistema.

2.4 Descripción de los casos de uso

A continuación se describen textualmente los casos de uso del sistema que fueron modelados en el diagrama anterior, especificando su propósito y sus condiciones de existencia.

CU1: Administrar el módulo Antivirus SAVUnix

Tabla 5: Caso de Uso “Administrar el módulo Antivirus SAVUnix”.

Caso de Uso	Administrar el módulo Antivirus SAVUnix.
Objetivo	Permite administrar e módulo Antivirus SAVUnix.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Gestión de software/Componentes de Zentyal* donde se muestra las pestañas con las opciones de instalación y desinstalación.
Complejidad	Media.
Prioridad	Crítico.

Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal y el módulo de antivirus debe existir en el repositorio.
Poscondiciones	El antivirus será instalado o desinstalado.
Referencias	RF1, RF2.

Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la opción *Gestión de software/Componentes de Zentyal*.	2. Muestra la interfaz Componentes de Zentyal que contiene las pestañas: • *Instalar* (Véase escenario Instalar el módulo antivirus SAVUnix). • *Borrar* (Véase escenario Desinstalar el módulo antivirus SAVUnix).
Escenario: "Instalar el módulo antivirus SAVUnix"	
Acción de actor	**Respuesta del Sistema**
3. Selecciona la pestaña *Instalar*.	4. Muestra un listado con todos los módulos que pueden ser instalados.
5. En la sección *Seleccionar* marca el cuadro correspondiente al módulo *Antivirus SAVUnix*. 6. Presiona el botón *Instalar*.	7. Muestra mensaje de confirmación.
8. Presiona el botón *Aceptar*.	9. Instala el módulo. 10. Muestra un mensaje de éxito.
Flujos Alternos	
Acción del actor	**Respuesta del Sistema**
8.1 Presiona el botón *Cancelar*.	9.1 Cierra el mensaje de confirmación.
Escenario: "Desinstalar el módulo antivirus SAVUnix"	
Acción del actor	**Respuesta del Sistema**
3. Selecciona la pestaña *Borrar*.	4. Muestra un listado con todos los módulos que pueden ser desinstalados.

5. En la sección *Seleccionar* marca el cuadro correspondiente al módulo *Antivirus SAVUnix*. 6. Presiona el botón *Borrar*.	7. Muestra mensaje de confirmación.
8. Presiona el botón *Aceptar*.	9. Desinstala el módulo. 10. Muestra un mensaje de éxito.
Flujos Alternos	
Acción del actor	**Respuesta del Sistema**
8.1 Presiona el botón *Cancelar*.	9.1 Cierra el mensaje de confirmación.

CU2: Actualizar antivirus SAVUnix

Tabla 6: Caso de Uso “Actualizar antivirus SAVUnix”.

Caso de Uso	Actualizar antivirus SAVUnix.
Objetivo	Permite actualizar el antivirus y configurar las opciones de actualización.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Configuración* y selecciona la pestaña *Antivirus de fichero*, luego podrá configurar las opciones de actualización y procederá a la actualización del antivirus.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber instalado el antivirus y haber cargado la licencia del software.
Poscondiciones	Se actualizará el antivirus teniendo en cuenta la configuración realizada.
Referencias	RF3.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de Fichero*.	2. Muestra la interfaz de Actualización del antivirus donde se configura las opciones de actualización.
3. Entra los datos de configuración. 4. Presiona el botón *Actualizar*.	5. Guarda la nueva configuración. 6. Actualiza el antivirus.
Flujos Alternos	**Flujos Alternos**

Acción del actor	**Respuesta del Sistema**
3.1 Entra datos incorrectos.	4.1 Muestra mensaje de error.

CU3: Cargar licencia del antivirus SAVUnix

Tabla 7: Caso de Uso “Cargar licencia del antivirus SAVUnix”.

Caso de Uso	Cargar licencia del antivirus SAVUnix.
Objetivo	Permite cargar la licencia del antivirus SAVUnix.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Configuración* y luego selecciona la pestaña *Antivirus de fichero*, luego podrá cargar la licencia del antivirus SAVUnix.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber instalado el antivirus.
Poscondiciones	Se carga la licencia del antivirus.
Referencias	RF4.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de fichero*.	2. Muestra un campo que permite *Examinar* los directorios locales para seleccionar la licencia.
3. Examina los directorios locales y selecciona la licencia. 4. Presiona el botón *Cargar Licencia*.	5. Carga la licencia del antivirus.
Flujos Alternos	**Flujos Alternos**
Acción del actor	**Respuesta del Sistema**
3.1 Entra datos incorrectos.	4.1 Muestra mensaje de error.

CU4: Cargar licencia del antivirus SAVUnix Milter

Tabla 8: Caso de Uso “Cargar licencia antivirus SAVUnix Milter”.

Caso de Uso	Cargar licencia del antivirus SAVUnix Milter.
Objetivo	Permite cargar la licencia del antivirus SAVUnix Milter.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Configuración* y selecciona la pestaña *Antivirus de correo,* luego podrá cargar la licencia del antivirus SAVUnix Milter.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber instalado el antivirus SAVUnix.
Poscondiciones	Se carga la licencia del antivirus SAVUnix Milter.
Referencias	RF5.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de correo*.	2. Muestra un campo que permite *Examinar* los directorios locales para seleccionar la licencia.
3. Examina los directorios locales y selecciona la licencia. 4. Presiona el botón *Cargar Licencia*.	5. Carga la licencia del antivirus.
Flujos Alternos	
Acción del actor	**Respuesta del Sistema**
3.1 Entra datos incorrectos.	4.1 Muestra mensaje de error.

CU5: Mostrar reportes SAVUnix

Tabla 9: Caso de Uso “Mostrar reportes SAVUnix”.

Caso de Uso	Mostrar reportes SAVUnix.
Objetivo	Muestra los reportes del último escaneo del antivirus SAVUnix.
Actores	Administrador.

Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Reportes* y selecciona la pestaña *Antivirus de fichero*, permitiendo esta visualizar los reportes dados por el antivirus durante el último escaneo.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El administrador debe haber configurado el antivirus SAVUnix.
Referencias	RF6.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de fichero.*	2. Muestra los reportes dados por el antivirus SAVUnix durante el último escaneo realizado.

CU6: Mostrar reportes SAVUnix Milter.

Tabla 10: Caso de Uso “Mostrar reportes SAVUnix Milter”.

Caso de Uso	Mostrar reportes SAVUnix Milter.
Objetivo	Muestra los reportes del último escaneo del antivirus SAVUnix Milter.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Reportes* y selecciona la pestaña *Antivirus de correo*, permitiendo esta visualizar los reportes dados por el antivirus de correo durante el último escaneo.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	El administrador debe haber configurado el antivirus SAVUnix Milter.
Referencias	RF7.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de correo.*	2. Muestra los reportes dados por el antivirus SAVUnix Milter durante el último escaneo realizado.

CU7: Configurar antivirus SAVUnix Milter.

Tabla 11: Caso de Uso “Configurar antivirus SAVUnix Milter”.

Caso de Uso	Configurar antivirus SAVUnix Milter.
Objetivo	Configurar el antivirus SAVUnix Milter.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Configuración* y selecciona la pestaña *Antivirus de correo*, permitiendo esta configurar las opciones del antivirus SAVUnix Milter.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber instalado el antivirus SAVUnix.
Referencias	RF8.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la pestaña *Antivirus de correo.*	2. Muestra la interfaz de configuración del antivirus SAVUnix Milter donde se muestran diversas opciones de configuración.
3. Entra los datos de configuración. 4. Presiona el botón *Guardar*.	5. Guarda la nueva configuración.

CU8: Escanear directorio

Tabla 12: Caso de Uso “Escanear directorio”.

Caso de Uso	Escanear directorio.
Objetivo	Permite gestionar los directorios a escanear y detectar los directorios infectados.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Antivirus SAVUnix/Escanear directorio*. El antivirus realizará el escaneo teniendo en cuenta la personalización realizada.
Complejidad	Alta.

Prioridad	Crítico.
Precondiciones	El administrador debe haber configurado el antivirus.
Poscondiciones	Se generará reportes con los resultados del escaneo
Referencias	RF9, RF10, RF11, RF12.

Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona la opción *Escanear directorio*.	2. Muestra la interfaz de Escaneo que contiene las opciones: • Añadir nuevo/a (Véase escenario Adicionar directorio) • Listado con los directorios que se tienen seleccionados para su posterior escaneo. Este listado contiene las opciones de Eliminar (Véase escenario Eliminar directorio) y Modificar (Véase escenario Modificar directorio) para cada uno de los directorios existentes. • Escanear (Véase escenario Revisar directorios)
Escenario: "Adicionar directorio"	
Acción de actor	**Respuesta del Sistema**
3. Selecciona la opción *Añadir nuevo/a*.	4. Muestra un campo para introducir el directorio.
5. Llena el campo con la ruta del directorio deseado. 6. Presiona el botón *Añadir*.	7. Agrega la dirección al listado de directorios.
Flujos Alternos	
Acción del actor	**Respuesta del Sistema**
5.1 Entra datos incorrectos.	4.1 Muestra mensaje de error.
Escenario: "Eliminar directorio"	
Acción del actor	**Respuesta del Sistema**
3. En la columna *Acción* del listado de directorios	4. Elimina el directorio del listado.

selecciona la opción *Borrar* (imagen) correspondiente al directorio que desea eliminar.	5. Muestra mensaje de éxito.
Escenario: "Modificar directorio"	
Acción del actor	**Respuesta del Sistema**
3. En la columna *Acción* del listado de directorios selecciona la opción *Editar* (imagen) correspondiente al directorio que desea modificar.	4. Muestra una interfaz para modificar el directorio.
5. Introduce los nuevos datos. 6. Presiona el botón *Cambiar*.	7. Se modifica el directorio.
Escenario: "Revisar directorio"	
Acción del actor	**Respuesta del Sistema**
3. Presiona el botón *Escanear*.	4. Comienza la búsqueda de malware en los directorios seleccionados.

CU9: Desinfectar directorio

Tabla 13: Caso de Uso "Desinfectar directorio".

Caso de Uso	Desinfectar directorio.
Objetivo	Permite desinfectar los directorios que hayan sido reportados como contaminados.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción *Descontaminar,* luego se procederá a la desinfección de los ficheros contaminados.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber seleccionado directorios para escanear.
Poscondiciones	Se desinfectarán los directorios contaminados.
Referencias	RF13.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Marca la opción *Descontaminar.*	2. Desinfecta los ficheros contaminados.

CU10: Personalizar opciones de escaneo

Tabla 14: Caso de Uso “Personalizar opciones de escaneo”.

Caso de Uso	Personalizar opciones de escaneo.
Objetivo	Permite desinfectar los directorios que hayan sido reportados como contaminados.
Actores	Administrador.
Resumen	El caso de uso inicia cuando el administrador selecciona las opciones de escaneo.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe haber seleccionado directorios para escanear.
Poscondiciones	Se guarda en /etc/crontab la nueva configuración.
Referencias	RF14.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. Selecciona las opciones de escaneo.	2. Se guardan la nueva personalización en el fichero */etc/crontab* del sistema.

2.5 Arquitectura de la aplicación

Para la arquitectura de la aplicación se hará uso del patrón Modelo-Vista-Controlador. El mismo permite separar los datos en tres componentes distintos:

- Modelo: Administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).
- Vista: Maneja la visualización de la información.
- Controlador: Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

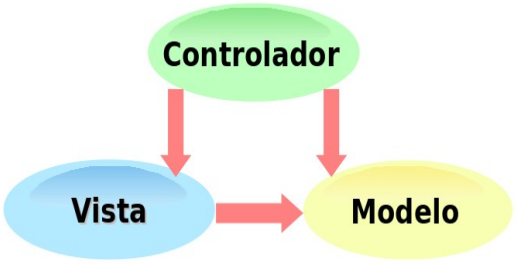


Figura 2: Arquitectura Modelo-Vista-Controlador.

2.7 Diagrama de Clases del Diseño

Representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. Un diagrama de clases está compuesto por los siguientes elementos: Clase: atributos, métodos y visibilidad; y Relaciones: herencia, Composición, Agregación, Asociación y Uso (Sommerville, 2005).

En la figura se muestran las once clases definidas para la propuesta de solución. La clase Antivirus, es la encargada de recoger todos los parámetros de entrada al sistema. Se administra el antivirus teniendo en cuenta que puede ser de 2 tipos: de fichero y de correo, cada tipo es contiene las clases necesarias de acuerdo a sus funcionalidades.

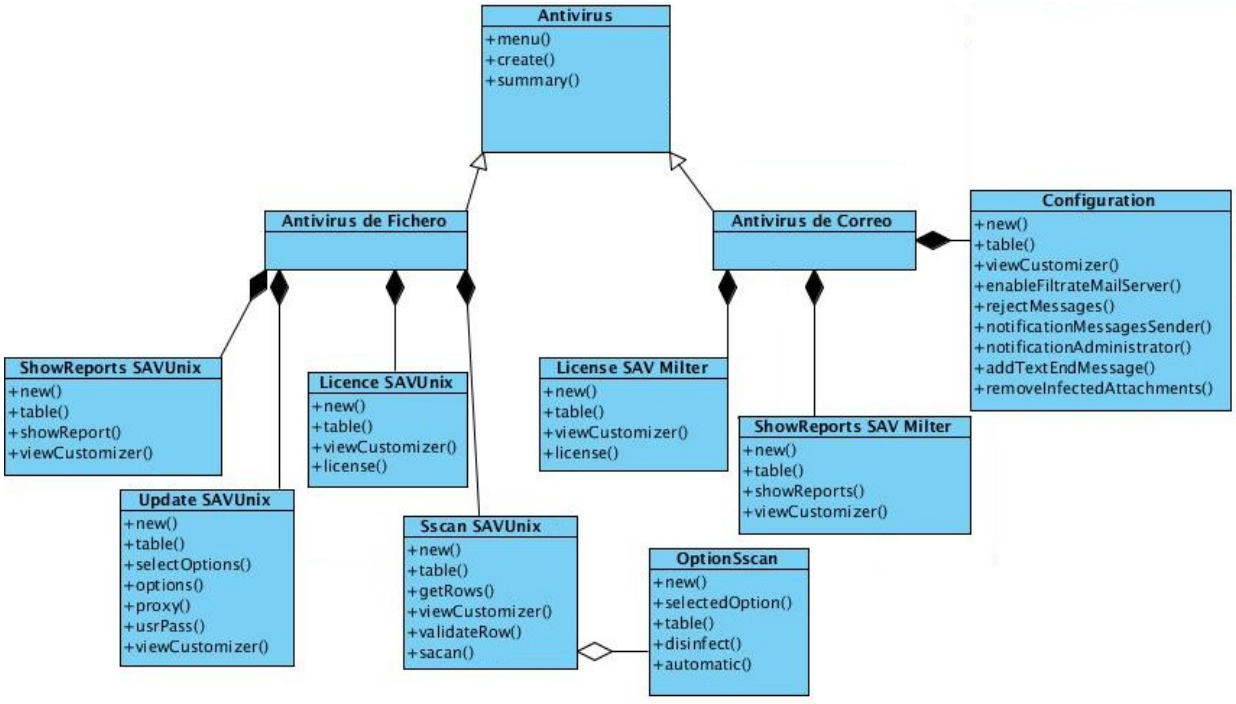


Figura 3: Diagrama de Clases del Diseño.

2.8 Diagrama de Secuencia

Un Diagrama de Secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario (Piñero, 2008), y mensajes intercambiados entre los objetos.

A continuación se muestran el Diagrama de Secuencia correspondiente al requisito funcional *Adicionar directorio* que forma parte del caso de uso *Escanear Directorio*.

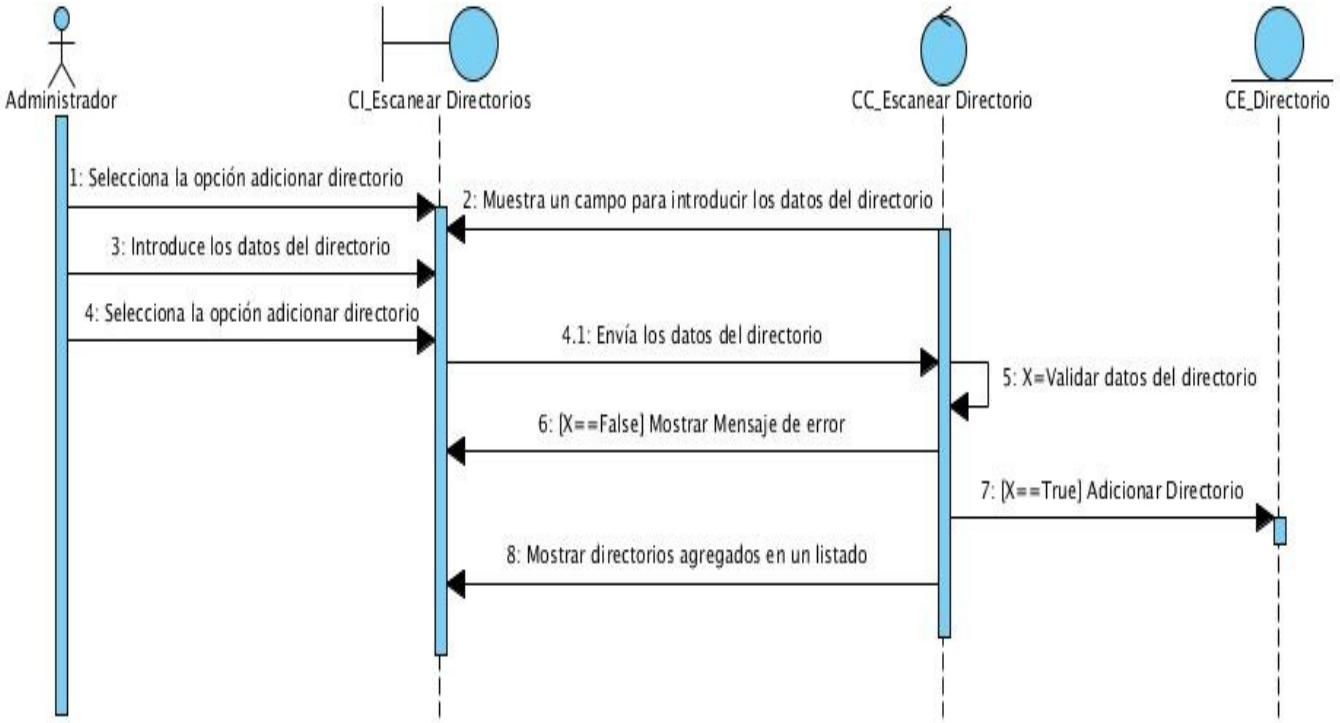


Figura 4: Diagrama de Secuencia del RF Adicionar Directorio.

2.9 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces (Larman, 2004), brindando de esta forma una solución ya probada y documentada.

2.9.1 Patrones GRASP

GRASP es un acrónimo de General Responsibility Assignment Software Patterns o Patrones Generales de Software para Asignar Responsabilidades. Los patrones GRASP son utilizados para describir los

principios fundamentales del diseño y la asignación de responsabilidades (Pressman, 2001). Los patrones GRASP utilizados para el diseño del módulo para la administración de SAVUnix utilizando la plataforma Zentyal son:

- ***Experto en información:*** Indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada, posibilitando la disminución del acoplamiento.

  Este patrón se pone de manifiesto en todas las clases del módulo diseñado, ya que cada clase tiene las responsabilidades que corresponden a la información que manejan. Por ejemplo, la clase *Configuration* es la única que maneja información referente a la configuración del antivirus SAVUnix Milter, por tanto es la que tiene la responsabilidad de realizar la configuración y contiene los métodos necesarios para lograr su propósito.

- ***Alta cohesión:*** La cohesión es la medida de la fuerza que une a las responsabilidades de una clase. Una clase con baja cohesión hace muchas cosas no relacionadas, o hace demasiado trabajo. Tales clases no son convenientes pues son difíciles de mantener, de reutilizar y de entender. Una clase con alta cohesión mejora la claridad y la facilidad de su uso, su mantenimiento se simplifica y es fácil de reutilizar.

  En el diseño de las clases se asignaron las responsabilidades de manera que la cohesión permaneciera alta. Por ejemplo, la clase *Sscan SAVUnix* es la encargada del escaneo de directorios, esta no realiza acciones que no estén relacionadas con su responsabilidad y para aliviar el trabajo de esta se le dio la tarea a la clase *OptionSscan* de gestionar y automatizar las opciones de escaneo.

- ***Bajo acoplamiento:*** Su principio es tener las clases lo menos ligadas posibles entre sí. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

  Se evidencia la parecencia del patrón bajo acoplamiento en la clases del diseño, debido a que existe solo la dependencia necesaria entre clases.

Conclusiones del capítulo

En este capítulo se definieron los aspectos relacionados con el análisis y el diseño del módulo para la administración de SAVUnix en Nova para Servidores. Se estableció como actor del sistema el administrador, además, se definieron los requisitos funcionales, los no funcionales y los casos de uso de la aplicación. Se describió cada caso de uso teniendo en cuenta el objetivo, complejidad, prioridad, precondiciones, poscondiciones, los flujos normales de eventos y los flujos alternos.

La arquitectura utilizada es Modelo Vista Controlador, la cual permite dividir la lógica de negocio del diseño, proporcionando mayor escalabilidad.

Se modeló el Diagrama de Clases del Diseño y los Diagramas de Secuencia, estos ofrecen una visión de las funcionalidades con las que debe contar el sistema en un lenguaje técnico.

Finalmente se exponen los patrones de diseño GRASP utilizados para asignar responsabilidades a las clases y lograr bajas dependencias entre clases, poco impacto ante un cambio y el incremento de la reutilización.

Capítulo 3: Implementación y realización de pruebas

Introducción

En el presente capítulo se documentan formalmente los resultados obtenidos durante la aplicación del análisis y diseño realizado en el capítulo anterior. Para ello se plasman elementos como el Diagrama de Componentes que muestra las dependencias lógicas entre componentes de software (fuentes, binarios, ejecutables) y el Diagrama de Despliegue que es utilizado para capturar los elementos de configuración del procesamiento y las conexiones entre ellos. También se realizó la implementación de la herramienta, para la cual se muestran el código fuente y los principales métodos. Asimismo, se le realizan las pruebas al módulo utilizando la técnica de caja negra.

3.1 Diagrama de Componentes

Un componente representa una unidad de código (fuente, binario o ejecutable) (Piñero, 2008) que permite mostrar las dependencias en tiempo de compilación y ejecución (Sommerville, 2005). Las instancias de componentes de software muestran unidades de software en tiempo de ejecución y generalmente ayudan a identificar sus dependencias y su localización en nodos. Pueden mostrar también que interfaces implementan y qué objetos contienen.

El paquete *Nova para Servidores* contiene la plataforma Zentyal que funcionará como un subsistema, este a su vez cuenta con el *Módulo Antivirus SAVUnix* del cual se establece comunicación con el *Sistema de Archivo* de Nova para Servidores para utilizar el *crontab* del sistema. Desde el *Módulo Antivirus SAVUnix* también se establece comunicación con el subsistema *Antivirus SAVUnix*, del cual se hace necesario manejar los ficheros de configuración, actualización y de reportes de cada uno de los antivirus (SAVUnix y SAVUnix Milter). El subsistema *Módulo Antivirus SAVUnix* cuenta con un conjunto de componentes, tales como *configMilter.pm*, *update.pm*, *licenseSAVUnix.pm*. Estos componentes contienen el código que dan respuesta a las funcionalidades necesarias en la aplicación y utilizan según su función los componentes del resto de los subsistemas. El siguiente diagrama muestra lo anteriormente descrito.

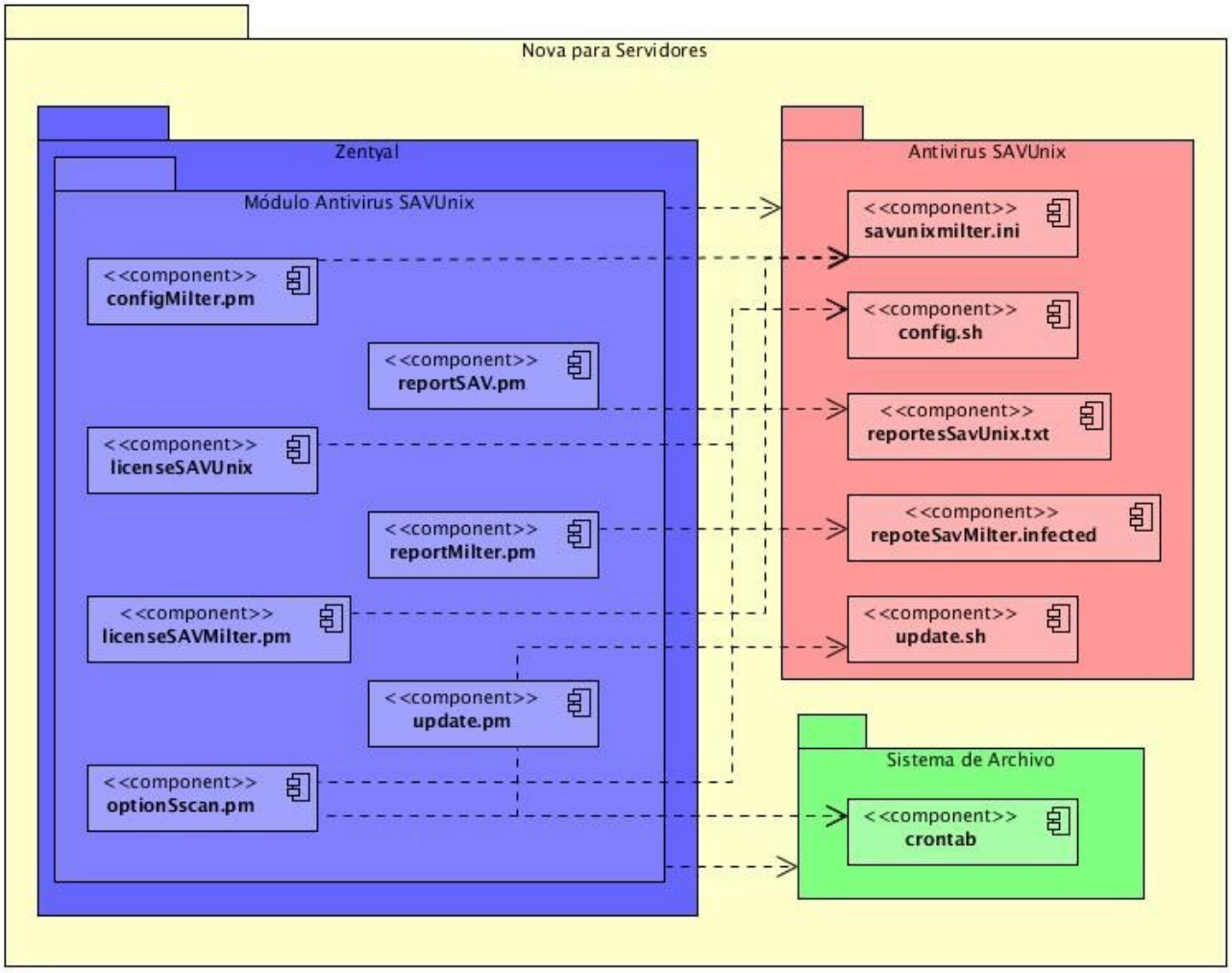


Figura 5: Diagrama de Componentes.

3.2 Diagrama de Despliegue

El Diagrama de Despliegue se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Los elementos que usa este tipo de diagrama son nodos, componentes y asociaciones (Sommerville, 2005).

Los elementos usados por este tipo de diagramas son:

- Nodos: los elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos.
- Dispositivos: los nodos son estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.

- Conectores: expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.

A continuación se muestra el Diagrama de Despliegue de la aplicación:

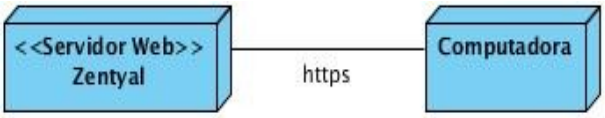


Figura 6: Diagrama de Despliegue.

3.3 Resultados obtenidos

Para obtener la aplicación deseada se tomaron como requerimientos básicos lograr la familiarización con el lenguaje de programación correspondiente (Perl) pues aunque los módulos están escritos precisamente en dicho lenguaje, resulta necesario el fácil entendimiento de las estructuras de datos y la sintaxis y la elección para la configuración del entorno de construcción de una máquina virtual con Zentyal o usando la propia máquina.

3.3.1 Usando una máquina virtual con Zentyal

Para esto se debe contar con los siguientes aspectos:

- Se puede utilizar VirtualBox con una instalación de Zentyal 3 o superior.
- Se puede establecer una conexión SSH a la máquina virtual para trabajar más cómodamente.
- Hay que tener todos los paquetes necesarios para la construcción que están disponibles en los repositorios o crear un repositorio en la propia máquina con los paquetes y dependencias necesarias.

3.3.2 Usando la propia máquina

Esta opción es menos recomendable porque cualquier acción que se haga, sin tener conocimientos sólidos puede afectar el sistema:

- Se tendrá que añadir los PPA de Zentyal con el fin de poder instalar las herramientas de construcción.
- Se necesita una distribución Linux (Debian, Ubuntu, Nova) para trabajar con el repositorio.
- Se puede construir los paquetes localmente y luego copiarlos a las máquinas virtuales utilizando

scp.

3.3.3 Implementación del módulo antivirus cubano para Zentyal

Zentyal está diseñado precisamente pensando en la extensibilidad y es relativamente sencillo crear nuevos módulos.

Cualquiera con conocimientos del lenguaje Perl puede aprovecharse de las facilidades que proporciona Zentyal para la creación de interfaces Web, y también beneficiarse de la integración con el resto de módulos y las demás características comunes de Zentyal.

El diseño de Zentyal es completamente orientado a objetos y hace uso del patrón Modelo-Vista-Controlador (MVC), de forma que el desarrollador sólo necesita definir qué características desea en su modelo de datos, y el resto será generado automáticamente por Zentyal. En las versiones anteriores de Zentyal cuando se quería crear un módulo se utilizaba la herramienta ***zmoddev***, una colección de scripts de conveniencia que ayudaba a crear la estructura y los archivos básicos para un módulo de Zentyal. En Zentyal 3.0 o superior esta herramienta no está disponible, es por ello que será necesario conocer cómo hacer un módulo desde cero.

3.3.4 Preparando el entorno de trabajo

Primero se procede a instalar las herramientas que facilitan el trabajo en el proceso de empaquetamiento.

- ***build-essential*** es un metapaquete que depende de libc6-dev, gcc, g++, make y dpkg-dev.
- ***devscripts*** contiene muchos scripts que facilitan considerablemente las tareas de un mantenedor de paquetes. Algunos de los más usados son debdiff, dch, debuild y debsing.
- ***debhelper*** son un conjunto de scripts que realizan tareas comunes de empaquetado.
- ***dh-make*** se utiliza para crear una plantilla para tu futuro paquete.
- ***diff*** y ***patch*** permite crear y aplicar parches, respectivamente.
- ***quilt*** administra los parches.
- ***gnupg*** es un completo y libre reemplazo para PGP usado para firmar digitalmente archivos (incluyendo paquetes).
- ***fakeroot*** simula ejecutar comandos con privilegios de root. Es útil para crear paquetes binarios

para uso personal.

- ***lintian*** disecciona paquetes Debian y avisa de errores y/o violaciones de la normativa.
- ***pbuilder*** crea un sistema chroot y construye un paquete en el mismo.
- ***zbuildtools*** permite construir e instalar módulos de Zentyal.

3.3.5 Estructura de un módulo de Zentyal

La estructura de un módulo de Zentyal es la siguiente:

Directorio con el nombre zentyal-nombreDelMódulo-versión. Este paquete contiene los siguientes directorios:

- debian: Contiene los ficheros para la debianización del paquete.
- schemas: Contiene un fichero (nombreDelModulo.yaml), en el cual se especifican las clases, dependencias, modelos y relaciones existentes entre los modelos. Por ejemplo:

```
class: 'EBox::AntiVirus'
depends:
    - network
models:
    - Config
    - Sscan
    - OptionSscan
    - Update
composites:
    General: [Scan, OptionsUpdate,Report]
    Scan: [Sscan, OptionSscan]
    OptionsUpdate: [Update]
    Report: [Sscan]
```

- src: Tiene dentro un directorio Ebox que contiene un fichero con el nombre del módulo y extensión .pm y un directorio con el mismo nombre. Este último posee dos directorios:

- Model: Implementa todas las vistas.
- Composite: Cómo mostrar las vistas.

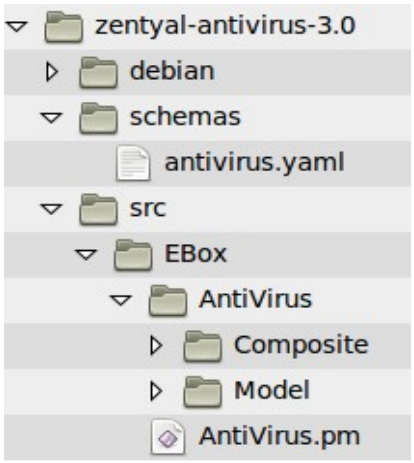


Figura 7: Estructura general del módulo Antivirus.

3.3.6 Construir e instalar el paquete

Por ahora no hay ninguna entrada en el menú para ver la vista asociada a este módulo, pero se puede fácilmente conocer la dirección URL al saber el nombre del modelo.

En el directorio (AntiVirus/src/EBox/AntiVirus/Model) se encuentran los modelos *Sscan*, *License*, *Update*, *OptionSscan*, *ShowReport,* entre otros.

En el método *_table* se definen los valores con los que va a contar la tabla con los datos de interés, con sus tipos de datos correspondientes y las acciones que se pueden hacer con los mismos. En la figura que se muestra a continuación se pueden observar los elementos por los que se compone la tabla correspondiente al modelo Sscan.

```
sub _table {
        my ($self) = @_;
        my @tableDesc =
        (
        new Ebox::Types::Text(
                                fieldName     => 'dir',
                                printableName => __('Directories to scan'),
```

```
                                        'optional'     => 0,
                                        editable       => 1,
                                        unique         => 1,
                                        ),
                );
                my $dataTable = {
                        'tableName'          => 'Sscan', #__PACKAGE__->nameFromClass(),
                        'printableTableName' => __('Listado de directorios a escanear'),
                        'pageTitle'          => __('Antivirus'),
                        'modelDomain'        => 'AntiVirus',
                        'defaultActions'     => ['editField', 'add','del','update','changeView'],
                        'tableDescription'   => \@tableDesc,
                        'messages'           => { (add => __('Adicionar nuevo directorio')),
                        (del =>__('Eliminar directorio')), (update => __('Actualizar directorio'))
                        },
                };
                return $dataTable;
        }
```

El método *_scan* del modelo OptionSscan permitirá escanear los directorios seleccionados.

```
        sub _scan {
                my ($self) = @_;
                my $obj = $self->parentModule()->model('Sscan');
                my @listDir = @{$obj->getRows()};
                $self->_automatic();
                EBox::Sudo::root('date +"%A %d of %B of %Y" > /opt/sav/doc/savunix.txt');
                foreach my $dir (@listDir) {
                        if ($self->desconValue()) {
```

```
                    EBox::Sudo::root ('echo "Analizado el directorio: "'."$dir".'">> /opt/sav/doc/savunix.txt');
                    EBox::Sudo::root ('sscan -i -d '."$dir".'">> /opt/sav/doc/savunix.txt');
                }
                else {
                    EBox::Sudo::root ('echo "Analizado el directorio: "'."$dir".'">> /opt/sav/doc/savunix.txt');
                    EBox::Sudo::root ('sscan -i '."$dir".'">> /opt/sav/doc/savunix.txt');
                }
            }
        }
```

El método *menu* como precisamente el nombre lo indica es un menú opcional que se declara en la clase principal, a diferencia de los métodos anteriores.

```
sub menu {
    my ($self, $root) = @_;
    my $folder = new EBox::Menu::Folder('name' => 'Antivirus',
                                        'text' => $self->printableName(),
                                        'separator' => 'Office',
                                        'order' => 580,
                                       );
    $folder->add(new EBox::Menu::Item('url' => 'AntiVirus/Composite/General',
                                      'text' => __('Antivirus de fichero')));
    $folder->add(new EBox::Menu::Item('url' => 'AntiVirus/View/OptionSscan',
                                      'text' => __('Antivirus de correo')));
      $root->add($folder);
}
```

Hasta el momento solamente se tiene una vista preliminar, sin embargo aún no está funcional. Para poder lograr el correcto funcionamiento del módulo hay que crear los métodos que harán posible la consistencia

de la utilización del servicio en cuestión.

Luego de tenerlo todo listo se corren los siguientes comandos: **cd zentyal-antivirus-3.0** para entrar al directorio donde se encuentra el código del módulo, y posteriormente se corre el comando: **debuild ; sudo dpkg -i ../zentyal-antivirus_3.0nova1.0_all.deb**, este permite construir el .deb y luego instalarlo.

3.4 Pruebas de software

Durante todo el ciclo de elaboración del software es preciso velar, controlar y garantizar su correcta calidad, haciendo posible el cumplimiento de los requerimientos que precisamente satisfacen las necesidades del cliente. Este aspecto debe estar presente de forma paralela desde la concepción del producto hasta la fase de producción del mismo. Para verificar que una vez culminado el desarrollo del producto este cumpla con los objetivos trazados se recurre a la realización de las pruebas de software.

Entre algunas de las técnicas que se llevan a cabo para el proceso de prueba se encuentran las técnicas de caja negra y de caja blanca (Sommerville, 2005).

- ***Pruebas de caja negra***

Prueba de caja negra es aquel elemento que se estudia desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce, sin tener en cuenta su funcionamiento interno. En otras palabras, de una caja negra solamente interesará su forma de interactuar con el medio que le rodea, entendiendo qué es lo que hace, pero sin dar importancia a cómo lo hace. Por tanto, de una caja negra deben estar muy bien definidas sus entradas y salidas, es decir, su interfaz; en cambio, no se precisa definir ni conocer los detalles internos de su funcionamiento.

Los métodos de prueba de la caja negra se centran en los requisitos funcionales del producto e intentan encontrar errores de las siguientes categorías: (Sommerville, 2005)

1. Funciones incorrectas o ausentes.
2. Errores de interfaz.
3. Errores en estructuras de datos o en acceso a bases de datos externas
4. Errores de rendimiento.
5. Errores de inicialización y terminación.

A continuación se muestran los Casos de Pruebas. Para probar las variables se definieron los valores Válido (V) e Inválido (I).

CP1. Administrar el módulo Antivirus SAVUnix

Tabla 15: Caso de Prueba “Administrar el módulo Antivirus SAVUnix”.

Caso de Prueba	Administrar el módulo Antivirus SAVUnix.		
Condiciones de Ejecución	El módulo antivirus SAVUnix debe existir en el repositorio.		
Descripción General	El administrador selecciona la opción *Gestión de software/Componentes de Zentyal* donde se muestra las pestañas con las opciones de instalación y desinstalación.		
Verificación	Se muestra en el separador Office del menú la opción Antivirus SAVUnix que incluye a los submenús Configuración, Escanear directorio y Reportes.		
Sección 1: “Instalar el módulo Antivirus SAVUnix”			
Descripción	Se instala el antivirus SAVUnix.		
Flujo Central	1. El administrador selecciona la pestaña *Instalar*. 2. El sistema muestra un listado con todos los módulos que pueden ser instalados. 3. En la sección *Seleccionar* el administrador marca el cuadro correspondiente al módulo *Antivirus SAVUnix* y presiona el botón *Instalar*. 4. El sistema muestra un mensaje de confirmación. 5. El administrador presiona el botón *Aceptar*. 6. Se instala el módulo y se muestra un mensaje de éxito.		
Escenarios	**Variables** Módulo	**Respuesta del sistema**	**Resultado de la prueba**
El módulo no está instalado	Antivirus SAVUnix	Instala el módulo Antivirus SAVUnix y muestra un mensaje de éxito.	Satisfactorio
El módulo ya está instalado	Antivirus SAVUnix	No se muestra el módulo en el listado de módulos a instalar.	Satisfactorio
Sección 2: “Desinstalar el módulo Antivirus SAVUnix”			
Descripción	Se desinstala el antivirus SAVUnix.		
Flujo Central	1. El administrador selecciona la pestaña *Borrar*. 2. El sistema muestra un listado con todos los módulos que pueden ser desinstalados. 3. En la sección *Seleccionar* el administrador marca el cuadro correspondiente al módulo *Antivirus SAVUnix* y presiona el botón *Borrar*. 4. El sistema muestra un mensaje de confirmación. 5. El administrador presiona el botón *Aceptar*. 6. Se desinstala el módulo y se muestra un mensaje de éxito.		
Escenarios	**Variables**	**Respuesta del sistema**	**Resultado de la prueba**

	Módulo		
El módulo está instalado	Antivirus SAVUnix	Desinstala el módulo Antivirus SAVUnix y muestra un mensaje de éxito.	Satisfactorio
El módulo está desinstalado	Antivirus SAVUnix	No se muestra el módulo en el listado de módulos a desinstalar.	Satisfactorio

CP2. Actualizar antivirus SAVUnix

Tabla 16: Caso de Prueba “Actualizar antivirus SAVUnix”.

Caso de Prueba	Actualizar antivirus SAVUnix.
Condiciones de Ejecución	El administrador debe haber instalado el antivirus y haber cargado la licencia del software.
Descripción General	El administrador selecciona en la opción *Antivirus SAVUnix/Configuración* la pestaña *Antivirus de fichero*, luego podrá configurar las opciones de actualización.
Verificación	Chequeando en los registros del antivirus (/var/log/sav/savdeamon.log) de que no haya ocurrido un error.
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de Fichero*. 2. El sistema muestra la interfaz de Actualización del antivirus. 3. El administrador configura las opciones de actualización y presiona el botón *Actualizar*. 4. Se guarda la nueva configuración.
Variables	Servidor web (SW), IP, Puerto (P), Usuario (U), Contraseña (C), Directorio de actualización (DA).

Escenarios	**Variables**						**Respuesta del sistema**	**Resultado de la prueba**
	SW	IP	P	U	C	DA		
Introduce todos los datos correctamente	V	V	V	V	V	V	Actualiza el antivirus SAVUnix.	Satisfactorio
Introduce algún dato incorrecto.	I	V	V	V	V	V	Muestra un mensaje especificando el campo que tuvo el error.	Satisfactorio
	V	I	V	V	V	V		
	V	V	I	V	V	V		
	V	V	V	I	V	V		
	V	V	V	V	I	V		
	V	V	V	V	V	I		

CP3. Cargar licencia del antivirus SAVUnix

Tabla 17: Caso de Prueba “Cargar licencia del antivirus SAVUnix”.

Caso de Prueba	Cargar licencia del antivirus SAVUnix.

Condiciones de Ejecución	El administrador debe haber instalado el antivirus.
Descripción General	El administrador selecciona en la opción *Antivirus SAVUnix/Configuración* la pestaña *Antivirus de fichero*, luego carga la licencia del antivirus SAVUnix.
Verificación	Si se cargó correctamente la licencia, se habilita la opción de Actualizar SAVUnix.
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de fichero*. 2. El sistema muestra un campo que permite *Examinar* los directorios locales. *3. El administrador* Examina los directorios locales, selecciona la licencia y presiona el botón *Cargar Licencia*. 4. Se carga la licencia del antivirus.
Variables	Ruta de la licencia (RL), Licencia (L).

Escenarios	**Variables**		**Respuesta del sistema**	**Resultado de la prueba**
	RL	L		
Introduce todos los datos correctamente.	V	V	Carga la licencia del antivirus SAVUnix y muestra mensaje de éxito	Satisfactorio
Introduce algún dato incorrecto.	I	V	Muestra un mensaje especificando el tipo error cometido.	Satisfactorio
	V	I		

CP4. Cargar licencia del antivirus SAVUnix Milter

Tabla 18: Caso de Prueba “Cargar licencia antivirus SAVUnix Milter”.

Caso de Prueba	Cargar licencia del antivirus SAVUnix Milter.
Condiciones de Ejecución	El administrador debe haber instalado el antivirus.
Descripción General	El administrador selecciona en la opción *Antivirus SAVUnix/Configuración* la pestaña *Antivirus de correo*, luego carga la licencia del antivirus SAVUnix Milter.
Verificación	Si se cargó correctamente la licencia entonces se habilita la opción de Configurar SAVUnix Milter.
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de correo*. 2. El sistema muestra un campo que permite *Examinar* los directorios locales para seleccionar la licencia. *3. El administrador* Examina los directorios locales, selecciona la licencia y presiona el botón *Cargar Licencia*. 4. Se carga la licencia del antivirus SAVUnix Milter.
Variables	Ruta de la licencia (RL), Licencia (L).

Escenarios	Variables		Respuesta del sistema	Resultado de la prueba
	RL	L		
Introduce todos los datos correctamente.	V	V	Carga la licencia del antivirus SAVUnix y muestra mensaje de éxito	Satisfactorio
Introduce algún dato incorrecto.	I	V	Muestra un mensaje especificando el tipo error cometido.	Satisfactorio
	V	I		

CP5. Mostrar reportes SAVUnix

Tabla 19: Caso de Prueba “Mostrar reportes SAVUnix”.

Caso de Prueba	Mostrar reportes SAVUnix.
Condiciones de Ejecución	El administrador debe haber configurado el antivirus SAVUnix.
Descripción General	El administrador selecciona la opción *Antivirus SAVUnix/Reportes* y selecciona la pestaña *Antivirus de fichero*, permitiendo esta visualizar los reportes dados por el antivirus durante el último escaneo.
Verificación	Se visualizan correctamente los reportes generados por el antivirus SAVUnix.
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de fichero.* 2. El sistema muestra los reportes dados por el antivirus SAVUnix durante el último escaneo realizado.
Respuesta del Sistema	Muestra los reportes del antivirus SAVUnix.
Resultado de la prueba	Satisfactorio

CP6. Mostrar reportes SAVUnix Milter

Tabla 20: Caso de Prueba “Mostrar reportes SAVUnix Milter”.

Caso de Prueba	Mostrar reportes SAVUnix Milter.
Condiciones de Ejecución	El administrador debe haber configurado el antivirus SAVUnix Milter.
Descripción General	El administrador selecciona la opción *Antivirus SAVUnix/Reportes* y selecciona la pestaña *Antivirus de correo*, permitiendo esta visualizar los reportes dados por el antivirus SAVUnix Milter durante el último escaneo.
Verificación	Se visualizan correctamente los reportes generados por el antivirus SAVUnix Milter.
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de correo.* 2. El sistema muestra los reportes dados por el antivirus SAVUnix Milter durante el último escaneo realizado.
Respuesta del Sistema	Muestra los reportes del antivirus SAVUnix Milter.

Resultado de la prueba	Satisfactorio

CP7. Configurar antivirus SAVUnix Milter

Tabla 21: Caso de Prueba “Configurar antivirus SAVUnix Milter”.

Caso de Prueba	Configurar antivirus SAVUnix Milter.		
Condiciones de Ejecución	El administrador debe haber instalado el antivirus SAVUnix.		
Descripción General	El administrador selecciona en la opción *Antivirus SAVUnix/Configuración* la pestaña *Antivirus de correo*, permitiendo esta configurar las opciones del antivirus SAVUnix Milter.		
Verificación	Chequeando en los registros del antivirus SAVUnix Milter (/var/log/savunixmilter/savunixmilter.log) de que no haya ocurrido un error.		
Flujo Central	1. El administrador selecciona la pestaña *Antivirus de correo.* 2. El sistema muestra la interfaz de configuración del antivirus SAVUnix Milter. 3. El administrador entra los datos de configuración y presiona el botón *Guardar*. 4. El sistema guarda la nueva configuración.		
Escenarios	**Variables**	**Respuesta del sistema**	**Resultado de la prueba**
	Correo del administrador		
Introduce todos los datos correctamente.	V	Configura el antivirus SAVUnix Milter.	Satisfactorio
Introduce una dirección de correo inválida.	I	Muestra un mensaje de error de correo.	Satisfactorio

CP8. Escanear directorio

Tabla 22: Caso de Prueba “Escanear directorio”.

Caso de Prueba	Escanear directorio.
Condiciones de Ejecución	El administrador debe haber configurado el antivirus.
Descripción General	El administrador selecciona la opción *Antivirus SAVUnix/Escanear directorio*. El antivirus realizará el escaneo teniendo en cuenta la personalización realizada.
Verificación	Se muestran cambios en los reportes generados del escaneo.
Sección 1: “Adicionar directorio”	
Descripción	Se adiciona a la lista un nuevo directorio.
Flujo Central	1. El administrador selecciona la opción *Añadir nuevo/a*. 2. El sistema muestra un campo para introducir el directorio.

	3. El administrador llena el campo con la ruta del directorio deseado y presiona el botón *Añadir*. 4. Se agrega la dirección al listado de directorios.		
Escenarios	**Variables** Ruta	**Respuesta del sistema**	**Resultado de la prueba**
Introduce una ruta correcta.	V	Se adiciona el directorio a la lista de directorios a escanear.	Satisfactorio
Introduce una ruta incorrecta.	I	Muestra un mensaje de error de ruta.	Satisfactorio
Sección 2: “Eliminar directorio”			
Descripción	Se elimina un directorio de la lista de directorios a escanear.		
Flujo Central	1. El administrador selecciona en la columna *Acción* del listado de directorios la opción *Borrar* correspondiente al directorio que desea eliminar. 2. Se elimina el directorio del listado.		
Respuesta del Sistema	Elimina el directorio seleccionado de la lista de directorios a escanear.		
Resultado de la prueba	Satisfactorio		
Sección 3: “Modificar directorio ”			
Descripción	Se modifica un directorio de la lista de directorios a escanear.		
Flujo Central	1. El administrador selecciona en la columna *Acción* del listado de directorios la opción *Editar* correspondiente al directorio que desea modificar. 2. El sistema muestra una interfaz para modificar el directorio. 3. El administrador introduce los nuevos datos y presiona el botón *Cambiar*. 4. Se modifica el directorio.		
Respuesta del Sistema	Modifica el directorio seleccionado de la lista de directorios a escanear.		
Resultado de la prueba	Satisfactorio		
Sección 4: “Revisar directorio”			
Descripción	Se escanean los directorios de la lista para determinar cuáles están infectados.		
Flujo Central	3. El administrador presiona el botón *Escanear*. 2. Se escanean los directorios seleccionados en busca de malware.		
Respuesta del Sistema	Escanea los directorios seleccionados en búsqueda de malware.		
Resultado de la prueba	Satisfactorio		

CP9. Desinfectar directorio

Tabla 23: Caso de Prueba “Desinfectar directorio”.

Caso de Prueba	Desinfectar directorio.
Condiciones de Ejecución	El administrador debe haber seleccionado directorios para escanear.
Descripción General	Se realiza la desinfección de los ficheros contaminados.
Verificación	Se muestran cambios en los reportes generados del escaneo.
Flujo Central	1. El administrador selecciona la opción *Descontaminar.* 2. Se desinfectan los ficheros contaminados.
Respuesta del Sistema	Desinfecta los directorios escaneados.
Resultado de la prueba	Satisfactorio

CP10. Personalizar opciones de escaneo

Tabla 24: Caso de Prueba “Personalizar opciones de escaneo”.

Caso de Prueba	Personalizar opciones de escaneo.
Condiciones de Ejecución	El administrador debe haber seleccionado directorios para escanear.
Descripción General	El administrador personaliza las opciones de escaneo.
Verificación	Se chequea en */etc/crontab* que la personalización haya sido guardada.
Flujo Central	1. El administrador selecciona las opciones de escaneo. 2. Se guardan la nueva personalización en el fichero */etc/crontab* del sistema.
Respuesta del Sistema	Guarda la personalización realizada.
Resultado de la prueba	Satisfactorio

El módulo desarrollado fue evaluado como parte de la Distribución Cubana de GNU/Linux Nova para Servidores por el Centro Nacional de Calidad de Software (Calisoft), obteniendo resultados satisfactorios. Para más detalles ver Anexo 12.

Conclusiones del capítulo

Mediante el Diagrama de Componentes se representó la división en componentes y las dependencias entre estos del módulo para la administración de SAVUnix en Zentyal. Por otro lado el Diagrama de Despliegue permitió representar las dependencias entre nodos físicos en los que funcionará la aplicación.

Para el desarrollo del módulo fue necesario lograr la familiarización con el lenguaje de programación Perl,

con las herramientas que facilitan el trabajo en el proceso de empaquetamiento y con la estructura general de los módulos en Zentyal.

Una vez obtenido el módulo se le realizaron pruebas de caja negra, utilizando como guía un conjunto de condiciones o variables establecidas en los casos de pruebas correspondientes a cada uno de los casos de usos, obteniendo resultados completamente satisfactorios.

Conclusiones

Al concluir el presente trabajo:

- El estudio de la bibliografía consultada permitió tener mayor claridad para realizar el análisis sobre las principales características de los antivirus que cumplen con el artículo 50 de la Resolución 127 del 2007 del MIC.
- Se demostró la necesidad de tener en cuenta la soberanía tecnológica y la seguridad del país a la hora de tomar la decisión de utilizar el antivirus SAVUnix.
- Se describió la solución propuesta mediante diagramas y tablas para una mejor comprensión del módulo, teniendo en cuenta la arquitectura y los patrones de diseños utilizados.
- Se documentó el proceso de construcción de un módulo para la plataforma de desarrollo Zentyal, para las nuevas generaciones de desarrolladores del proyecto Nova.
- Se obtuvo un módulo de Zentyal que provee una interfaz amigable e intuitiva que permite la administración del antivirus SAVUnix.
- Se le realizaron pruebas de caja negra, utilizando como guía un conjunto de condiciones o variables establecidas en los casos de pruebas definidos, obteniendo resultados satisfactorios.

El resultado de esta investigación le brinda valor agregado a la distribución Nova para Servidores, ya que además de cumplir los objetivos trazados, se incrementó la soberanía tecnológica y la seguridad de Nova para Servidores y se estableció lazos de colaboración con la entidad cubana Segurmática y con la comunidad internacional de desarrolladores de Zentyal, posibilitando mayor visibilidad y prestigio para el proyecto productivo Nova.

Recomendaciones

El análisis, diseño e implementación del módulo se efectuó de forma satisfactoria, sin embargo, con la finalidad de alcanzar mejores resultados se recomienda:

- Incluir el antivirus SAVUnix Dansguardian para la desinfección de proxy.

Bibliografía

1. Álvarez, G. (2010). *Herramientas y Técnicas de Hacking y Cracking y cómo protegerse ante ellas*. Madrid, España.
2. Panda Labs. (2011). *Informe anual Panda Labs*.
3. Beekmans, G. (2007). *Linux from Scratch*.
4. Bidot, J. (2012). *Programas malignos y otras amenazas a la Seguridad Informática. Acciones para su enfrentamiento*. La Habana, Cuba.
5. CESOL. (2012). Catálogo de Productos y Servicios. La Habana, Cuba.
6. Corletti, A. (2011). *Seguridad por Niveles*. Madrid, España.
7. Dirección de Redes y Seguridad Informática UCI. (2012). *Políticas de Seguridad de la UCI*. La Habana, Cuba.
8. Elizalde, R y Manzaneda, J. (2011). *De la Ciberguerra a la Ciberdefensa activa*. La Habana, Cuba.
9. ESET. (2010). *Tipos de malware y otras amenazas informáticas*.
10. ESET Global LLC. (2010). *Consejos de Seguridad*.
11. Fernández, E. (2011). *Sistema para el Análisis de Sensibilidad en la plataforma BioSyS*.
12. Geany. (2012). [Disponible en: http:www.geany.org/]
13. Gostev, A. (2012). *The Flame: Questions and Answers*.
14. Hernández, R y Coello, S. (2011). *El proceso de investigación científica*. La Habana, Cuba.
15. Herrera, A. (2009). *Seguridad local básica en Gnu/Linux*. La Habana, Cuba.
16. Herrera, A; Hurtado, M; Goñi, A; Meneses, A; Hernández, D; Machín, J y Miranda, R. (2009). *Nova, Distribución Cubana de GNU/Linux*. La Habana, Cuba.
17. Kaspersky Lab España. (2012). [Disponible en: http://www.kaspersky.com/sp/]
18. Keizer, G. (2010). *Is Stuxnet the best malware ever?*
19. Larman, C. (2004). *UML y Patrones*.

20. Matrosov, A; Rodionov, E; Harley, D y Malcho, J. (2012). *Stuxnet under the microscope*.

21. McMillan, R. (2010). *Siemens: Stuxnet worm hit industrial systems*.

22. Merelo, J. (2009). *Tutorial de Perl*.

23. MIC. (2007). *Resolución 127*. La Habana, Cuba.

24. MIC. (2010). *Informatización de la sociedad*. [Disponible en: http://www.mic.gov.cu/]

25. Microsoft TechNet. (2009). *Defining Malware: FQA*.

26. Montilva, J. (2003). *Desarrollo de Software Basado en Componentes*.

27. Open UP. (2012). [Disponible en:http://epf.eclipse.org/wikis/openup/]

28. Panda Labs. (2011). *Informe anual Panda Labs Resumen 2001*.

29. PC World. (2010). *Zombie PC: Silent, Growing Thera*t.

30. Perl. (2012). *Perl Programming Documentation*. [Disponible en: http://perldoc.perl.org/]

31. Pfleeger, C y Lawrence, S. (2006). *Security in Computing*.

32. Pierra, A. (2011). *Nova, Distribución Cubana de GNU/Linux. Reestructuración estratégica de su proceso de desarrollo*. La Habana, Cuba.

33. Pierra, A. (2011). *Un Acercamiento a: Nova, Distribución Cubana de GNU/Linux*. La Habana, Cuba.

34. Piñero, P. (2008). *Introducción a los modelos de desarrollo de software*.

35. Pressman. (2001). *Ingeniería de Software. Un enfoque práctico*. Madrid, España.

36. Ray, Y. (2005). *The short life and hard times of a Linux virus*.

37. Rosales, E. (2012). *Módulo para la administración de NAS en Nova para Servidores*. La Habana, Cuba.

38. Sagasti, I y Baz, I. (2010). *Curso Linux: Administración de Sistema y Servicios*. [Disponible en: http://www.irontec.com]

39. Segurmática. (2012). [Disponible en: http://www.segurmatica.cu/]

40. Symantec. (2012). [Disponible en: http://www.symantec.com]

41. Sommerville, I. (2005). *Ingeniería de Software*. Madrid, España.

42. Stallman, R. (2004). *Software Libre para una Sociedad Libre*. [Disponible en: http://www.gnu.org/philosophy/fsfs/free_software.es.pdf]

43. Tori, C. (2008). *Hacking Ético.* Buenos Aires, Argentina.

44. Visual Paradigm. (2012). Visual Paradigm for UML. [Disponible en: http://www.visual-paradigm.com/]

45. Zentyal. (2010). *Zentyal: servidor Linux para pymes*. [Disponible en: http://doc.zentyal.org/es/presentation.html]

46. Zentyal. (2011). Primeros pasos con Zentyal. [Disponible en: http://doc.zentyal.org/es/firststeps.html]

47. Zentyal. (2012). *Zentyal 3.0 Documentación Oficial*.

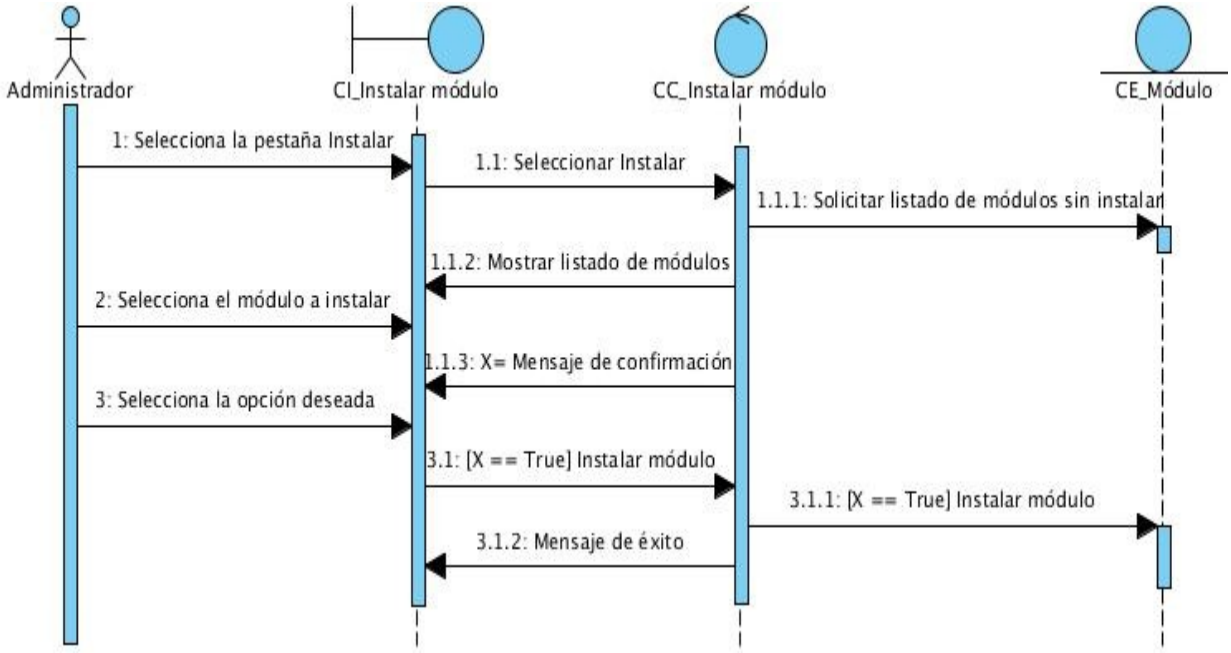
Glosario de términos

- Agente de Transferencia de Correo (MTA): Es uno de los programas que ejecutan los servidores de correo, y tiene como fin transferir un conjunto de datos de una computadora a otra.
- Ciberguerra: Guerra en la que intervienen Estados, ejércitos y servicios secretos en una nueva ecología, la de las redes digitales, que mediatiza los conflictos y ha aportado sus propios instrumentos, pero no ha modificado sustancialmente la conciencia del hombre.
- Escalabilidad: Indica la habilidad de un sistema para adaptarse sin perder calidad, o bien manejar el crecimiento continuo, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos.
- Malware: También llamado badware, código maligno, software malicioso o software malintencionado, es un tipo de software que tiene como objetivo infiltrarse o dañar una computadora y/o sistema de información sin el consentimiento de su propietario.
- Metodología: Define quién hace qué, cómo y cuándo.
- Milter: Es una extensión ampliamente usada por los Agente de Transferencia de Correo para el procesamiento de correos. Permite al filtro examinar y modificar el contenido del mensaje y la metainformación del mismo durante la transacción SMTP.
- Protocolo para la transferencia simple de correo electrónico (SMTP): Es un protocolo basado en texto, utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos.
- Stakeholders: Trabajadores, organizaciones sociales, accionistas y proveedores, entre muchos otros actores clave que se ven afectados por las decisiones de una empresa.

Anexos

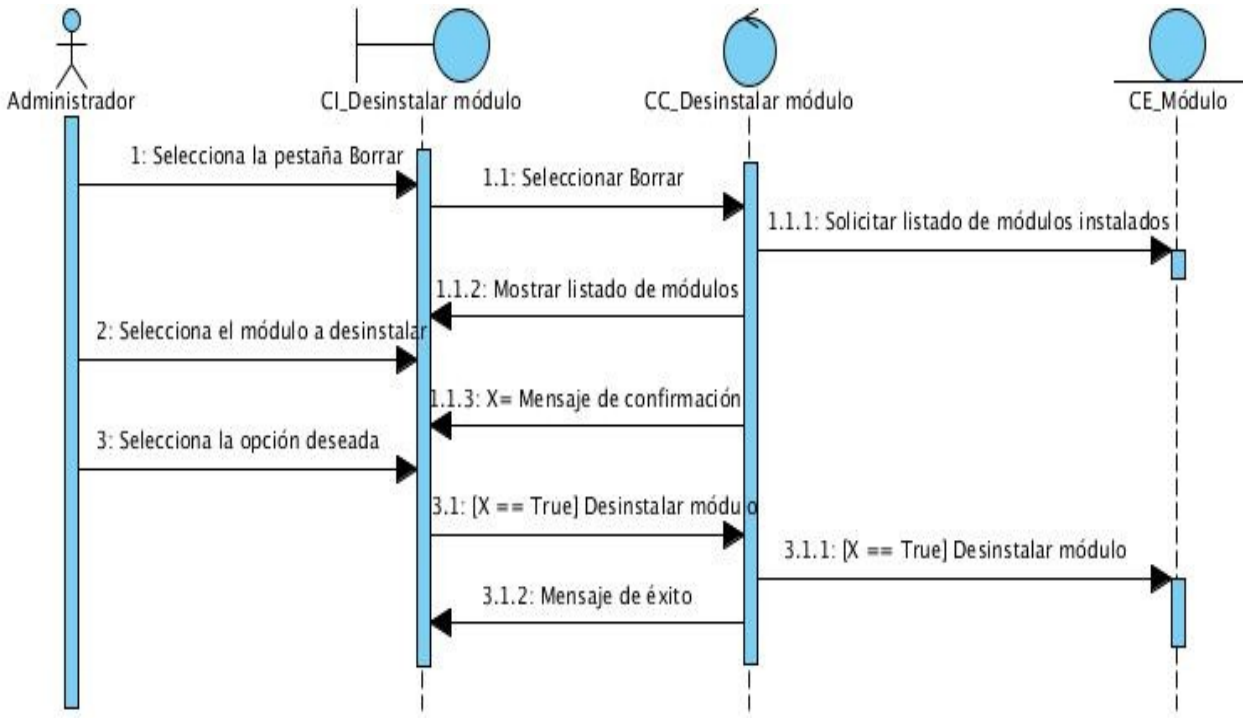
Anexo 1: Instalar el módulo Antivirus SAVUnix.

Anexo 2: Diagrama de secuencia del RF Instalar el módulo Antivirus SAVUnix.



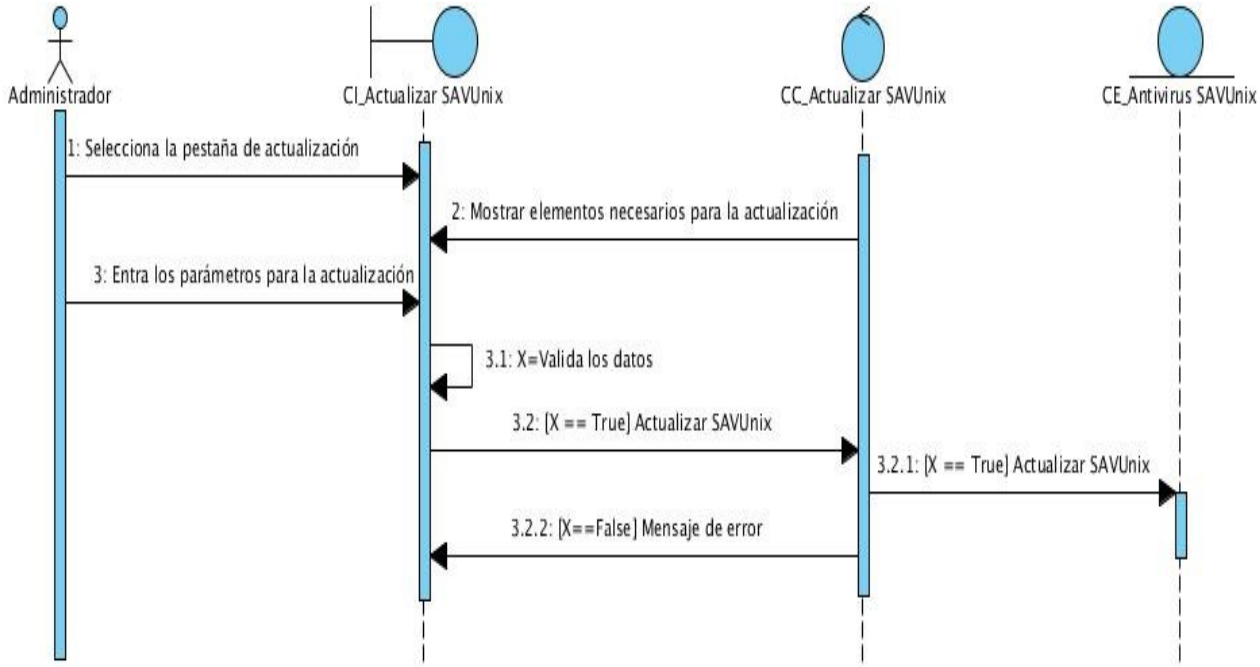
Anexo 3: Desinstalar el módulo Antivirus SAVUnix.

Anexo 4: Diagrama de Secuencia del RF Desinstalar el módulo Antivirus SAVUnix.

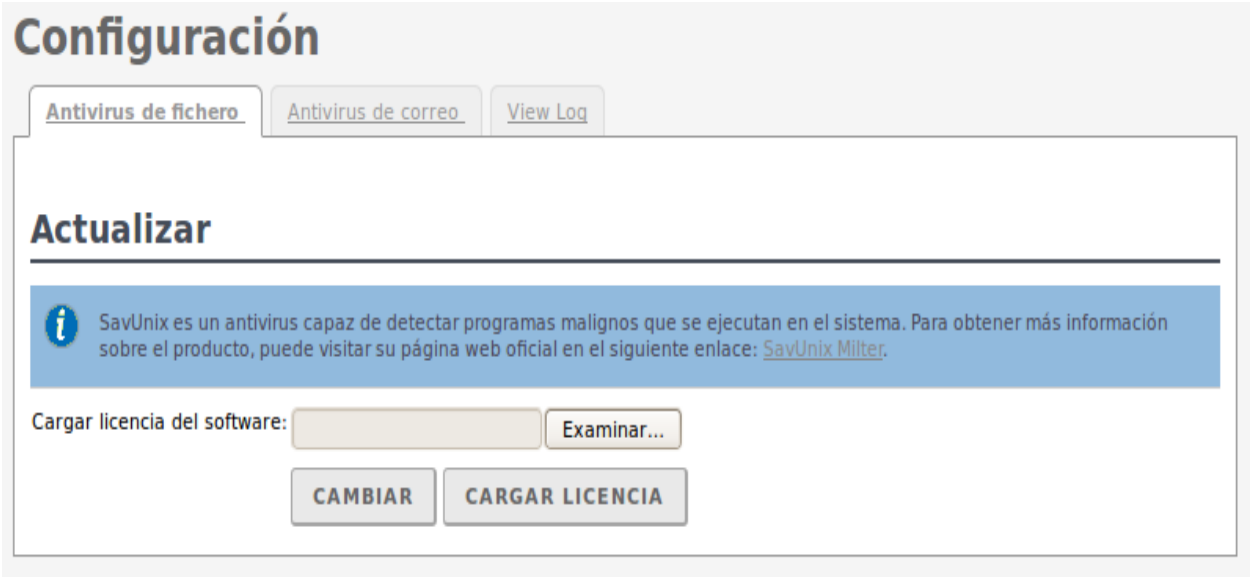


Anexo 5: Actualizar antivirus SAVUnix.

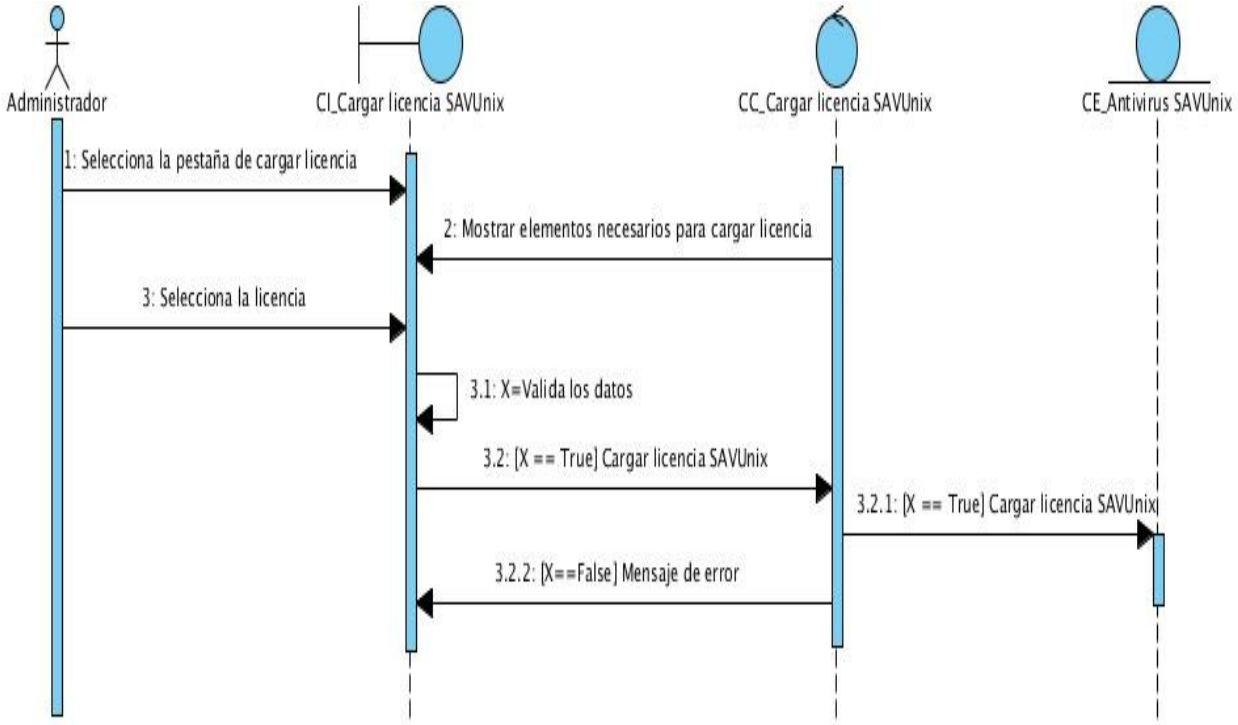
Anexo 6: Diagrama de Secuencia del RF Actualizar antivirus SAVUnix.



Anexo 7: Cargar licencia SAVUnix.

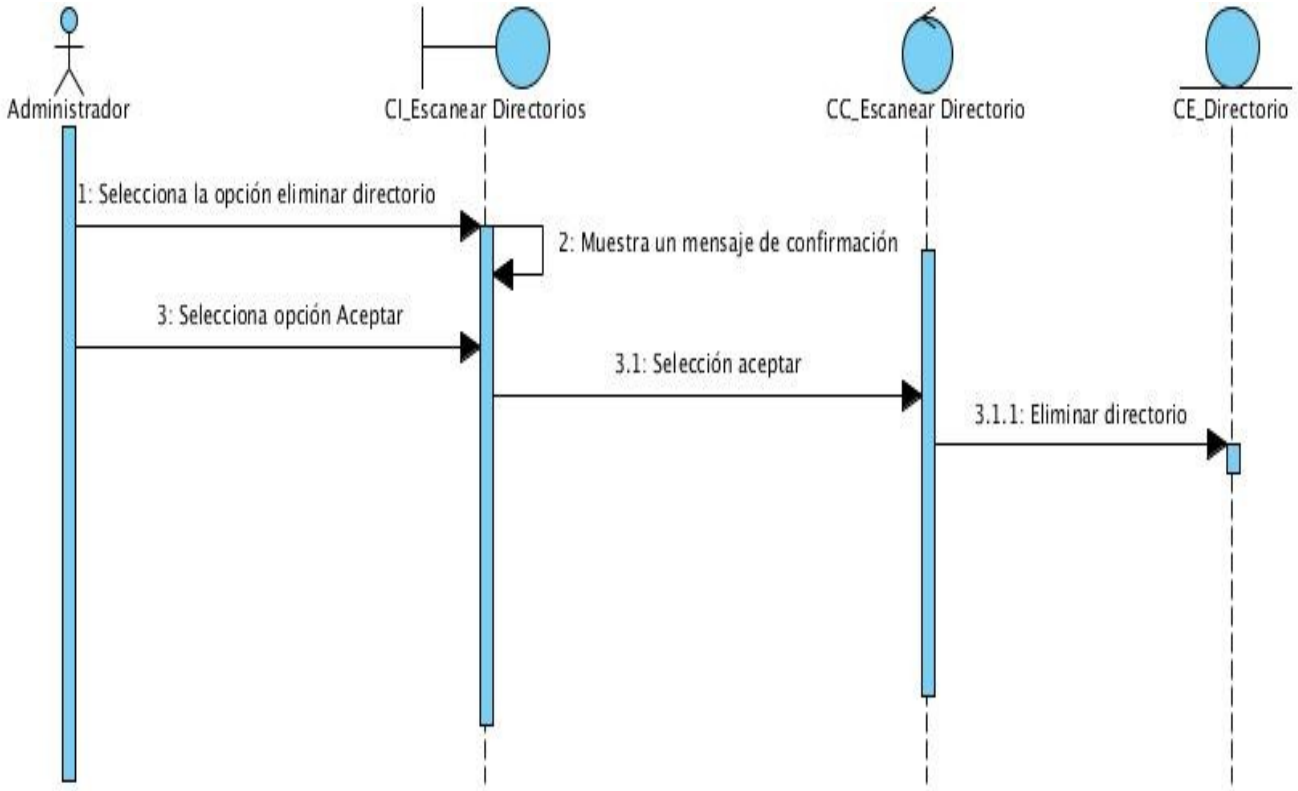


Anexo 8: Diagrama de Secuencia del RF Cargar licencia SAVUnix.

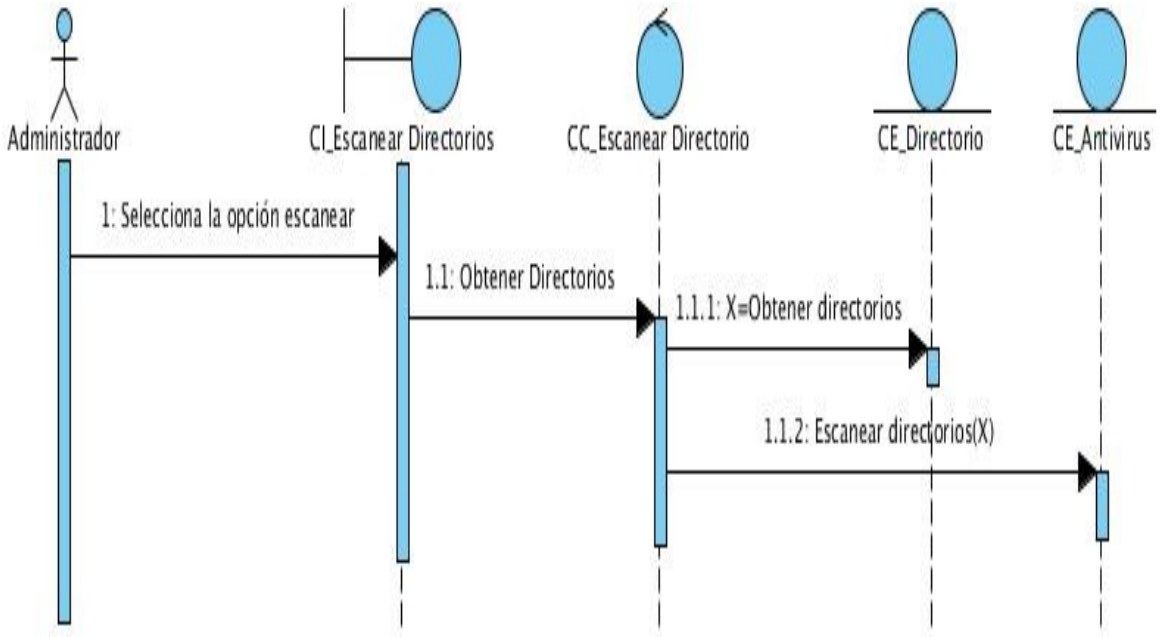


Anexo 9: Escanear directorios.

Anexo 10: Diagrama de Secuencia del RF Eliminar Directorio.



Anexo 11: Diagrama de Secuencia del RF Revisar Directorio.



Anexo 12: Acta de liberación del producto Nova para Servidores 2013.

ACTA DE LIBERACIÓN
EVALUACIÓN DE PRODUCTOS DE SOFTWARE

Fecha de emisión del acta: 11/04/2013

Emitida a favor del Proyecto: Nova Servidores 2013

Nombre del producto: Nova Servidores 4.0

Datos del producto:

Artefacto	Versión	Tipos de pruebas realizados	Cantidad de iteraciones	Estado Final	Fecha de Liberación
Nova Servidores 2013	4.0	Funcionales	3 iteraciones	1 NC	10/04/2013

Anexo:
A continuación se listan las No Conformidades (NC) que fueron declaradas como No Procede, pero que el equipo de pruebas considera debe ser evaluada la solución de las mismas, por el proyecto.

Descripción de la NC	Nivel de importancia de la NC	Respuesta del equipo de desarrollo
Internacionalización del panel.	No Significativa.	Se intentó cambiar el idioma para cada pestaña del panel pero esto provoca mal funcionamiento de los módulos, ya que estas categorías están insertadas sin internacionalización dentro del código y guardan relación estrecha con la base de datos de la aplicación.

Yailin Fundora Carrasco

Ing. Yailin Fundora Carrasco
Coordinador de la Evaluación de Productos

Alionuska Velázquez Cintra

Ing. Alionuska Velázquez Cintra
Jefa del Departamento de Evaluación de Productos

Ing. Abel Alfonso Fírvida Donéstvez
Líder del Proyecto
Nova Servidores 2013