Universidad de las Ciencias Informáticas

Módulo para el cifrado de los datos informáticos almacenados en NovaNAS

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

Autora:

Shadet Asnay Ariosa Aquia

Tutores:

MSc. Yasiel Pérez Villazón

MSc. Ruth Yurina Vega Cutiño

La Habana, Junio del 2019

Dedicatoria

*A mi **madre** por apoyarme en cada decisión que he tomado en la vida, guiarme en el camino a ser una mejor persona y brindarme todo su amor.*

*A mi **familia** por alentarme en cada momento y creer en mí para la realización de este sueño.*

A ustedes va dedicado con todo el cariño y el amor del mundo.

"Hay, en verdad, dos cosas diferentes: saber y creer que se sabe. La ciencia consiste en saber; en creer que se sabe está la ignorancia".

-Hipócrates.

Agradecimientos

Durante estos años de la carrera para convertirme en ingeniera en ciencias informáticas he cruzado mi camino con personas que de una forma u otra han influido en mi cambio hacia la persona que soy hoy. Por eso quiero agradecer:

A mi madre por brindarme su apoyo incondicionalmente en cada paso que doy en la vida, hoy cumplo la promesa que le hice, haciendo realidad el sueño de las dos.

A mi familia por la fuerza que me dan para seguir adelante en el camino al éxito al que hoy doy un paso más.

A todos mis compañeros de aula con los que llegue hasta aquí, con alegrías, llantos, discusiones, bromas y que a pesar de todo fueron como una familia aquí en la UCI,

A los amigos que tuvieron la paciencia, la comprensión y el cariño para apoyarme en los peores momentos y siempre con las risas hacer más fáciles los días complicados. A mi grupo de estudios, hermanos, compañeros, más que amigos que me cuidaron y tuvieron toda la paciencia del mundo conmigo pero también me dieron muchos motivos para reír.

A cada uno de los profesores que me enseñaron más que solo materias, me enseñaron de la vida, y sin saberlo se convirtieron en un apoyo importante para mí sobre todo estos últimos dos años en los que a veces necesite un consejo, y sin importar la hora ni el momento siempre estuvieron para ayudarme. Como un buen padre que ayuda a su hijo.

A mis tutores que fueron un apoyo súper importante en estos últimos momentos, y que me ayudaron en los momentos más tensos, por su apoyo y dedicación los considero personas extraordinarias.

Declaración Jurada de Autoría

Declaro por este medio que yo Shadet Asnay Ariosa Aquia, con carné de identidad 96051108116, ser la única autora de este trabajo de diploma titulado “Módulo para el cifrado de los datos informáticos almacenados en NovaNAS” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmo la presente a los ___ días del mes de _____ del año 2019.

_____

Firma del autor

Shadet Asnay Ariosa Aquia

_____

Firma del tutor

MSc. Yasiel Perez Villazón

_____

Firma del tutor

MSc. Ruth Yurina Vega Cutiño

RESUMEN

Como parte del proceso de desarrollo de software y la migración nacional a software libre surge NovaNAS, una aplicación web desarrollada en el Centro de Software Libre de la Universidad de las Ciencias Informáticas para la administración de un servidor de almacenamiento en la red. En dicha aplicación existen problemas de seguridad relacionados con la confidencialidad de los datos informáticos que en este se almacenan. El presente trabajo de diploma tuvo como objetivo desarrollar un módulo basado en tecnologías de cifrado que aumente la confidencialidad de los datos informáticos almacenados en NovaNas. Para ello se analizaron herramientas utilizadas actualmente para proteger los datos informáticos almacenados en las diferentes tecnologías y que pudiesen ser integradas a través de la implementación de un módulo a NovaNAS, seleccionando la herramienta LUKS como la solución más acertada. Además, se documentaron las tecnologías, herramientas, lenguajes a utilizar en la construcción del módulo y la definición de los elementos necesarios para el exitoso desarrollo del mismo, así como los artefactos requeridos por la metodología de desarrollo Variación del Proceso Unificado Ágil para la Universidad de las Ciencias Informáticas. Además este módulo fue sometido a pruebas de software para verificar su calidad y correcto funcionamiento. El principal resultado de la investigación fue la obtención de un módulo para el cifrado de los datos informáticos almacenados en NovaNAS que permite aumentar la confidencialidad de los datos informáticos almacenados en este sistema.

Palabras clave: almacenamiento de datos en red, cifrado, confidencialidad, datos informáticos, NovaNAS.

Índice

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

A lo largo de los años el almacenamiento de información se ha convertido en una necesidad para el hombre, debido al valor de la gestión de la información en el desarrollo de cada uno de los procesos de la sociedad. Con las Tecnologías de la Información y las Comunicaciones (TIC)er y su acelerado desarrollo, ha tenido lugar el surgimiento de nuevas formas de almacenamiento de información de manera digital, entre las que se encuentran, como una de las tecnologías que más ha contribuido en el almacenamiento de grandes volúmenes de información, los servidores NAS[1](Pradhan, 2015).

El almacenamiento conectado en red (NAS), es una tecnología de almacenamiento dedicada a compartir la capacidad de almacenamiento en un servidor con computadoras personales o servidores clientes a través de una red (normalmente TCP/IP), haciendo uso de un sistema operativo optimizado para dar acceso con los protocolos SMB[2], NFS[3] o FTP[4](Rouse, 2017). Este tipo de tecnología resulta de gran utilidad porque proporciona el almacenamiento centralizado a computadoras clientes en entornos con grandes cantidades de datos (Villazón, 2018).

Con el objetivo de contar con una tecnología para la gestión del almacenamiento de datos informáticos se desarrolló en el Centro de Software Libre (CESOL) de la Universidad de las Ciencias Informáticas (UCI), a partir de la distribución cubana GNU/Nova, una personalización de Nova Servidores para la administración del servicio NAS. Está basada en tecnologías libres y tiene como finalidad la gestión de información en los Organismos de la Administración Central del Estado, así como en otras empresas e instituciones en Cuba, como parte del proceso de informatización del país y la migración a software libre.

NovaNAS es una plataforma web que cuenta con un conjunto de módulos para la administración del servicio de Almacenamiento, Disponibilidad, Seguridad y Monitoreo de los datos de forma centralizada. Posee una

[1] El almacenamiento conectado en red (NAS, del inglés *Network Attached Storage*)
[2] La compartición de archivos (SMB, del inglés *Server Message Block*)
[3] Compartición en la red (NFS, del inglés *Network File System*)
[4] La transferencia de archivos (FTP, del inglés *File Transfer Protocol*)

interfaz intuitiva de monitoreo para el control y seguimiento de los recursos de hardware. Además permite administrar servicios y tecnologías como RAID[5], FTP, NFS, SMB/CIFS, ClamAV y Firewall (Villazón, 2018).

Cuando se utilizan tecnologías para la gestión de grandes volúmenes de información, como es NovaNAS, es necesario tener en cuenta las ventajas que, en materia de seguridad, estas brindan; puesto que cuando se trata con datos o información ya sea personal o de una empresa es importante que esos datos estén a salvo en todo momento, sobretodo si se trata de información sensible (SNIA, 2017). La información que se maneja debe cumplir con los pilares de seguridad: confidencialidad, disponibilidad e integridad de la información.

En la gestión del almacenamiento en NovaNAS intervienen un conjunto de tecnologías que propician el correcto funcionamiento de este servicio permitiendo que los clientes puedan escribir y leer información en cualquier momento garantizando la seguridad, disponibilidad e integridad de la información (Villazón, 2018). La relación entre estas tecnologías forma el Componente para la Gestión de la Integridad (CGI).

Esta relación está definida por un proceso que comienza cuando a cada carpeta compartida se le actualiza el hash de verificación tras una operación de escritura de sus datos por agentes autorizados. La información que se encuentra en estas carpetas es cifrada y sólo pueden ser accedidas una vez que se descifren al montarlas como sistemas de archivos virtuales mediante llaves públicas que solo las poseen los clientes autorizados, este proceso ocurre en el Componente para la Gestión del Almacenamiento (CGA). Mientras que, en el Componente para la Gestión de la Disponibilidad (CGD), el antivirus escanea mediante tareas programadas la información alojada en el sistema de almacenamiento tras operaciones de escritura (ver Figura 1)

[5] Conjunto redundante de discos independientes (RAID, del inglés *Redundant Array of Independent Disks*)

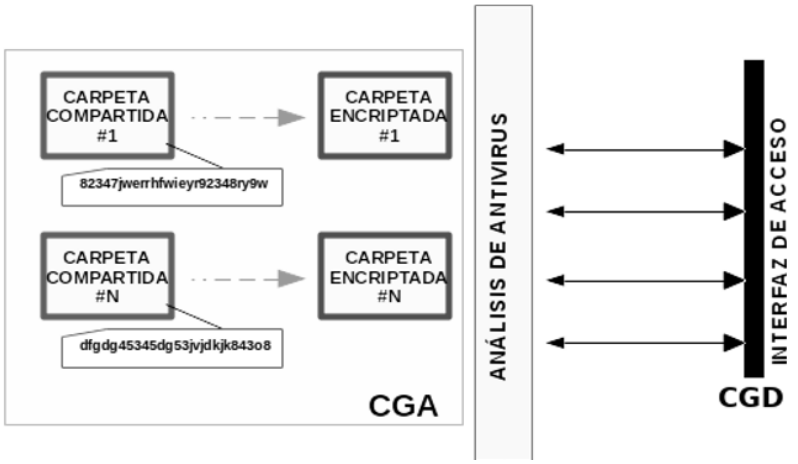


FIGURA 1: DESPLIEGUE DEL COMPONENTE DE INTEGRIDAD (FUENTE: (VILLAZÓN, 2018))

A pesar de la relación que se establece entre estas tecnologías, durante el proceso no se limita el acceso de usuarios no autorizados a los datos informáticos almacenados en NovaNAS debido a que no cuenta con un mecanismo que contribuya a la confidencialidad de los datos a través de la ocultación de la información a clientes no autorizados. Además, la existencia de datos compartidos por varios protocolos de comunicación, da lugar a que cierto número de elementos confidenciales puedan ser accedidos sin autorización y sin un mecanismo que gestione el control de acceso a estos datos.

Todo esto trae consigo el riesgo de que se recopile información confidencial almacenada que puede ser destruida o manipulada por un usuario sin ser detectado, se publique o modifique información privada sin autorización del autor original, o que a consecuencia del robo de esta tecnología la información sensible almacenada sea divulgada.

Por lo antes expuesto se define el siguiente **problema a resolver**: ¿Cómo aumentar la confidencialidad de los datos informáticos almacenados en NovaNAS?

De acuerdo al problema identificado y para guiar la investigación, se propone como **objeto de estudio**: el proceso para el cifrado de datos informáticos.

Teniendo en cuenta el problema a resolver se plantea como **objetivo general** de la investigación: Desarrollar un módulo basado en tecnologías de cifrado que aumente la confidencialidad de los datos informáticos almacenados en NovaNAS.

Centrando así el **campo de acción** en las tecnologías de cifrado aplicables a los datos informáticos almacenados en NovaNAS.

Para darle cumplimiento al objetivo general propuesto se definen los siguientes **objetivos específicos**:

1. Elaborar el marco teórico de la investigación sobre el proceso de cifrado de datos informáticos.
2. Identificar soluciones aplicables que aumenten la confidencialidad de los datos informáticos almacenados en NovaNAS.
3. Diseñar un módulo para NovaNAS que aumente la confidencialidad de los datos informáticos almacenados.
4. Implementar un módulo para NovaNAS que aumente la confidencialidad de los datos informáticos almacenados.
5. Evaluar el módulo para el cifrado de los datos informáticos almacenados en NovaNAS mediante pruebas de software.

Se plantean así las siguientes **preguntas científicas**:

1. ¿Qué fundamentos teóricos y metodológicos sustentan el desarrollo de un módulo para NovaNAS que aumente la confidencialidad de los datos informáticos almacenados?
2. ¿Cuáles son las tecnologías y herramientas existentes más adecuadas para el desarrollo de un módulo que aumente la confidencialidad de los datos informáticos almacenados en NovaNAS?
3. ¿Qué aspectos deben tenerse en cuenta para realizar el análisis y diseño del módulo para el cifrado de los datos informáticos almacenados en NovaNAS?
4. ¿Qué pruebas de software aplicar para validar el módulo que aumenta la confidencialidad de los datos informáticos almacenados en NovaNAS?
5. ¿Cómo comprobar el aumento de la confidencialidad luego del desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS?

Para dar cumplimiento a los objetivos específicos expuestos anteriormente se plantean las siguientes **tareas de investigación**:

1. Estudio de las soluciones existentes para el cifrado de datos informáticos.

2. Caracterización de la metodología, las tecnologías y las herramientas a utilizar para el desarrollo de la solución.
3. Definición de requisitos funcionales y no funcionales.
4. Diseño de la estructura y comportamiento de los componentes de la propuesta de solución.
5. Implementación de los componentes diseñados.
6. Validación de la propuesta de solución.

Métodos Científicos

La investigación realizada se sustenta en el empleo de los siguientes métodos científicos:

Métodos teóricos:

Analítico-Sintético: en cuanto se realizó el estudio de las tecnologías existentes que permiten aumentar la confidencialidad de los datos almacenados en sistemas informáticos; este método ayudó a analizar cuál de dichas tecnologías era la más apropiada para el desarrollo del módulo para NovaNAS. Fue utilizado durante el estudio y análisis de documentos, libros, artículos y otras fuentes bibliográficas de diferentes autores lo que permitió realizar una amplia investigación sobre las características de los principales algoritmos de cifrado, tecnologías y soluciones para aumentar la confidencialidad de los datos informáticos almacenados en NovaNAS.

Modelación: se utilizó para representar el contexto mediante diagramas que facilitaron la comprensión y desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS.

Método Inductivo-Deductivo: este método se utilizó para obtener conclusiones generales a partir de premisas particulares. En la investigación esto se realizó a través de enunciados que dan un sentido a la investigación (preguntas científicas).

Métodos empíricos:

Entrevista: se aplicó una entrevista al Msc. Yasiel Perez Villazón (especialista que estuvo involucrado en el desarrollo de NovaNAS), en dicha entrevista se identificaron con claridad los servicios que debía proporcionar el módulo para el cifrado de los datos informáticos almacenados en NovaNAS (ver Anexo 1).

Medición: este método se desarrolló con el objetivo de obtener una información numérica acerca de una propiedad, en este caso confidencialidad, donde se comparan magnitudes medibles conocidas. Para esto se cuenta con procedimientos estadísticos. Con el uso de este método se pudo conocer el cumplimiento del objetivo general de la investigación: desarrollar un módulo basado en tecnologías de cifrado que aumente la confidencialidad de los datos informáticos almacenados en NovaNAS.

El presente trabajo de diploma se encuentra dividido en tres capítulos estructurados de la siguiente forma:

Capítulo I: **Fundamentación teórica del módulo para el cifrado de los datos informáticos almacenados en NovaNAS**. Se definen los principales conceptos relacionados con el tema de investigación para lograr un mejor entendimiento del problema a resolver, lo que incluye un estudio del proceso de cifrado de datos y un análisis de las tecnologías homólogas existentes que aumentan la confidencialidad de los datos informáticos, así como una descripción de la metodología, las herramientas y tecnologías a emplear resultando en la selección de una de estas para llevar a cabo el desarrollo de un módulo para el cifrado de los datos informáticos almacenados en NovaNAS.

Capítulo II: **Análisis y diseño del módulo para el cifrado de los datos informáticos almacenados en NovaNAS**. En este capítulo se describen elementos tales como: los principales conceptos relacionados con el módulo para el cifrado de los datos informáticos almacenados en NovaNAS, la técnica empleada para la captura y validación de requisitos, requisitos funcionales y no funcionales, diseño arquitectónico que se aplicará al módulo para el cifrado de los datos informáticos almacenados en NovaNAS, diagrama de clases del diseño según el marco de trabajo que se utiliza y patrones de diseño.

Capítulo III: **Implementación y prueba del módulo para el cifrado de los datos informáticos almacenados en NovaNAS**. En este capítulo se comienza definiendo los estándares de codificación, luego el diseño de casos de prueba para realizar la validación del módulo para el cifrado de los datos informáticos almacenados en NovaNAS, además se muestran los resultados de las pruebas realizadas a dicho módulo mediante las estrategias de prueba. Las mismas se dividen en niveles de pruebas y cada nivel tiene un método de prueba. De estos métodos se explica su funcionamiento y los resultados una vez realizadas las pruebas, con el objetivo de detectar errores relacionados con sus funcionalidades y para comprobar la veracidad del código. Finalmente se realiza la evaluación del objetivo final a través de la obtención y comparación de los valores de confidencialidad.

CAPÍTULO I: Fundamentación teórica del módulo para el cifrado de los datos informáticos almacenados en NovaNAS

En este capítulo se definen los principales conceptos que se relacionan con la investigación sobre la confidencialidad de los datos almacenados en sistemas informáticos, incluye un análisis detallado de las tecnologías homólogas para el cifrado de los datos informáticos, se exponen sus principales características y una breve explicación sobre el proceso de cifrado de los datos informáticos, lo que contribuye a esclarecer el objeto de estudio acerca del proceso de cifrado de los datos informáticos. Además se describe la metodología, las herramientas y tecnologías seleccionadas para el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS.

1.1 Conceptos y definiciones

El siguiente epígrafe contiene la definición de los conceptos que a criterio de la autora son los principales en el desarrollo del trabajo investigativo. A continuación se describen dichos conceptos.

Cifrado**:** la encriptación o cifrado de archivos, es un procedimiento que vuelve completamente ilegibles los datos de un documento o de cualquier archivo. De esta manera, el archivo se vuelve prácticamente inutilizable para un usuario no autorizado a leerlo, ya que incluso si lo ha interceptado o lo ha copiado, no podrá leerlo o visualizarlo (Galicia, 2010).

Herramienta de cifrado**:** es aquella que permite proteger la información mediante un proceso que altera su contenido de legible a ilegible, esto con el objetivo de evitar que un tercero pueda tener acceso a ella (ALEGSA, 2018).

Confidencialidad: es la capacidad de un producto de software o sistema para asegurar que los datos son accesibles solo para aquellos usuarios autorizados a tener acceso (NC, 2016).

Nova Servidores: distribución de GNU/Linux cubana orientada a servidores basada en Ubuntu Server. Es un sistema optimizado para reducir al máximo el consumo de los recursos de hardware (no tiene interfaz gráfica) (Villazón, 2018).

NAS: tecnología de almacenamiento dedicada a compartir la capacidad de almacenamiento en un servidor con computadoras personales o servidores clientes a través de una red (normalmente TCP/IP), haciendo uso de un sistema operativo optimizado para dar acceso con los protocolos SMB, NFS o FTP(Villazón, 2018).

A continuación se realiza una breve descripción de NovaNAS, para una mejor comprensión de las características que posee esta tecnología, así como de las herramientas y tecnologías con que fue desarrollada.

1.2 Descripción de NovaNAS

Consiste en una personalización de una distribución cubana de GNU/Linux orientada a brindar servicio de almacenamiento conectado a la red (NAS). Como bien plantea Villazón esta es: "…una aplicación web para la administración de un servidor de almacenamiento en la red (NAS) desplegada con Nova Servidor 2015 que gestiona los servicios y herramientas de forma centralizada y provee una interfaz intuitiva…"(Villazón, 2018). Dentro de sus características más relevantes se encuentra su administración la cual es basada en la red, contiene ACLs[6], cuenta con un soporte multilenguaje y posee licencia GPL[7].

1.2.1 Arquitectura de NovaNAS

NovaNAS consta de una arquitectura en 2-Capas. En la figura se muestran las capas que componen la arquitectura, las cuales se describen a continuación:

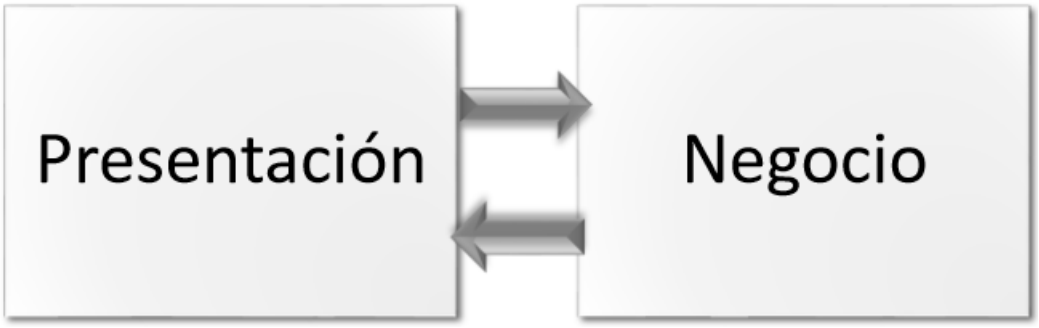


FIGURA 2: ARQUITECTURA DE NOVANAS (FUENTE: (VILLAZÓN, 2018))

[6] Listas de control de acceso (ACL, del inglés Access Control List)
[7] Licencia Pública General (GPL, del inglés *General Public License*)

- ✓ Capa de Presentación: es la encargada de presentar al usuario los conceptos de negocio mediante una UI[8], facilitar la explotación de dichos procesos, informar sobre la situación de los procesos de negocio e implementación de las reglas de validación de dicha interfaz. Permite interactuar con la aplicación. Se comunica con la capa de Negocio. Presenta todos los servicios que pueden ser administrados desde NovaNAS, así como diferentes opciones de configuración, seguridad, diagnóstico e información. Es desarrollada mediante Ext JS.
- ✓ Capa de Negocio: es donde son procesados los datos. Recibe y procesa peticiones del usuario. Establece las reglas del negocio. Emite respuestas a la capa de Presentación. Tiene asignada la responsabilidad de contener el código necesario para el manejo de datos. Es desarrollada mediante PHP[9] (Villazón, 2018).

1.2.2 Requisitos para integrar un módulo

Un módulo brinda nuevas funcionalidades, así como contenido específico. Según el Diccionario de Informática y Tecnología un módulo es: "…un software que agrupa un conjunto de subprogramas y estructuras de datos. Los módulos son unidades que pueden ser compiladas por separado y los hace reusables y permite que múltiples programadores trabajen en diferentes módulos en forma simultánea, produciendo ahorro en los tiempos de desarrollo…" (ALEGSA, 2018).

En el caso de NovaNAS el módulo debe cumplir con los requisitos siguientes:

- ✓ Estar en correspondencia con la arquitectura n-capas, de modo que se integre a cada una de las capas definidas en NovaNAS (ver epígrafe 1.2.1).
- ✓ Uso correcto y adecuado de las herramientas y tecnologías empleadas en el desarrollo de NovaNAS: Ext JS y PHP.

La autora considera necesario, además de la descripción de NovaNAS, realizar una breve explicación sobre qué es el cifrado, cómo funciona, qué tipo de información se puede cifrar, qué métodos de cifrado existen y por qué es necesario cifrar información. Lo que se pretende es ofrecer una introducción de manera general

[8] Interfaz de usuario (UI, del inglés *User Interface*)
[9] PHP, del inglés *Hypertext Pre-processor*

a los conceptos y procesos que intervienen en el cifrado de discos. Sobre estos temas trata el siguiente epígrafe.

1.3 Cifrado de discos

El cifrado de discos garantiza que los archivos siempre se almacenan en el disco de forma cifrada. Los archivos solo están disponibles en forma legible para el sistema operativo y las aplicaciones, mientras el sistema está funcionando y desbloqueado por un usuario de confianza. Una persona no autorizada que observe el contenido del disco directamente solo encontrará datos aleatorios confusos en lugar de los archivos reales. A continuación, se describen conceptos claves para una mejor comprensión sobre el cifrado de discos, comenzando por el principio básico de funcionamiento de cifrado.

1.3.1 Principio básico de funcionamiento de cifrado

A los efectos del cifrado de discos, cada dispositivo de bloque (o archivo individual en el caso del cifrado de sistemas de archivos apilados) se divide en **sectores** de igual longitud, por ejemplo, 512 bytes (4096 bits). El cifrado/descifrado se realiza después en función de cada sector, por lo que el sector n del archivo/dispositivo de bloque en el disco almacenará la versión cifrada del sector n de los datos originales (Loshin, 2013). Cada vez que el sistema operativo o una aplicación solicita un determinado fragmento de datos del archivo/dispositivo de bloque, todo el sector (o sectores) que contienen los datos serán leídos desde el disco, descifrados sobre la marcha y almacenados temporalmente en la memoria.

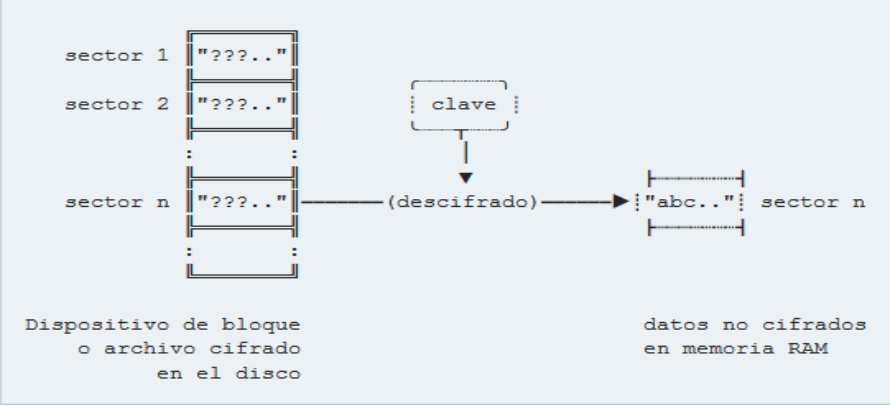


FIGURA 3: FUNCIONAMIENTO BÁSICO DEL CIFRADO DE DISCOS (FUENTE: (GALICIA, 2010))

Del mismo modo, en cada operación de escritura, todos los sectores que se ven afectados deben ser cifrados de nuevo completamente (mientras que el resto de los sectores permanecen intactos).

Con el fin de ser capaz de descifrar/cifrar los datos, el sistema de cifrado del disco necesita saber la "clave" única secreta asociada a él. Cada vez que el dispositivo de bloque o la carpeta en cuestión, sea montada, su clave correspondiente (llamada a partir de ahora "llave maestra") debe ser suministrada (Loshin, 2013).

Con relación a la clave se debe destacar que es de suma importancia mantenerla en secreto y protegida de usuarios no deseados. Para ello existen también técnicas de cómo almacenar y asegurar criptográficamente una llave maestra con un archivo de claves que se mencionan a continuación:

- ✓ Almacenada en un archivo de claves de texto plano.
- ✓ Almacenada en forma de frase de acceso protegida en un archivo de claves o en el propio disco.
- ✓ Generada de forma aleatoria sobre la marcha para cada sesión.

Existen diferentes métodos para el cifrado de discos, en el siguiente epígrafe se realizará una breve descripción de estos métodos.

1.3.2 Métodos de cifrado de discos

Cifrar sistemas de archivos apilados:

Las soluciones del cifrado del sistema de archivos apilados se implementan como una capa que se superpone en la parte superior de un sistema de archivos existente, haciendo que todos los archivos escritos a una carpeta habilitada para cifrarlos se cifren sobre la marcha antes de que el sistema de archivos subyacente los grabe en el disco, y se descifran cuando el sistema de archivos los lee desde el disco. De esta manera, los archivos se almacenan en el sistema de archivos anfitrión, pero, aparte, siguen existiendo en ese sistema de archivos como lo harían sin cifrar, a modo de archivos normales / enlaces simbólicos, entre otros.

Cifrar dispositivos de bloque:

Los métodos de cifrado de dispositivo de bloque, por el contrario, operan como una capa debajo del sistema de archivos y se aseguran que todo lo escrito en un determinado dispositivo de bloques (es decir, un disco

completo o una partición) quede cifrado. Esto significa que mientras el dispositivo de bloque no está en línea, todo su contenido se parece a un conjunto de datos aleatorios, y no hay forma de determinar qué tipo de sistema de archivos y qué datos contiene. El acceso a los datos ocurre, de nuevo, mediante el montaje del contenedor protegido (en este caso el dispositivo de bloque) a una ubicación arbitraria con un método especial.

1.3.3 Algoritmos de cifrado y modalidades de operación

El algoritmo real utilizado para servir de traductor entre las piezas de datos no cifradas y las cifradas (llamado "texto plano" y "texto cifrado", respectivamente), que se interrelacionan entre sí a través de una clave de cifrado dada, se llama un "algoritmo de cifrado" (Loshin, 2013).

El cifrado de discos emplea "algoritmos de cifrado de bloques", que operan sobre bloques de datos de una longitud fija, por ejemplo 16 bytes (128 bits). En el momento de escribir estas líneas, los más usados son:

TABLA 1: ALGORITMOS BÁSICOS DE CIFRADO (FUENTE: (GALICIA, 2010))

Nombre	Tamaño del bloque	Tamaño de la clave
AES	128 bits	128, 192 o 256 bits
Blowfish	64 bits	32 - 448 bits
Twofish	128 bits	128, 192, o 256 bits
Serpent	128 bits	128, 192 o 256 bits

1.4 Análisis de soluciones para el cifrado de datos informáticos

Existen diversas tecnologías en el mercado que garantizan la confidencialidad de los datos informáticos a través del cifrado de datos, tanto soluciones propietarias como libres. Cuando se utilizan soluciones propietarias no se tiene control de lo que está haciendo el software, no se pueden comprobar por vulnerabilidades o puertas traseras y por tanto este software puede ser no seguro (Halcrow, 2016). En cuanto al software de cifrado de código abierto, el código fuente está abierto, las vulnerabilidades se pueden encontrar y reparar, y las puertas traseras se pueden notar (Halcrow, 2016).

Todo esto, unido a las políticas establecidas en el país guiadas hacia la informatización de la sociedad y la migración a software libre, descarta entonces la idea de emplear un software de cifrado propietario. En GNU/Linux existen múltiples opciones para el cifrado de los datos informáticos: eCryptfs o EncFS sobre

directorios, TrueCrypt o VeraCrypt sobre dispositivos y CryFS; otra de las opciones es LUKS[10], que se puede usar sobre discos, particiones, volúmenes lógicos (LVs) o ficheros loop.

A continuación, se presenta la información obtenida de un estudio realizado sobre las tecnologías existentes para el cifrado de los datos informáticos, así como, un análisis sobre cuál de ellas es la que se adapta al desarrollo del módulo para NovaNAS.

Truecrypt

Inicialmente lanzado en febrero de 2004, es un programa de código abierto multiplataforma para cifrado de archivos y de disco completo. Es una utilidad de software para el cifrado sobre la marcha. Puede crear un disco virtual cifrado dentro de un archivo, o cifrar una partición o todo el dispositivo de almacenamiento con autenticación previa al arranque. Su última versión permite elegir entre varios algoritmos, incluyendo Serpent, AES y Twofish. El soporte a TrueCrypt ha sido descontinuado hace algún tiempo y tiene algunas vulnerabilidades (Halcrow, 2016). Como alternativa a este programa se incluye un proyecto de software gratuito basado en el código TrueCrypt, llamado VeraCrypt.

VeraCrypt

VeraCrypt se ejecuta en Windows, Linux y Mac, y se considera una herramienta de cifrado segura para cifrar los archivos localmente. Mantiene los archivos confidenciales, pero no protege la integridad, es decir, un usuario no deseado no puede leer los archivos, pero podrían modificarlos sin que el usuario autorizado se dé cuenta. Ofrece un sistema de archivos virtual y los archivos sin cifrar nunca se almacenan en el disco duro. VeraCrypt luego almacena todos los datos del sistema de archivos en un archivo contenedor cifrado (Halcrow, 2016).

A pesar de ser muy conveniente, seguro y fácil de usar, VeraCrypt no está diseñado para ser utilizado en la nube y un archivo contenedor puede ser muy grande. En realidad, se debe elegir un tamaño máximo para el sistema de archivos con anticipación y el archivo contenedor tendrá este tamaño, sin importar la cantidad de espacio que realmente se utilice. Por lo tanto, VeraCrypt es una buena opción si solo se están cifrando los archivos localmente y no se necesita integridad.

[10] LUKS, del inglés *Linux Unified Key Setup*

EncFs

EncFS es un software de código abierto, licenciado bajo la licencia GPL. Proporciona un sistema de archivos cifrado en el espacio de usuario y se ejecuta sin ningún permiso especial, es decir, opera utilizando el método de cifrado de sistemas de archivos apilados. De hecho, no es tanto un sistema de archivos como un programa que traduce solicitudes (cifrándolos o descifrando según corresponda) y los pasa al sistema de archivos subyacente. Utiliza la biblioteca del sistema de archivos en el espacio de usuario (FUSE)[11] y el módulo del kernel para proporcionar la interfaz del sistema de archivos (Encfsmp, 2010).

Utiliza un sistema de archivos de paso en lugar de un dispositivo de bloque cifrado. Con un dispositivo de bloque, debe pre-asignar el tamaño del espacio de datos que desea cifrar, mientras que un sistema de archivos de paso permite que el tamaño de los datos cifrados crezca o se reduzca sin ser reformateado. Además puede realizar copias de seguridad archivo por archivo y, en el caso de copias de seguridad incrementales o diferenciales, el programa de copia de seguridad sabrá qué archivos han cambiado desde la última copia de seguridad, incluso si el programa puede descifrar lo que contiene cada archivo (Loshin, 2013).

Opcionalmente, EncFS ofrece una implementación de integridad, pero la implementación no sigue ningún estándar y contiene algunas fallas. Un usuario no deseado puede modificar sus archivos simplemente desactivando la verificación de integridad y nunca se notaría. Incluso si se almacena el archivo de configuración localmente donde un usuario no deseado no lo pueda modificar, EncFS solo ofrece integridad en un nivel por archivo. No impide que se agreguen o eliminen archivos y directorios (Galicia, 2010). Por lo tanto, la versión actual de EncFS no se puede recomendar para ninguna aplicación.

eCryptfs

eCryptfs es un sistema de archivos apilados criptográficos empresariales POSIX para Linux. eCryptfs almacena metadatos criptográficos en el encabezado de cada archivo, para que los archivos cifrados se puedan copiar entre los hosts (computadoras); el archivo será descifrado con la clave adecuada en el anillo de claves del kernel de Linux. No es necesario realizar un seguimiento de la información adicional, aparte

[11] Sistema de archivos en el espacio de usuario (FUSE, del inglés *Filesystem in Userspace*)

de lo que ya está en el archivo cifrado. Se puede pensar en eCryptfs como una especie de "gnupg como sistema de archivos" (Loshin, 2013).

eCryptfs es ampliamente utilizado, como base para el directorio de inicio cifrado de Ubuntu, nativo dentro del ChromeOS de Google, y se incrusta de manera transparente en varios dispositivos de almacenamiento conectado a la red (NAS). Es lo que utiliza Ubuntu cuando se activa el directorio *home* cifrado (Ecryptfs, 2017). Funciona como una herramienta combinada entre GnuPG y Cryptsetup. eCryptfs está actualmente mantenido activamente por Dustin Kirkland (de Canonical, Inc) y Tyler Hicks (de Canonical, Ltd).

CryFS

Esta solución ofrece un sistema de archivos virtual y el usuario puede trabajar con sus archivos sin pensar en el cifrado que está ocurriendo en el fondo. CryFS sigue los estándares de seguridad establecidos. A diferencia de VeraCrypt, mantiene sus datos en pequeños bloques cifrados y al cambiar un archivo pequeño se obtiene una pequeña cantidad de datos para volver a cargarlos. A diferencia de EncFS y eCryptfs, no solo cifra el contenido de sus archivos, sino también los tamaños de los archivos, los metadatos de los archivos y la estructura de directorios (CryFS, 2015).

CryFS ofrece confidencialidad de sus datos, pero solo es un nivel básico de integridad. Evita que los usuarios no deseados introduzcan nuevo contenido en sus archivos o agreguen nuevos archivos o eliminen archivos, pero no evita que los atacantes hagan retroceder sus archivos o directorios a una versión anterior válida(CryFS, 2015). Otro posible inconveniente de CryFS es que es relativamente nuevo y en este momento solo funciona para Linux. La seguridad de CryFS se ha demostrado en una tesis de maestría en 2015. Los desarrolladores aún no consideran que la versión actual sea estable y, si se decide utilizarla, se recomienda realizar copias de seguridad periódicas (CryFS, 2015).

LUKS

Es una solución de cifrado integrada al kernel como módulo, ofreciendo acceso a las API de Crypto del kernel de Linux. Mientras la mayoría del software de cifrado de discos implementa diferentes e incompatibles formatos no documentados, LUKS especifica un formato estándar en disco, independiente de plataforma, para usar en varias herramientas. Esto no solo facilita la compatibilidad y la interoperabilidad entre los

diferentes programas, sino que también garantiza que todas ellas implementen gestión de contraseñas en un lugar seguro y de manera documentada (Galicia, 2010).

La web de tecnología Phoronix publicó una comparativa de rendimiento entre LUKS y eCryptfs. Usó la versión, existente a esa fecha, de Ubuntu 13.10 y obtuvo mejor rendimiento cifrando todo el disco con LUKS, que solo el directorio de datos de usuario con eCryptfs. Entre sus ventajas están, además: la sencillez de uso, incluido en el propio kernel y la capacidad de asignar, cambiar y revocar varias claves para un mismo dispositivo.

1.4.1 Comparación entre las tecnologías homólogas existentes

Es necesario que en el momento de establecer los parámetros para la selección de la solución para el cifrado de datos, se tenga en cuenta que la solución se pondrá en práctica en un entorno empresarial, partiendo de la idea que NovaNAS fue creado con el fin de proveer a los Organismos de Administración Central del Estado (OACE), empresas y/o instituciones en Cuba de una tecnología para la gestión del almacenamiento de datos informáticos, basado en tecnologías libres, para la gestión de información; y lo que se pretende es garantizar la confidencialidad de los datos informáticos almacenados en esta tecnología desplegada en un entorno empresarial.

Existen diferentes soluciones en cuanto al cifrado de datos informáticos, pero la mayoría poseen desventajas, a continuación en la tabla 2 se cita una comparación entre las tecnologías estudiadas en el epígrafe anterior (CryFS, 2015).

TABLA 2: COMPARACIÓN ENTRE LAS SOLUCIONES PARA EL CIFRADO DE DATOS INFORMÁTICOS (FUENTE:(CRYFS, 2015))

Parámetros de comparación	CryFS	EncFS	eCryptfs	VeraCrypt	LUKS
Fácil de usar	✓	✓	✓	✓	✓
Pequeños cambios hacen que solo se vuelva a cargar una pequeña cantidad de datos	✓	✓	✓	NO	✓
No se conocen fallas de seguridad	✓	Presenta algunas	✓	✓	✓

		fallas de seguridad			
Cifra el contenido del archivo	✓	✓	✓	✓	✓
Mantiene la confidencialidad de los datos	✓	✓	✓	✓	✓
Disponible en Linux	✓	✓	✓	✓	✓
Soporte para redimensionar (manualmente) el dispositivo de bloque cifrado	✓	NO	✓	NO	✓
Capacidad para añadir, eliminar y modificar varias claves para un mismo dispositivo	NO	NO	NO	NO	✓

Analizando la tabla comparativa y teniendo en cuenta el estudio de las tecnologías homólogas existentes para el cifrado de datos, el análisis del modo de actuación y la comparación entre las diferentes soluciones libres, se define como solución a emplear LUKS. Esta herramienta es una especificación de cifrado de disco creado por Clemens Fruhwirth, originalmente destinado para Linux. LUKS, diseñada e implementada teniendo en cuenta el entorno empresarial, aborda directamente varios de los problemas que se presentan en la práctica durante la aplicación de alguna herramienta de cifrado de datos informáticos (J. Callas, 1998).

Dentro de las principales características de esta herramienta se encuentran:

- ✓ Es una herramienta para Linux, integrada al kernel como un módulo. Ha sido diseñada para ajustarse a la clave de configuración TKS1 de sistema seguro
- ✓ Emplea el método de cifrado a nivel de bloque.
- ✓ Soporta los algoritmos de cifrado AES[12], Blowfish[13], Twofish[14].

[12] AES, del inglés *Advance Encryption Standard* (Es un esquema de cifrado por bloques)
[13] Blowfish, en criptografía, es un codificador de bloques simétricos.
[14] Twofish, en criptografía, es un método de criptografía simétrica con cifrado por bloques.

- ✓ Ofrece encriptación transparente y la posibilidad de mapear dispositivos y particiones en niveles de bloques virtuales
- ✓ Permite cifrar particiones, discos duros completos, volúmenes RAID, volúmenes lógicos, archivos o unidades extraíbles (ver Figura 4)
- ✓ Se incrusta de manera transparente en dispositivos de almacenamiento conectado a la red (NAS).
- ✓ Las particiones o unidades de almacenamiento externo cifradas con LUKS, pueden ser utilizadas desde Windows utilizando FreeOTE.
- ✓ Almacena metadatos criptográficos en el encabezado de cada archivo.
- ✓ Permite claves múltiples de usuario para descifrar una clave maestra que se usa para el cifrado de la partición.

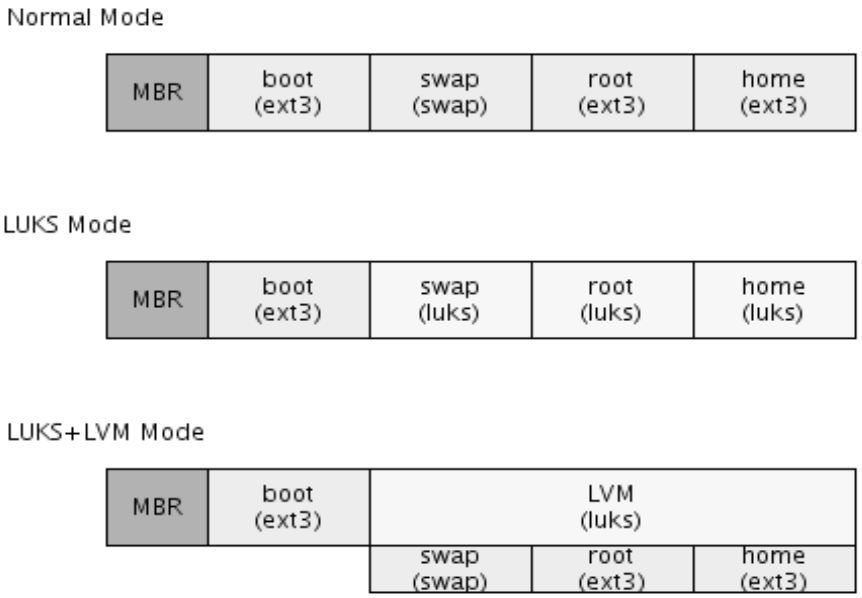


FIGURA 4: ESTRUCTURA DE DISPOSITIVOS CON FORMATO LUKS (FUENTE: (GALICIA, 2010))

Descripción general de la arquitectura de LUKS

LUKS es una especificación de cifrado de disco estándar que no solo facilita la compatibilidad y la interoperabilidad entre los diferentes programas de cifrado que lo utilicen, sino que también garantiza que todas ellas implementen gestión de contraseñas en un lugar seguro y de manera documentada. La

implementación en GNU/Linux se encuentra como añadido en cryptsetup, utilizando dm-crypt como la interfaz de cifrado de disco.

LUKS es único de la mayoría de las soluciones de sistemas de archivos criptográficos, ya que almacena un conjunto completo de metadatos criptográficos junto con cada archivo individual. Los archivos cifrados con PGP[15] (J. Callas, 1998)están formateados, esto permite que los archivos cifrados se transfieran a través de dominios de confianza mientras se mantiene la capacidad para que aquellos con las credenciales adecuadas obtengan acceso a esos archivos (Halcrow, 2016). Debido a que el cifrado y el descifrado tienen lugar en la capa del sistema de archivos virtual (VFS)[16], el proceso se hace transparente desde la perspectiva de la aplicación.

LUKS se implementa como un módulo de kernel aumentado con varias utilidades de espacio de usuario para realizar funciones de administración de claves. Esta provee cifrado transparente de dispositivos de bloque y ofrece una capa de abstracción. Las escrituras a este dispositivo serán cifradas y las lecturas descifradas. Debido a esa capa de abstracción que ofrece se podrá montar un sistema de archivos en el mismo de la manera habitual, pero no se podrán acceder los datos sin la clave.

Funcionamiento de LUKS

El modo de funcionamiento de LUKS va más allá del funcionamiento tradicional de las herramientas de cifrado, utilizando mecanismos para garantizar la seguridad, estos mecanismos son el establecimiento de un doble nivel de clave y el almacenamiento seguro de la clave principal. El establecimiento de un doble nivel de clave consiste en que, en lugar de utilizar la clave suministrada por el usuario para cifrar los datos, lo que hace es generar una clave maestra del tamaño deseado, criptográficamente segura, y almacenarla cifrada con la clave del usuario. Cuando se quiere acceder a los datos cifrados, el sistema solicita al usuario su contraseña, con ella y la clave maestra cifrada inicializa un sistema de protección y con este accede a los datos.

En el caso del mecanismo para el almacenamiento seguro de la clave principal, lo que ocurre es que no se almacena la clave maestra cifrada en un sector concreto, sino que se particiona en varios trozos de forma

[15] Privacidad bastante buena (PGP, del inglés *Pretty Good Privacy*). Consiste en un criptosistema híbrido que combina técnicas de criptografía simétrica y criptografía asimétrica.
[16] Sistema de archivos virtual (VFS, del inglés *Virtual filesystem*)

que todos sean necesarios para recuperar la clave (por ejemplo, mediante un XOR) y almacenando cada trozo en un lugar diferente. La suma de ambos mecanismos es lo que se conoce como esquema TKS1 y es la base del funcionamiento de LUKS: una clave principal almacenada de forma segura, con la que se cifran los datos, y una o varias contraseñas que los usuarios utilizan para acceder a la clave principal y desbloquear el acceso a los datos.

Cuando se hace referencia al formato LUKS se emplean términos que en ocasiones parecen similares pero tratan, en realidad, de conceptos diferentes: clave de cifrado, contraseña y cabecera LUKS.

Clave de cifrado: esta clave se genera automáticamente cuando se realiza el formato LUKS al dispositivo. Se genera de forma aleatoria (por defecto usando /dev/urandom) y con el tamaño por defecto (256 bits) o el que se especifique.

Contraseña: puede ser tecleada al momento, o un fichero usado como tal, es la que permite acceder al dispositivo LUKS. Esta contraseña no es usada para cifrar los datos. De hecho, una de las ventajas de usar LUKS es que se pueden usar varias contraseñas/ficheros contraseñas para un mismo dispositivo.

Cabecera LUKS: el formato LUKS se caracteriza por una cabecera que incluye la información del algoritmo de cifrado, tamaño de clave, contraseñas en uso, entre otros. Se puede consultar dicha cabecera pero no las contraseñas.

1.5 Metodología del software

Las metodologías de desarrollo de software proporcionan un camino más fácil y seguro para el desarrollo, siendo éstas un conjunto de procedimientos, técnicas, herramientas y un soporte documental, que ayuda a los desarrolladores a realizar nuevo software. La metodología define roles los cuales indican quién debe hacer qué, cuándo y cómo debe hacerlo para obtener los distintos productos parciales y finales.

No existe una metodología de software universal, por lo que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Por lo tanto, para el desarrollo del módulo para el cifrado de datos informáticos almacenados en NovaNAS se decide emplear la metodología variación AUP[17], porque ésta se adapta a las características

[17]Proceso Unificado Ágil (AUP, del inglés *Agil Unified Process*)

del proyecto según los recursos disponibles, el número de personas involucradas y al ciclo de vida definido para la actividad productiva de la UCI.

Una metodología de desarrollo de software tiene entre sus objetivos aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas, para ello nos apoyaremos en el Modelo CMMI-DEV v1.3. El cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas se centran en el desarrollo de productos y servicios de calidad(Sánchez, 2014).

Las 3 fases que se proponen en la metodología Variación AUP para el ciclo de vida de los proyectos de la UCI se muestra a continuación:

- ✓ Inicio: En esta fase se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- ✓ Ejecución: Durante esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.
- ✓ Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Esta metodología consta además de 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno), y 4 escenarios para la disciplina de requisitos. Con la Variación AUP para la actividad productiva en la UCI se logra estandarizar el proceso de desarrollo de software, dando cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3. (Sánchez, 2014). Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos. Se redujo a 1 la cantidad de metodologías que se usaban y de más de 20 roles en total que se definían se redujeron a 11(Sánchez, 2014).

1.6 Tecnologías y herramientas de desarrollo

Para llevar a cabo el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS, dar cumplimiento a los objetivos de esta investigación, y teniendo en cuenta las características del producto que se desea obtener; es necesario realizar una descripción de las herramientas y tecnologías implicadas en el proceso de desarrollo. A continuación, se describen las tecnologías y herramientas de desarrollo que se utilizarán para realizar este módulo.

Lenguaje de modelado: UML 2.0

Se utiliza UML[18] en su versión 2.0 como lenguaje de modelado, ya que el mismo permite al equipo de desarrollo visualizar, especificar, construir y documentar los artefactos del sistema. Esto proporciona una forma estándar de escribir los planos del mismo y cubrir tanto las cosas conceptuales, tales como procesos del negocio y funciones del sistema, como las cosas concretas, tales como las clases escritas en un lenguaje de programación específico, esquemas de bases de datos y componentes software reutilizables (Sánchez, 2014).

Herramienta CASE: Visual Paradigm 8.0

Es una herramienta que facilita y permite al equipo de desarrollo visualizar, diseñar, integrar y distribuir la aplicación que se necesita. Esto se logra mediante la creación de modelos UML, el modelado de base de datos lo que proporciona una mayor documentación de la misma y diagramas de mapeo de relación de objetos, el modelo de procesos de negocios lo que permite la visualización, improvisación y entendimiento de los procesos. Además ayuda a crear un modelo de excelencia durante la creación y distribución del proceso de desarrollo de aplicaciones (Larman, 2004).

Lenguaje de programación: PHP 5.4.4

Es un lenguaje interpretado de propósito general ampliamente usado, diseñado especialmente para desarrollo web y que puede ser incrustado dentro de código HTML2. Es muy rápido, pues permite al equipo de desarrollo la generación dinámica de páginas, su integración con las bases de datos y el servidor Apache.

[18] Lenguaje Unificado de Modelado (UML, del inglés Unified Modeling Language)

PHP es multiplataforma y funciona tanto para Unix como para Microsoft Windows, de manera que el código creado para Unix no sufre modificaciones al ser ejecutado en Windows y viceversa.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. Debido a su amplia distribución y su característica de código abierto, PHP está perfectamente soportado por una gran comunidad de desarrolladores, lo que propicia una rápida reparación de los fallos de funcionamiento que sean detectados (Rosales, 2014).

Entorno de desarrollo Integrado (IDE):

Un Entorno de Desarrollo Integrado (IDE[19]) es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. Los componentes de cualquier entorno de desarrollo integrado son un editor de texto, un compilador, un intérprete, un depurador, que tenga posibilidad de ofrecer un sistema de control de versiones y que ayude en la construcción de interfaces gráficas de usuario (Quiroz, 2012).

PhpStorm

Es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación PHP como lo indica su nombre (Perez, 2009). Estas son sólo algunas de las características principales:

- ✓ Permite la gestión de proyectos fácilmente.
- ✓ Proporciona un fácil autocompletado de código.
- ✓ Soporta el trabajo con PHP 5.5.
- ✓ Sintaxis abreviada.

Framework de desarrollo

Un framework de desarrollo es un software que ofrece una infraestructura para la creación de otros programas. El framework contiene librerías de código y módulos ya listos que resumen las tareas de

[19] Entorno de Desarrollo Integrado (IDE, del inglés Integrated Development Environment)

creación de elementos recurrentes en el desarrollo de aplicaciones, a la vez que define una arquitectura para el desarrollo de software (Soto, 2015).

Ext JS

Es una librería Javascript que permite construir aplicaciones complejas en Internet. Esta librería incluye:

- ✓ Componentes UI del alto performance y personalizables.
- ✓ Modelo de componentes extensibles.
- ✓ Un API fácil de usar.
- ✓ Licencias Open Source y comerciales.
- ✓ Permite el desarrollo de interfaces en la capa de Presentación. Comunica y captura información.

1.6 Conclusiones del capítulo

En este capítulo, luego de un estudio sobre el estado actual y funcionamiento de NovaNas se demostró la necesidad de crear una aplicación para el cifrado de los datos informáticos almacenados, que se adapte a las necesidades y prestaciones del sistema de almacenamiento conectado a la red actualmente en funcionamiento (NovaNAS).

El estudio, análisis y comparación de las diferentes soluciones existentes para el cifrado de datos informáticos; arrojó como resultado la selección de la solución libre para el cifrado de datos informáticos LUKS. Se realizó una descripción de la metodología, herramientas y tecnologías de desarrollo seleccionadas para llevar a cabo el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS, definiendo la Variación AUP-UCI como metodología a utilizar, PhpStorm como plataforma de desarrollo, Visual Paradigm 8.0 como herramienta de modelado y PHP como lenguaje de programación para el desarrollo del software

CAPITULO II: Análisis y diseño del módulo para el cifrado de los datos informáticos almacenados en NovaNAS

En el siguiente capítulo se describen las características del módulo para el cifrado de los datos informáticos almacenados en NovaNAS; se especifica mediante la representación de conceptos y sus relaciones la situación actual del sistema NovaNAS y su funcionamiento, las características que debe poseer el módulo a desarrollar, planteadas en requisitos funcionales y no funcionales. Además se representa el diseño arquitectónico a emplear durante el desarrollo de dicho módulo, el diagrama de clases del diseño según el marco de trabajo que se utiliza y patrones de diseño. Por otra parte, se elabora el modelo de implementación el cual está representado por los estándares de codificación.

2.1 Descripción de la propuesta de solución

Analizando la situación problemática planteada en la investigación se propone como solución el desarrollo de un módulo para el cifrado de los datos informáticos que se integre a NovaNAS. Para ello se realizó un estudio de las tecnologías existentes para el cifrado de los datos informáticos, se concluyó que la solución más acertada es la selección de la herramienta libre para el cifrado de datos LUKS. Este módulo ofrece la posibilidad de cifrar/descifrar dispositivos de bloque, lo que permite limitar el acceso solo a los usuarios autorizados a acceder a los datos informáticos en cuestión, lo que además se traduce a un aumento de la confidencialidad de los datos informáticos que se almacenan en este servidor. Cuenta también con funcionalidades adicionales tales como: las claves múltiples de usuario para descifrar una clave maestra que se usa para el cifrado de la partición y la consulta de detalles del disco cifrado.

2.2 Representación de la propuesta de solución

Para la representación de la propuesta de solución se identificaron los conceptos significativos en el problema, identificando los atributos y las asociaciones entre ellos. Es una representación de las entidades, ideas, conceptos u objetos del mundo real (Larman, 2004). Es un diccionario visual de conceptos e información del dominio del problema. A continuación, se muestran los principales conceptos de la investigación, las relaciones entre ellos y sus atributos, para una mejor comprensión del contexto de la investigación (ver Figura 5).

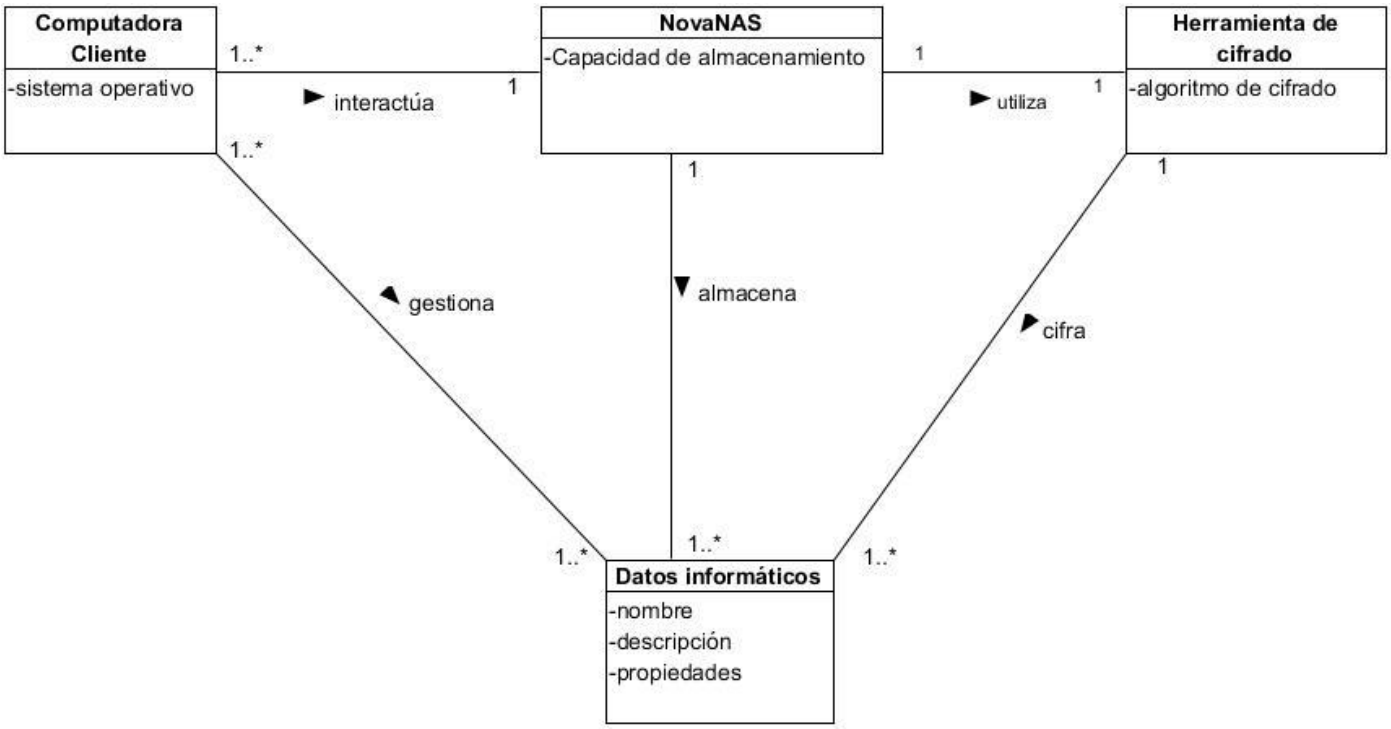


FIGURA 5: REPRESENTACIÓN DE LA PROPUESTA DE SOLUCIÓN (FUENTE: ELABORACIÓN PROPIA)

En la representación anterior una **Computadora cliente** o personal gestiona **Datos informáticos** e interactúa con el servidor de almacenamiento **NovaNAS**. Este utiliza una **Herramienta de cifrado** para cifrar los datos informáticos almacenados.

2.3 Descripción de conceptos

Computadora cliente: ordenador que consume un servicio remoto en otro ordenador conocido como servidor, normalmente a través de una red de telecomunicaciones (ALEGSA, 2018).

NovaNAS: consiste en una personalización de una distribución cubana de GNU/Linux orientada a brindar servicio de almacenamiento conectado a la red (NAS) (Villazón, 2018).

Herramienta de cifrado: es aquella que permite proteger la información mediante un proceso que altera su contenido de legible a ilegible, esto con el objetivo de evitar que un tercero pueda tener acceso a ella (Ecryptfs, 2017).

2.4 Requisitos

La ingeniería de requisitos comprende todas las tareas relacionadas con la determinación de las necesidades o de las condiciones a satisfacer para un software nuevo o modificado. Los requisitos del software son las condiciones necesarias sobre el contenido, forma o funcionalidades de un producto o servicio determinado. Los requisitos reflejan las necesidades de los clientes que ayudan a resolver determinado problema (Sommerville, 2011).

Existen diferentes técnicas para la captura de requisitos, la utilizada para la presente investigación se especifica a continuación**.**

Técnicas de captura de requisitos

- ✓ Entrevista: se le aplicó al MSc. Yasiel Perez Villazón, especialista involucrado en el proceso de desarrollo de NovaNAS, lo que permitió obtener información con la mayor calidad posible e interpretar las necesidades del cliente, identificando así los requisitos funcionales a los que debe responder el módulo.

Requisitos funcionales (RF)

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar la aplicación, de la manera en que debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. Describen lo que el sistema debe hacer (Pressman, 2002). Como resultado de la técnica aplicada para la captura de los requisitos, se identificaron un total de 12 Requisitos Funcionales. A continuación, se muestra una tabla con los requisitos identificados.

TABLA 3: REQUISITOS FUNCIONALES (FUENTE: ELABORACIÓN PROPIA)

No.	Nombre	Descripción	Prioridad
RF_1	Crear disco cifrado	El sistema debe permitir cifrar un disco.	Alta
RF_2	Mostrar listado de discos cifrados	El sistema debe permitir mostrar los discos cifrados con la información correspondiente (tamaño, el estado del disco (bloqueado o desbloqueado), dirección del disco cifrado, si tiene referencia (si, no, n/a), y las ranuras en uso)	Media
RF_3	Bloquear disco cifrado	El sistema debe permitir bloquear el acceso al disco cifrado	Media
RF_4	Desbloquear disco cifrado	El sistema debe permitir desbloquear el acceso al disco cifrado	Media
RF_5	Eliminar disco cifrado	El sistema debe permitir eliminar un disco cifrado.	Media
RF_6	Salvar cabecera	El sistema debe permitir realizar una copia de seguridad de la cabecera del disco cifrado.	Media
RF_7	Recuperar cabecera	El sistema debe permitir recuperar la cabecera del disco cifrado, previamente salvada(RF_6)	Media
RF_8	Añadir contraseña	El sistema debe permitir añadir una o varias contraseñas al disco cifrado (estableciendo un límite de 1 a 8 contraseñas por disco cifrado).	Media
RF_9	Eliminar contraseña	El sistema debe permitir eliminar una o varias contraseñas del disco cifrado.	Media
RF_10	Cambiar contraseña	El sistema debe permitir cambiar o modificar una o varias contraseñas de las establecidas en el disco cifrado.	Baja
RF_11	Comprobar contraseña	El sistema debe permitir comprobar la contraseña de un disco cifrado	Baja
RF_12	Mostrar detalles del disco cifrado	El sistema debe permitir mostrar detalles del disco seleccionado en cuanto a: algoritmo de cifrado, tamaño de clave, función hash y cantidad contraseñas establecidas.	Baja

Requisitos no funcionales (RNF)

Los RNF hacen referencia a las propiedades emergentes del sistema como restricciones de tiempo de respuesta, estándares, fiabilidad, capacidad de almacenamiento (Pressman, 2002). A continuación, se muestra un listado de los requisitos no funcionales definidos para el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS y agrupados según las clasificaciones de requisitos no funcionales existentes:

- ✓ Usabilidad:
  RNF_ 1 El sistema debe representar un diseño sencillo, con una interfaz agradable para el cliente y fácil de operar
- ✓ Fiabilidad:
  RNF_ 2 El sistema debe tener soporte para recuperación ante fallos y errores.
- ✓ Restricciones de diseño:
  RNF_3 El sistema utiliza el lenguaje de programación PHP.
  RNF_4 El sistema utiliza como marco de trabajo ExtJS.
  RNF_5 La interfaz visual debe mantener el estilo y diseño del servidor de almacenamiento NovaNAS.
- ✓ Mantenibilidad:
  RNF_6 Se debe hacer uso de los estándares de codificación definidos para NovaNAS.

Descripción de Historias de Usuario

A continuación, se realiza la descripción de los requisitos funcionales mediante las Historias de Usuario, según la metodología Variación AUP-UCI para los requisitos funcionales Crear disco cifrado y Añadir contraseña, el resto de las historias de usuario se encuentran en el Anexo 2

TABLA 4: HISTORIA DE USUARIO: CREAR DISCO CIFRADO (FUENTE: ELABORACIÓN PROPIA)

HISTORIA DE USUARIOS	
Número: RF_1	**Nombre del requisito:** Crear disco cifrado
Programador: Shadet Asnay Ariosa Aquia	**Iteración Asignada:**1
Prioridad: Alta	**Tiempo Estimado:** 3 semanas
Riesgo en Desarrollo: Alto	**Tiempo Real:** 3 semanas
Descripción: El sistema debe permitir crear un disco cifrado cuando el usuario seleccione la opción “Crear”. Se desplegará un formulario con los siguientes campos: ✓ Dispositivo (obligatorio): es un campo de selección, donde el usuario tendrá la opción de escoger qué disco desea cifrar. ✓ Contraseña: es un campo de texto donde el usuario debe introducir la contraseña para el disco cifrado. ✓ Confirmar contraseña: es un campo de texto donde el usuario debe introducir la contraseña para el disco cifrado. Esta contraseña debe coincidir exactamente con la introducida en el campo “Contraseña”, lo que permite confirmar que se tecleó correctamente. ✓ Key file: es un campo de selección/búsqueda donde el usuario tiene la opción de buscar y seleccionar un archivo para utilizar como contraseña.	
Observaciones: Si no se selecciona un archivo en el campo “Key file” los campos de “Contraseña” y “Confirmar contraseña” son obligatorios. Si el disco cifrado fue creado correctamente debe ser añadido al listado de discos cifrados.	

Prototipo elemental de interfaz gráfica de usuario:

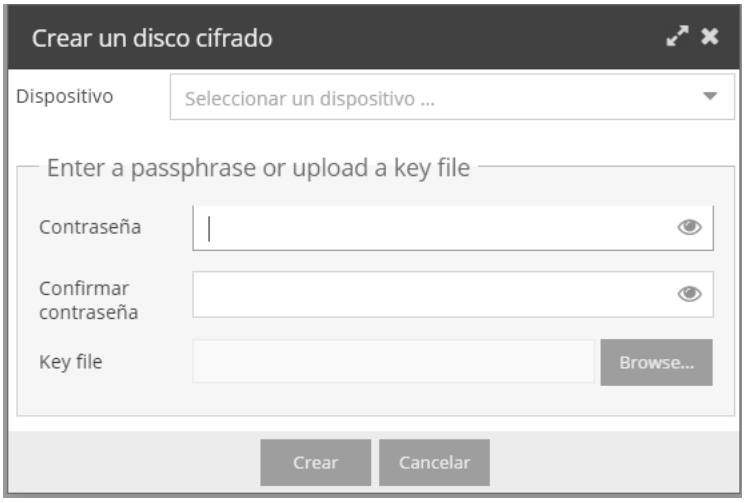


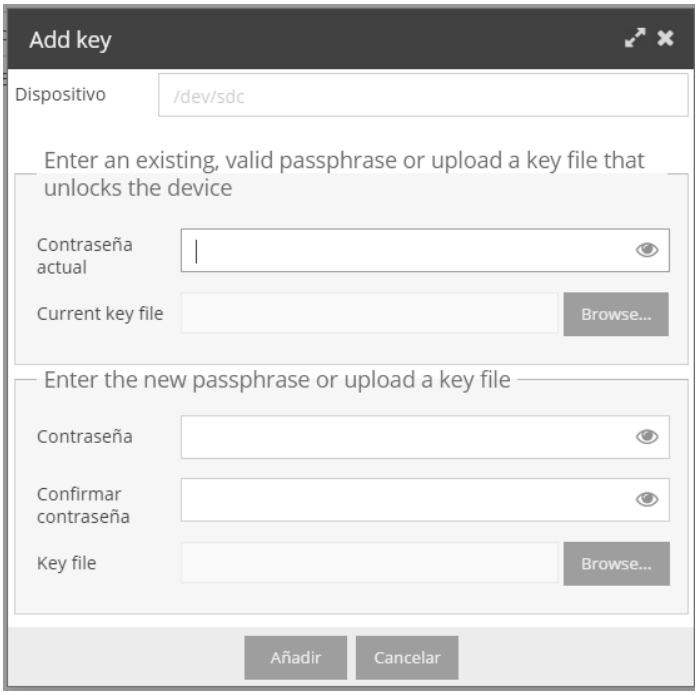
TABLA 5: HISTORIA DE USUARIO: AÑADIR CONTRASEÑA (FUENTE: ELABORACIÓN PROPIA)

HISTORIA DE USUARIOS	
Número: RF_ 7	**Nombre del requisito:** Añadir contraseña
Programador: Shadet Asnay Ariosa Aquia	**Iteración Asignada:**1
Prioridad: Media	**Tiempo Estimado:** 2 semanas
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 1semana
Descripción El sistema debe permitir añadir una o varias contraseñas al disco cifrado (estableciendo un límite de 1 a 8 contraseñas por disco cifrado). Al seleccionar la opción “Añadir contraseña” se desplegará un formulario con los siguientes campos: ✓ Dispositivo (campo obligatorio): es un campo de selección, donde el usuario tendrá la opción de escoger a qué disco le añadirá una contraseña. ✓ Contraseña actual: es un campo de texto, donde el usuario debe introducir la contraseña con la que accede al dispositivo.	

- ✓ Current key file: es un campo de búsqueda, donde el usuario selecciona el fichero de contraseña, en caso de tener alguno.
- ✓ Contraseña: es un campo de texto donde el usuario introduce la contraseña que desea añadir al dispositivo.
- ✓ Key file: es un campo de selección/búsqueda donde el usuario tiene la opción de buscar y seleccionar un archivo para utilizar como contraseña.

Observaciones: Debe existir al menos un disco cifrado.

Prototipo elemental de interfaz gráfica de usuario:



Técnicas de validación de requisitos

Revisiones técnicas formales: los requisitos fueron analizados sistemáticamente por el desarrollador y el cliente en busca de anomalías y/u omisiones.

Técnica de Prototipado: se utilizó para mostrar un modelo ejecutable del sistema a los usuarios finales y los clientes, para analizar si dicho modelo cumple con las necesidades presentadas (ver Anexo 3).

Diseño de casos de prueba: se realizó un caso de prueba para cada requisito, lo que permitió detectar errores o defectos relacionados con las funcionalidades del módulo y para comprobar la veracidad del código.

2.5 Análisis y diseño

El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema. Los objetivos del análisis y diseño son: transformar los requisitos al diseño del futuro sistema, desarrollar una arquitectura para el sistema y adaptar el diseño para que sea consistente con el entorno de implementación (Pressman, 2002).

En este método de análisis y diseño se crea un conjunto de modelos utilizando una notación acordada como, por ejemplo, el Lenguaje Unificado de Modelado (UML). Aplica técnicas de modelado de objetos para diseñar soluciones que mejoren los procesos involucrados y para analizar los requisitos en diferentes contextos, por ejemplo, un sistema de negocio, un conjunto de módulos de software, entre otros (Sommerville, 2011).

Patrón Arquitectónico

Los patrones arquitectónicos son plantillas que describen los principios estructurales globales que construyen las distintas arquitecturas de software viables. Plantean una organización estructural fundamental para un sistema de software, expresando un conjunto de subsistemas predefinidos, especificando responsabilidades y organizando las relaciones entre ellos. La selección de un patrón arquitectónico es además una decisión fundamental de diseño cuando se desarrolla un sistema de software (Rosanigo, 2007).

El patrón de arquitectura en n capas, es el que se aplicará en este diseño ya que brinda una organización jerárquica tal que cada capa proporciona servicios a la capa inmediata superior y se sirve de las prestaciones que le brinda la inmediata inferior. Este patrón es un sub-estilo dentro del estilo llamada y retorno, en el cual las restricciones topológicas del patrón pueden incluir una limitación que exige a cada capa operar solo con la adyacente y a elementos de una capa entenderse sólo con otros elementos de la misma capa (Llorente, 2010).

La propuesta de solución es un módulo que se debe integrar a NovaNAS, este software fue desarrollado utilizando una arquitectura n-capas, en su caso se definieron 2 capas: Presentación y Negocio; en el diseño de la propuesta de solución a desarrollar se definen la capa de Presentación y la capa de Lógica de negocio, por lo que se empleará el patrón en la variante dos capas.

- ✓ Presentación: es la capa diseñada para recibir solicitudes de almacenamiento o recuperación de información desde la capa de lógica de negocio. Es la encargada de presentar al usuario los conceptos de negocio mediante una interfaz de usuario (UI, del inglés *User Interface*), facilitar la explotación de dichos procesos, informar sobre la situación de los procesos de negocio e implementación de las reglas de validación de dicha interfaz. Permite interactuar con la aplicación. Se comunica con la capa de Negocio. Presenta todos los servicios que puede ofrecer el módulo creado desde NovaNAS.
- ✓ Lógica de Negocio: es la capa donde se alojan mayoritariamente las clases. En ella se reciben las peticiones del usuario y se envían las respuestas tras el procesamiento y la comunicación con la capa de Presentación. Debe su nombre a que es aquí donde se establecen todas las reglas del negocio que deben cumplirse. Es donde son procesados los datos. Recibe y procesa peticiones del usuario. Emite respuestas a la capa de Presentación. Tiene asignada la responsabilidad de contener el código necesario para el manejo de datos.

A continuación se representa la arquitectura empleada para el desarrollo de la solución.

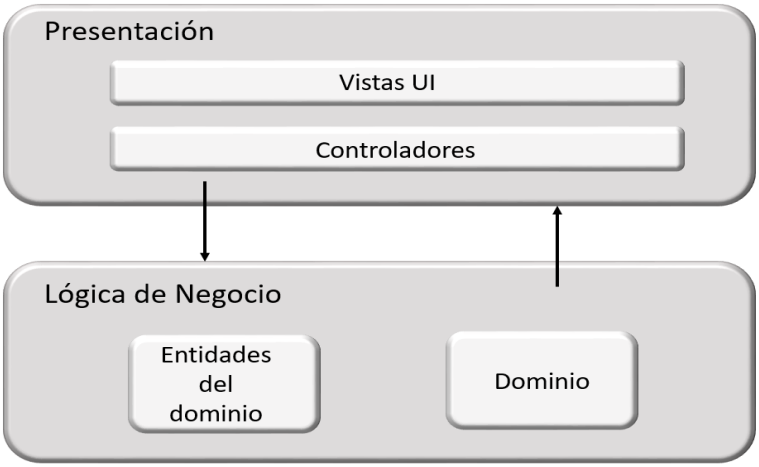


FIGURA 6: REPRESENTACIÓN DE LA ARQUITECTURA 2-CAPAS (FUENTE: (LLORENTE, 2010))

Diagrama de clases del diseño

El diagrama de clases con estereotipos web o diagrama de clases del diseño muestra la especificación para las clases software de la aplicación. Incluye información como clases, asociaciones, atributos, interfaces con sus operaciones y constantes, métodos, dependencias, muestra definiciones de entidades de software más que conceptos del mundo real, se añaden los detalles referentes al lenguaje de programación que se vaya a usar (Larman, 2004).

Para la representación se emplean estereotipos tales como: las *Server Page* las cuales permiten construir y controlar la información de las páginas clientes que serían las *Client Page* y estas muestran información en el cliente manejando los datos del formulario que contiene toda esa información.

Otros de los elementos empleados para la creación del diagrama de clases son las Clases Controladoras, las cuales controlan todo el funcionamiento de la aplicación y las Clases Entidades que contienen todos los datos necesarios de las entidades existentes. A continuación, se representa el diagrama de clases de diseño con estereotipos web realizado. Para la HU ¨Detalles¨ ver Anexo 4.

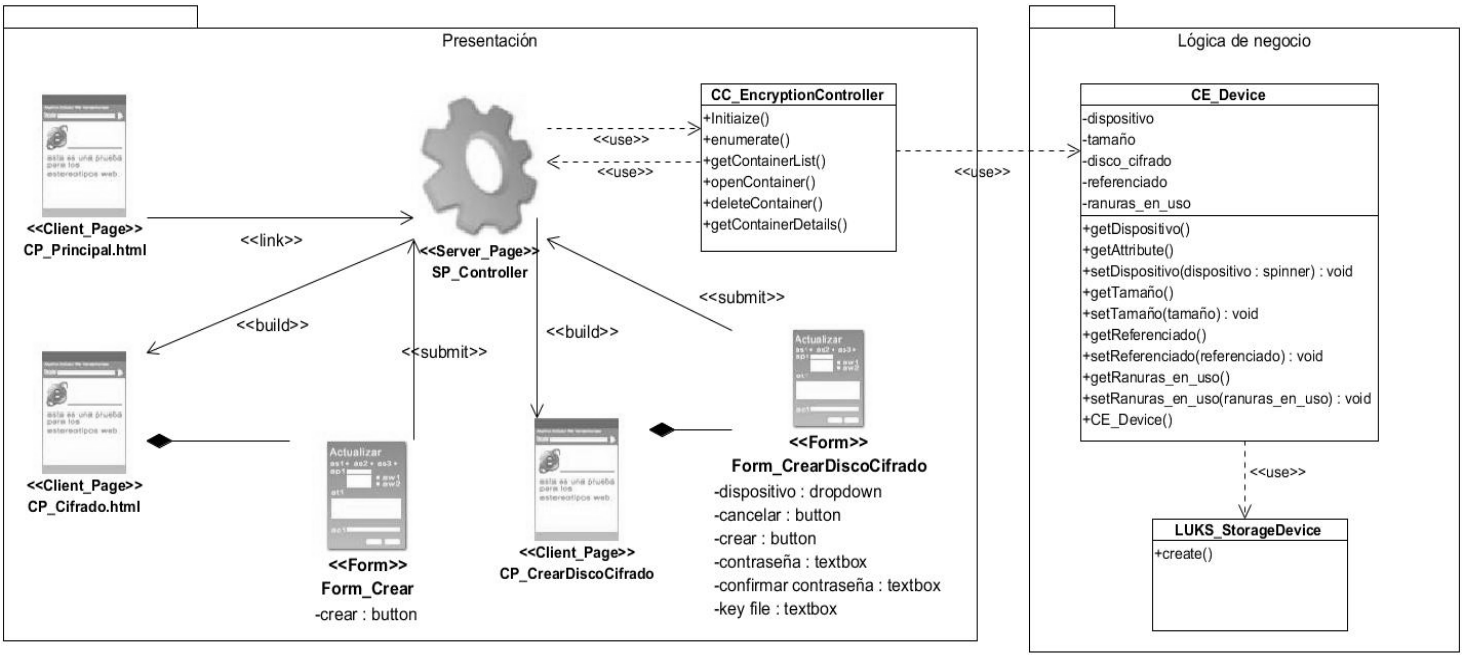


FIGURA 7: DIAGRAMA DE CLASES DEL DISEÑO DE LA HU CREAR DISCO CIFRADO (FUENTE: ELABORACIÓN PROPIA)

Patrones de diseño

Un patrón de diseño provee un esquema para refinar componentes de un sistema de software y la forma en que se relacionan entre sí. Describe una estructura generalmente recurrente de comunicación de componentes que resuelve un problema de diseño general dentro de un contexto particular (Rosanigo, 2007).

Entre sus características se debe mencionar que es reutilizable y aplicable a diferentes problemas de diseño en distintas circunstancias. Es una descripción de un problema y la solución que brinda y que se puede aplicar a nuevos contextos. Para resolver la asignación de responsabilidad en las clases del diseño se hizo uso de los Patrones Generales de Software para asignar Responsabilidades (Rosanigo, 2007).

Patrones GRASP

Experto: este patrón consiste en asignar una responsabilidad determinada a la clase que tenga la mayor cantidad de información para hacer esta tarea; la clase "experta" en información (Larman, 2004). Este patrón es aplicado en la clase *Device* y *Encryption Controller* pues esta última tiene acceso a todas las entidades necesarias y por ello a la información; mientras que ambas manejan y contiene la información necesaria para cumplir con las responsabilidades que le fueron asignadas, convirtiéndose en expertas para esta responsabilidad; facilitando así el entendimiento y contribuir al bajo acoplamiento.

Creador: el patrón asigna a una clase la responsabilidad de crear cuando contiene, agrega, compone, almacena o usa otra clase, lo que brinda una alta posibilidad de reutilizar la clase creadora. Este patrón se evidencia en las clases controladoras que, para cada una de las funcionalidades de la aplicación, son las encargadas de crear las instancias de los objetos que manejan (Larman, 2004). La clase *Encryption Controller* usa directamente las instancias creadas del objeto de tipo *Device,* lo cual le permite obtener la información de estos objetos que maneja para responder a las peticiones de usuario (ver figura 8).

Bajo acoplamiento: Este patrón es una medida de la fuerza con que una clase está conectada a otras, las conoce y recurre a ellas (Larman, 2004). En el diseño de la propuesta de solución existen clases con un bajo acoplamiento debido a que no dependen de muchas otras clases; una clase solo depende de otra, como máximo, para realizar sus funciones.

Alta cohesión: el patrón de alta cohesión asigna responsabilidades de manera que una clase no tenga muchas funcionalidades no relacionadas o no realice un trabajo excesivo (Larman, 2004). Este patrón se evidencia debido a la organización del trabajo en cuanto a la estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión. A cada una de las clases se le asignaron responsabilidades de tal forma, que están estrechamente relacionadas entre sí y no realizan un trabajo excesivo. La clase *Device* es un ejemplo de alta cohesión debido a que tiene la responsabilidad para definir las acciones sobre los dispositivos, esta contiene la información referente a un dispositivo para crearlo, y por tanto es responsable de las acciones de modificación sobre dicho dispositivo (ver figura 9).

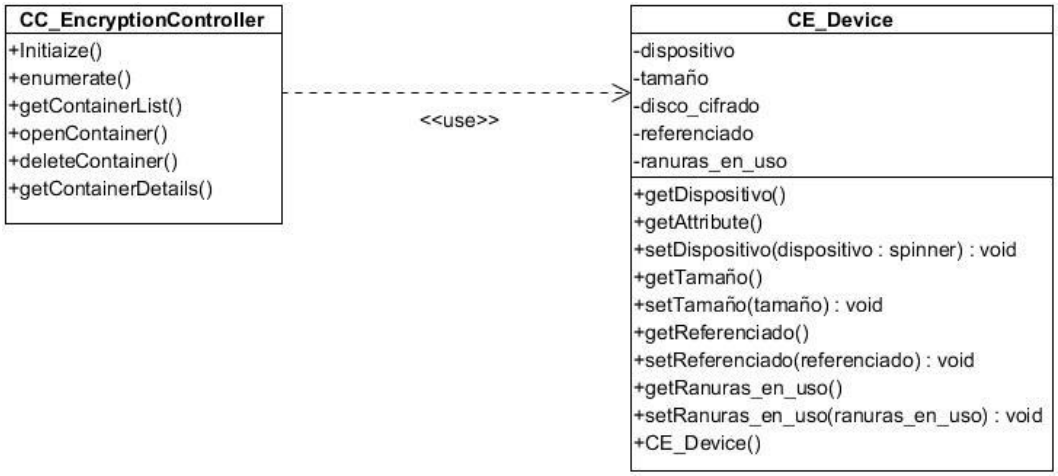


FIGURA 8: PATRÓN CREADOR (FUENTE: ELABORACIÓN PROPIA)

FIGURA 9: PATRÓN EXPERTO (FUENTE: ELABORACIÓN PROPIA)

2.6 Conclusiones del capítulo

Luego de realizar el análisis y diseño del módulo para el cifrado de los datos informáticos almacenados en NovaNAS se puede concluir que:

La entrevista al MSc. Yasiel Pérez Villazón permitió identificar los requisitos funcionales y no funcionales del módulo para el cifrado de los datos informáticos almacenados en NovaNAS y sus descripciones a través de Historias de Usuario lo cual posibilitó una mejor comprensión de todas las especificidades y funcionamiento de dicho módulo.

La creación de los diferentes diagramas y modelos, así como la identificación de los patrones de diseño, permitió una mejor comprensión en cuanto a la estructura del módulo para el cifrado de los datos informáticos almacenados en NovaNAS.

CAPÍTULO III: Implementación y prueba del módulo para el cifrado de los datos informáticos almacenados en NovaNAS

En el presente capítulo se desarrolla una descripción de la implementación y se abordan los temas relacionados con el flujo de trabajo correspondiente a las pruebas. Además se especifican las pruebas que se le realizaron al módulo para el cifrado de los datos informáticos almacenados en NovaNAS, para validar una correcta implementación de los requisitos del sistema y ofrecer un producto de calidad.

3.1 Implementación del sistema

Luego de los resultados del análisis y diseño, se comienza la implementación del módulo, logrando concebir la arquitectura y el sistema como un todo. A continuación, se describen los estándares de codificación que se utilizan en el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS.

Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código el cual refleja un estilo armonioso, como si un único programador hubiese escrito todo el código de una sola vez. Cuando el proyecto de software incorpora código fuente previo, o cuando realiza el mantenimiento de un sistema de software el estándar de codificación debería establecer cómo operar con la base de código existente.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento (Wiegers, 2015).

Reglas a seguir para el trabajo con las clases del módulo

- ✓ Las clases controladoras comienzan con el nombre seguido de la palabra Controller (EncryptionController.php).
- ✓ El código de las clases .php será colocado dentro de la carpeta usr\php\system\storage. El nombre de estas clases comenzará con mayúscula.

- ✓ Se utilizan los comentarios en caso de ser necesario para explicar la utilidad de la clase (ver Figura 10).
- ✓ Los nombres de las clases se escriben con la primera letra de cada palabra que lo compone en mayúscula (ver Figura 10).
- ✓ Los nombres de clases deben ser sustantivos y deben tener la primera letra en mayúsculas. Si el nombre es compuesto, cada palabra componente deberá comenzar con mayúsculas. Los nombres serán simples y descriptivos. Debe evitarse el uso de acrónimos o abreviaturas, salvo en aquellos casos en los que dicha abreviatura sea más utilizada que la palabra que representa (URL, HTTP, entre otras.).

```php
<?php
/** @license   http://www.gnu.org/licenses/gpl.html GPL Version 3 ...*/
namespace NOVANAS\System\Storage;

/**
 * Class for handling an open LUKS container,
 * i.e. from the decrypted device point of view.
 */
class StorageDeviceLuks extends StorageDeviceDM {

    public function getLuksEncryptedDeviceFile() {
        if(FALSE === ($slaves = $this->getSlaves()))
            return FALSE;
        if(count($slaves)!=1)
            return FALSE;
        return $slaves[0];  // Should only be one slave, just return the first
    }

```

FIGURA 10: REGLAS PARA EL TRABAJO CON CLASES (FUENTES: ELABORACIÓN PROPIA))

Reglas a seguir para nombrar las variables y métodos

- ✓ Los nombres deben ser descriptivos y concisos. No usar grandes frases y solo abreviaturas pequeñas.
- ✓ Las variables se escribirán siempre en minúsculas. Las variables compuestas tendrán la primera letra de cada palabra componente en mayúsculas.
- ✓ El identificador para las variables y los parámetros serán con letras en minúsculas y en caso de ser un nombre compuesto se divide cada palabra con mayúscula inicial (ver Figura 11).

```
43      protected $uuid         = "";
44      protected $isOpen       = FALSE;
45      protected $headerInfo   = "";
46      protected $usedKeySlots = 0;
47      protected $freeKeySlots = 8;
48
49      protected $deviceMapperDeviceFile = "";
50      protected $deviceMapperName       = "";
51
52      private $dataCached = FALSE;
53
```

FIGURA 11: REGLAS PARA LAS VARIABLES (FUENTE: ELABORACIÓN PROPIA)

- ✓ Los métodos deben ser verbos escritos en minúsculas. Cuando el método esté compuesto por varias palabras cada una de ellas tendrá la primera letra en mayúsculas (ver Figura 12).

```
328     public function getDescription() {
329         if ($this->getData() === FALSE)
330             return FALSE;
331         return sprintf(gettext("LUKS encrypted device %s[%s, %s]"),
332              args: ($this->getModel()) ? '('.$this->getModel().') ' : '',
333             $this->getDeviceFile(),
334             binary_format($this->getSize()));
335     }
336
337
338     public function create($key, $keyIsFile=FALSE) {
339         if (TRUE === $keyIsFile) {
340             $cmd = sprintf( format: "cryptsetup luksFormat -q %s %s",
341                 escapeshellarg($this->getDeviceFile()),
342                 escapeshellarg($key));
343         } else {
344             $cmd = sprintf( format: "echo -n %s | cryptsetup luksFormat -q %s",
345                 escapeshellarg($key),
346                 escapeshellarg($this->getDeviceFile()));
347         }
348         $process = new Process($cmd);
349         $process->setRedirect2to1();
350         $process->execute($output);
351         $this->refresh();
352         return TRUE;
353     }
```

FIGURA 12: REGLAS PARA DECLARAR LOS MÉTODOS (FUENTES: ELABORACIÓN PROPIA)

Reglas a seguir para la división de líneas

Cuando una expresión ocupe más de una línea, esta se podrá romper o dividir en función de los siguientes criterios:

- ✓ Tras una coma.
- ✓ Antes de un operador.
- ✓ Se recomienda las rupturas de nivel superior a las de nivel inferior.
- ✓ Alinear la nueva línea con el inicio de la expresión al mismo nivel que la línea anterior.

Ejemplos:

```
unMetodo(expresionLarga1, expresionLarga 2, expresionLarga 3,

 expresionLarga 4, expresionLarga 5);

if ((condicion1 && condicion2)

|| (condicion3 && condicion4)

||!(condicion5 && condicion6)) {unMetodo(); }
```

Reglas a seguir para la declaración de clases / interfaces

Durante el desarrollo de clases / interfaces se deben seguir las siguientes reglas de formateo:

- ✓ No incluir ningún espacio entre el nombre del método y el paréntesis inicial del listado de parámetros.
- ✓ El caracter inicio de bloque ("{") debe aparecer al final de la línea que contiene la sentencia de declaración.
- ✓ El caracter fin de bloque ("}") se sitúa en una nueva línea tabulada al mismo nivel que su correspondiente sentencia de inicio de bloque, excepto cuando la sentencia sea nula, en tal caso se situará detrás de "{".
- ✓ Los métodos se separarán entre sí mediante una línea en blanco.

```
class ClaseEjemplo extends OtraClase {
```

```
 $variable1;

 $variable2;

public function __construct() {

$variable1 = 0;
```

3.2 Pruebas de software

La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión de las especificaciones, del diseño y de la codificación (Pressman, 2002). Para la validación del módulo de cifrado de los datos informáticos almacenados en NovaNAS se utilizaron pruebas de caja blanca y caja negra, para lograr como resultado que disminuya el número de errores existentes en los sistemas y por ende una mayor calidad.

El proceso de validación del módulo para el cifrado de los datos informáticos almacenados en NovaNAS se realiza mediante la estrategia de pruebas. La estrategia de pruebas de software proporciona un mapa que describe los pasos que se darán para realizarlas, indica cuándo se planea y se darán dichos pasos, además cuánto tiempo, esfuerzo y recursos consumirán. Un software se prueba para descubrir los errores cometidos, si se realiza sin ningún plan seguramente se desperdicia tiempo y esfuerzo.

- ✓ **Pruebas de integración**: Abordan los conflictos asociados con los problemas duales de verificación y construcción de programas. Durante la integración, se usan más las técnicas de diseño de casos de prueba que se enfocan en entradas y salidas (Pressman, 2002).

Resultados de las pruebas de integración

Las pruebas de integración comprueban que los componentes funcionen como un todo, que son llamados correctamente y transfieren los datos correctos en el tiempo preciso a través de sus interfaces. Al realizar la integración del módulo para el cifrado de los datos informáticos almacenados en NovaNAS este debe aparecer en la barra lateral izquierda específicamente en la sección correspondiente al almacenamiento (ver Figura 13). Al seleccionar las opciones brindadas por el módulo para el cifrado este debe responder correctamente mostrando los mensajes en correspondencia con las acciones realizadas.

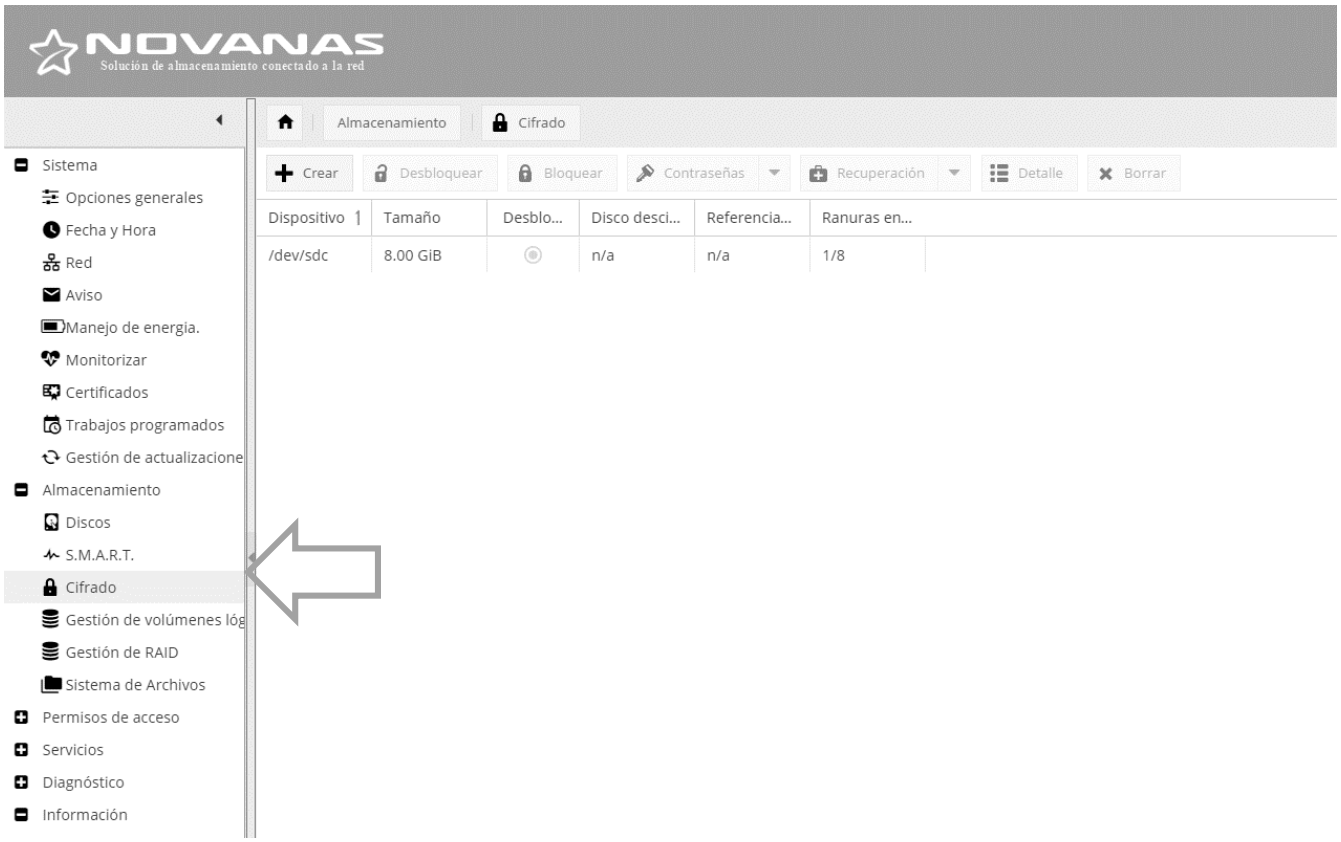


FIGURA 13: INTEGRACIÓN DEL MÓDULO AL SISTEMA (FUENTE: ELABORACIÓN PROPIA)

Teniendo en cuenta la arquitectura 2-Capas empleada, para la correcta integración del módulo a la arquitectura utilizada en NovaNAS y que dicho módulo debe incorporarse a una herramienta base; se emplea una estrategia de integración ascendente. En esta los componentes se integran de abajo hacia arriba. Las pruebas realizadas al módulo para el cifrado de los datos informáticos almacenados en NovaNAS demostraron una integración exitosa. Permitiendo un correcto funcionamiento para el cifrado de los datos informáticos almacenados.

- ✓ **Prueba funcional:** Las pruebas de caja negra o funcional tienen el objetivo de verificar el correcto manejo de las funciones externas provistas o soportadas por el software y que el comportamiento observado se corresponda con las especificaciones del producto y a las expectativas del usuario (Pressman, 2002).

Resultados de la prueba funcional

Esta prueba será realizada para encontrar errores de funciones incorrectas o ausentes, errores en accesos a bases de datos externas, errores de interfaz, errores de rendimiento (Pressman, 2002), entre otras; sin prestar atención al código. El objetivo es probar todos los requisitos de modo que la entrada sea aceptada de forma adecuada y ofreciendo una salida correcta.

A continuación se aplica la técnica de partición de equivalencia para realizar la validación. Esta técnica de prueba de caja negra consiste en dividir el dominio de entrada de un programa en un número finito de variables de equivalencia. Se definen dos tipos de variables de equivalencia, las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real de dato.

Las celdas de la tabla contienen valores válidos (V), valores inválidos (I), y valores no necesarios (N/A) que representan valores no relevantes para el caso de prueba.

TABLA 6: CASO DE PRUEBA PARA EL ESCENARIO " CREAR DISCO CIFRADO"

(FUENTE: ELABORACIÓN PROPIA)

ID del escenario	Escenario	Descripción	Variable 1	Variable 2	Variable 3	Respuesta del sistema	Flujo central
EC 1	Crear disco cifrado con datos válidos	El sistema debe permitir crear disco cifrado en caso de que no existan datos incorrectos	V	V	V	Añade el disco cifrado a la lista de discos	1- El usuario selecciona la opción *Crear* de la interfaz principal del módulo. 2- El usuario rellena los campos de forma correcta.

							3- El usuario da click en el botón *Crear* de la interfaz Crear disco cifrado
EC 2	Crear disco cifrado con datos inválidos	El sistema permite verificar si hay datos incorrectos	V	V	I	El sistema señala en color rojo los campos incorrectos	1- El usuario selecciona la opción *Crear* de la interfaz principal del módulo. 2- El usuario no rellena los campos de forma correcta. 3- El usuario da click en el botón crear de la interfaz Crear disco cifrado.

La siguiente tabla muestra las variables empleadas en el caso de prueba “Crear disco cifrado”.

TABLA 7: VARIABLES EMPLEADAS EN EL CASO DE PRUEBA "CREAR DISCO CIFRADO"

(FUENTE: ELABORACIÓN PROPIA)

No	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Dispositivo	Campo de selección	No	Contiene los discos disponibles, a los cuales se les puede realizar el cifrado
2	Contraseña	Campo de texto	No	Contiene caracteres

				alfanuméricos y especiales
3	Confirmar contraseña	Campo de texto	No	Contiene caracteres alfanuméricos y especiales

Para el proceso de esta prueba se llevaron a cabo 3 iteraciones. En la primera iteración se detectaron trece (13) no conformidades, de ellas cinco (5) corresponden a errores ortográficos y de idioma, cuatro (4) estaban relacionadas con las validaciones incorrectas y otras cuatro (4) no conformidades correspondientes a errores relacionados con la interfaz. Durante el resto de las iteraciones se solucionaron la mayoría de las no conformidades quedando un mínimo de tres (3) no conformidades relacionadas con errores de idiomas en la tercera iteración.

- ✓ **Pruebas de aceptación**: Son especificadas por el cliente y se centran en las características y funcionalidades generales del sistema que son visibles y revisables por parte del cliente. Las pruebas de aceptación se derivan de las Historias de usuario que se han implementado como parte de la liberación del software (Pressman, 2002).

Resultados de las pruebas de aceptación

El objetivo de esta prueba es encontrar los errores y las omisiones en la definición de los requisitos del sistema, debido a que los datos reales ejercitan el sistema en diferentes formas a partir de datos de prueba (Pressman, 2002). A continuación se muestran los casos de prueba realizados por el cliente a algunas Historias de Usuario (HU).

TABLA 8: CASO DE PRUEBA DE ACEPTACIÓN PARA LA HISTORIA DE USUARIO "CREAR DISCO CIFRADO"

(FUENTE: ELABORACIÓN PROPIA)

Caso de prueba de Aceptación
Nombre de la historia de usuarios: Crear disco cifrado
Nombre de la persona que realiza la prueba: Yasiel Perez Villazón

Descripción de la prueba: El sistema permite crear un disco cifrado cuando el usuario selecciona la opción Crear.
Condiciones de ejecución: El usuario debe estar autenticado en la plataforma NovaNAS.
Entrada/Pasos de ejecución: 1. Acceder al módulo de cifrado. 2. Seleccionar la opción Crear.
Resultado esperado: El sistema crea un disco cifrado a selección del usuario, disco que será adicionado al listado de discos cifrados.
Evaluación de la prueba: Satisfactoria

TABLA 9: CASO DE PRUEBA DE ACEPTACIÓN PARA LA HISTORIA DE USUARIO "AÑADIR CONTRASEÑA"

(FUENTE: ELABORACIÓN PROPIA)

Caso de prueba de aceptación
Nombre de la historia de usuario: Añadir contraseña
Nombre de la persona que realiza la prueba: Yasiel Perez Villazón
Descripción de la prueba: El sistema permite adicionar una nueva contraseña para acceder al disco cifrado-
Condiciones de ejecución: El usuario debe estar autenticado en la plataforma NovaNAS. Debe existir al menos un disco cifrado.
Entrada/Pasos de ejecución: 1. Acceder al módulo 2. Seleccionar la opción Añadir contraseña
Resultado esperado: El sistema adiciona una contraseña para acceder al disco cifrado
Evaluación de la prueba: Satisfactoria

Como resultado de las pruebas se obtuvo una evaluación satisfactoria por parte del cliente. Lo que representa el cumplimiento de los objetivos propuestos para esta prueba.

3.3 Evaluación del objetivo de la investigación

Esta investigación se desarrolla con el objetivo de ofrecer una solución a los problemas de confidencialidad de los datos informáticos almacenados en NovaNAS, proponiendo un aumento de la confidencialidad a través de mecanismos de cifrado de datos. En este epígrafe se valida la solución a través de procedimientos matemáticos y la comparación de magnitudes conocidas, que permitirán obtener información numérica para medir los valores de confidencialidad. Dando a conocer en qué medida se cumple o no el objetivo propuesto en esta investigación.

Las medidas de confidencialidad se utilizan para evaluar el grado en que un producto o sistema asegura que los datos son accesibles sólo para aquellos autorizados a tener acceso (NC, 2016). Existen diferentes funciones que permiten obtener una valoración numérica en cuanto al grado de confidencialidad en determinadas situaciones; para medir la confidencialidad en esta investigación, se emplean las medidas establecidas en la Norma Cubana ISO/IEC 25023. A continuación se muestra una tabla con estas medidas de confidencialidad.

TABLA 10: MEDIDAS DE CONFIDENCIALIDAD (FUENTE:(NC, 2016))

ID	Nombre	Descripción	Función de medición
SCo-1-G	**Control de acceso**	¿Qué proporción de datos confidenciales están protegidos contra accesos no autorizados?	$X = 1 - A/B$ A = Número de elementos de datos confidenciales a los que se puede acceder sin autorización B = Número de elementos de datos que requieren control de acceso
SCo-2-G	**Corrección de encriptación de datos**	¿Qué tan correcto es el encriptado / desencriptado de los elementos de datos implementados como se indica en la	$X = A/B$ A = Número de elementos de datos encriptado / desencriptado correctamente B = Número de elementos de datos que requieren encriptado / desencriptado

ID	Nombre	Descripción	Función de medición
		especificación de requisito?	
NOTA: Para obtener más detalles sobre la calidad de los datos relacionados, consulte Cnf-I-1 en ISO/IEC 25024.			
SCo-3-S	**Fortaleza de los algoritmos criptográficos**	¿Qué proporción de algoritmos criptográficos ha sido bien examinada?	$X = 1 - A/B$ A = Número de algoritmos criptográficos rotos o inaceptablemente riesgosos en uso B = Número de algoritmos criptográficos utilizados
NOTA 1: Es importante seleccionar un algoritmo bien evaluado que actualmente se considera fuerte por los expertos en el campo y seleccionar implementaciones bien probadas. Como con algunos mecanismos criptográficos, el código fuente tiene que estar disponible para el análisis. Por ejemplo, los sistemas del gobierno de los Estados Unidos requieren la certificación FIPS 140-2. NOTA 2: Existen otras maneras de medir la fortaleza de los algoritmos criptográficos, por ejemplo, utilizando la piratería ética.			

Durante el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS se utilizó una herramienta para el cifrado de dispositivos de bloques, dicha herramienta es la encargada de llevar a cabo el proceso de cifrado. De esta manera los factores relacionados con el cifrado y con los algoritmos que esta emplea están fuera de la acción del desarrollador (autora), esto significa que el desarrollo de un algoritmo de cifrado no forma parte de la especificación de los requisitos. Además el algoritmo AES, utilizado por defecto por la herramienta, fue parte del proceso de estandarización internacional (Joan Daemen, 2002). Por lo tanto, la medida que tiene en cuenta la corrección de encriptación de datos no formará parte de la evaluación del objetivo de esta investigación.

En cuanto a la medida referente a la fortaleza de los algoritmos criptográficos que posee como una de las variables el número de algoritmos "rotos"[20] , en el caso del algoritmo AES de llave 128 bits para considerar un ataque que rompa este algoritmo es necesario realizar operaciones de 2^{120} , operación que por ahora sería un ataque irrealizable (Joan Daemen, 2002).

Debido a que estas medidas ya han sido tomadas en cuenta en otras investigaciones a nivel internacional y con la intervención de varios expertos en el tema, para realizar la evaluación del objetivo de la investigación se selecciona entonces, la medida de confidencialidad definida para el control de acceso. En esta medida cuando se hace referencia al número de elementos de datos confidenciales a los que se puede acceder sin autorización correspondiente a la variable *A* de la función de medición, los elementos a tener en cuenta fueron: documentos clasificados, códigos fuente, expedientes de proyecto, datos personales, información financiera, entre otros documentos de importancia que se encontraban en el sistema en el momento en que se realizó la medición.

En el caso de la variable *B,* se refiere a los elementos que requieren control de acceso, es decir, al total de datos confidenciales que se desea que solo sean accedidos por usuarios autorizados. En cuanto al valor del resultado, este se encuentra entre cero (0) y uno (1). Normalmente, cuanto más cerca a uno (1), es mejor el valor. El mejor valor posible que se puede obtener es uno (1) y el peor valor posible es cero (0).

Para poner en práctica este método se selecciona una muestra, y se calculan los valores de confidencialidad antes y después de implementar el módulo para el cifrado de los datos informáticos almacenados en NovaNAS, comprobando si existe un aumento o no de estos valores. La siguiente tabla muestra los resultados de la evaluación del objetivo general de la investigación.

TABLA 11: RESULTADOS DE LA EVALUACIÓN DEL OBJETIVO GENERAL (FUENTE: ELABORACIÓN PROPIA)

Función de medición	Sin el módulo para el cifrado	Con el módulo para el cifrado
$X = 1 - A/B$	$X = 1 - 5/5$	$X = 1 - 0/5$
Resultados (X)	0	1

[20] En el contexto criptográfico se considera "roto" un algoritmo si existe algún ataque más rápido que una búsqueda exhaustiva (ataque por fuerza bruta).

Los resultados de la evaluación a través de las medidas de confidencialidad permiten afirmar el cumplimiento del objetivo propuesto con el desarrollo del módulo para el cifrado de datos informáticos almacenados en NovaNAS. Analizando los resultados obtenidos luego de aplicar la función de medición antes y después de la implementación del módulo para el cifrado de datos informáticos almacenados en NovaNAS se puede concluir que existe un aumento de la confidencialidad. Esto convierte a dicho módulo en una solución viable y correcta para aumentar la confidencialidad de los datos informáticos almacenados en esta plataforma.

3.4 Conclusiones del capítulo

El desarrollo de este capítulo permitió desarrollar un código reutilizable y de fácil comprensión porque fueron definidos los estándares de codificación que se emplearían durante el desarrollo de la propuesta de solución. Una vez diseñadas, documentadas y ejecutadas las pruebas; se detectaron las no conformidades, lo que permitió verificar la calidad del producto, además se realizó la evaluación del objetivo de la investigación, lo que permitió verificar su correcto cumplimiento.

Conclusiones generales

Luego de la investigación para el desarrollo del módulo para el cifrado de los datos informáticos almacenados en NovaNAS, se arriban a las siguientes conclusiones:

- ✓ El análisis de la situación problemática existente y el estudio de las soluciones para el cifrado de datos informáticos, demostró la necesidad de desarrollar un módulo para el cifrado de los datos informáticos almacenados en NovaNAS.
- ✓ La correcta selección de la metodología, las herramientas y tecnologías para el desarrollo de la solución, permitió la implementación del módulo para el cifrado de datos informáticos almacenados en NovaNAS.
- ✓ Los casos de pruebas descritos y efectuados permitieron detectar de manera temprana las no conformidades, posibilitando la corrección de las mismas con resultados satisfactorios, y contribuyendo a la calidad del producto final.
- ✓ La evaluación del objetivo general de la investigación, a través del cálculo de la variación de la confidencialidad según los indicadores definidos por la Norma Cubana ISO/IEC 25023, permitió verificar el cumplimiento del objetivo de la investigación.

Recomendaciones

Para contribuir a la continuidad de la investigación, se realiza la siguiente recomendación:

- ✓ Realizar un estudio para conocer de qué manera se gestionaría el acceso al disco cifrado en caso de que fallara un sector del disco duro, teniendo en cuenta el funcionamiento de la herramienta en cuanto al almacenamiento de la llave maestra.

Referencias

ALEGSA. (2018). *Diccionario de informática y tecnologías*. In. Retrieved from http://www.alegsa.com.ar/Diccionario/diccionario.php

CryFS, D. (2015). CryFS: Comparison of encrypted cloud storage solutions. www.cryfs.org

Ecryptfs, D. (Producer). (2017). Como cifrar archivos en linux con Ecryptfs. Retrieved from www.ecryptfs.org

Encfsmp, D. (Producer). (2010). Cifrado de archivos con Encfsmp. Retrieved from https://encfsmp.sourceforge.io

Galicia, R. B. (2010). Análisis de algoritmos criptográficos y su aplicación al cifrado de archivos. In. México D.F, México.

Halcrow, M. A. (2016). eCryptfs: An Enterprise-class Cryptographic Filesystem for Linux.

J. Callas, L. D., H. Finney, and R. Thayer. (1998). RFC 2440. Retrieved from https://www.ietf.org/rfc/rfc2440.txt

Joan Daemen, V. R. (2002). *The design of Rijndael: AES- The advanced Encryption*.

Larman, C. (2004). *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*

Llorente, C. d. l. T. (2010). *Guía de arquitectura N-capas*.

Loshin, P. (2013). *Simple Steps to Data Encryption*. MA, USA.

NC ISO/IEC 25023:2017 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality, (2016).

Perez, R. (Producer). (2009). Phpstorm 8 con soporte drupal. Retrieved from http: //inusual. com/artículos/phpstorm-8-con-soporte-para-drupal-8-y-wordpress/.

Pradhan, G. a. C., Bingxue (Producer). (2015). *Centralized storage of storage system resource data using a directory server*. Retrieved from https://patents.google.com/patent/US9037532B1/en

Pressman, R. S. (2002). *Ingeniería de software: Un enfoque práctico 5ta edicion* (M. Hill Ed. 5ta ed.). Nueva York, Estados Unidos.

Quiroz (Producer). (2012). Retrieved from https://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integradoide/.

Rosales, M. E. (2014). *Extensión de la herramienta Visual Paradigm para la generación de clases de acceso a datos con Doctrine*. La Habana, Cuba.

Rosanigo, Z. B. (2007). *Arquitectura de software: Estilos y Patrones*.

Rouse, M. (2017). ¿Qué es Almacenamiento de archivos? SearchDataCenter en Español. Retrieved from http://searchdatacenter.techtarget.com/es/definicion/Almacenamiento-de-archivos

Metodología de desarrollo para la Actividad productiva de la UCI, (2014).

SNIA. (2017). The Storage Management Initiative (SMI) | SNIA. *Advancing Storage and Information Technology* Retrieved from https://www.snia.org/forums/smi

Sommerville, I. (2011). *Software Engineering 9na edition* (9na ed.). EEUU.

Soto, E. (2015). Desarrollo web. Retrieved from http://www.desarrolloweb.com

Villazón, Y. P. (2018). *Solución para la gestión del almacenamiento de datos en las instituciones cubanas.* Cuba

Wiegers, K. E. (2015). Software Requirements 2: Practical Tchniques for gathering and managing requirements throughout the product development cycle.

Anexos

Anexo 1: Entrevista realizada al MSc. Yasiel Pérez Villazón

Estimado(a) compañero(a):

La presente entrevista tiene como objetivo conocer, analizar y determinar las debilidades que presenta NovaNAS respecto a su funcionamiento y el tratamiento de la confidencialidad de los datos informáticos almacenados en éste. El resultado de esta entrevista resulta de vital importancia para la presente investigación debido a que contribuirá a la correcta elaboración de la propuesta de solución.

Datos generales del entrevistado:

Título universitario: _____

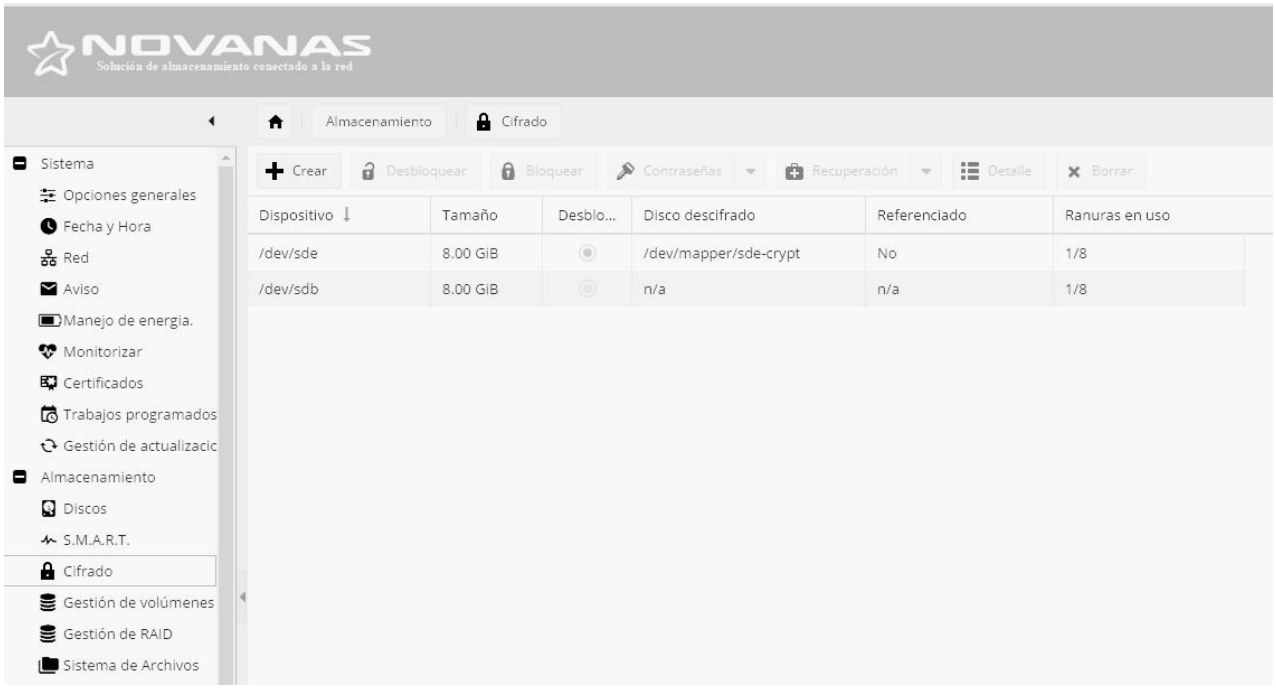
Categoría científica: _____

Categoría docente: _____

Instrucciones:

1. ¿A qué tipo de clientes (usuarios) va dirigido el producto NovaNAS? ¿En qué entorno es desplegado NovaNAS?
   _____
2. ¿Qué servicios ofrece NovaNAS?
   _____
   _____
3. ¿Existe algún mecanismo para restringir el acceso al servidor por parte de los usuarios?
   _____
4. ¿Existe algún mecanismo de seguridad para proteger la información? ¿Cuáles?
   _____
   _____
5. ¿Existe algún mecanismo para tratar la confidencialidad de los datos informáticos almacenados en NovaNAS?
   _____
   _____
6. ¿Qué otras características considera que deba presentar el sistema, en cuanto a la usabilidad, seguridad, interfaz u otro aspecto que garantice su calidad?
   _____

Anexo 2: ***Historia de usuarios: Mostrar listado de discos cifrados***

HISTORIA DE USUARIOS	
Número: RF_1	**Nombre del requisito:** Mostrar listado de discos cifrados
Programador: Shadet Asnay Ariosa Aquia	**Iteración Asignada:**1
Prioridad: Media	**Tiempo Estimado:** 1,3 semanas
Riesgo en Desarrollo: Medio	**Tiempo Real:** 2 semanas
Descripción: El sistema debe mostrar un listado de los discos cifrados con la información adicional correspondiente a cada disco como: tamaño, el estado del disco (bloqueado o desbloqueado), dirección del disco cifrado, si tiene referencia (si, no, n/a), y las ranuras en uso.	
Observaciones: N/A	
Prototipo elemental de interfaz gráfica de usuario: 	

Anexo 3: Interfaz de usuario

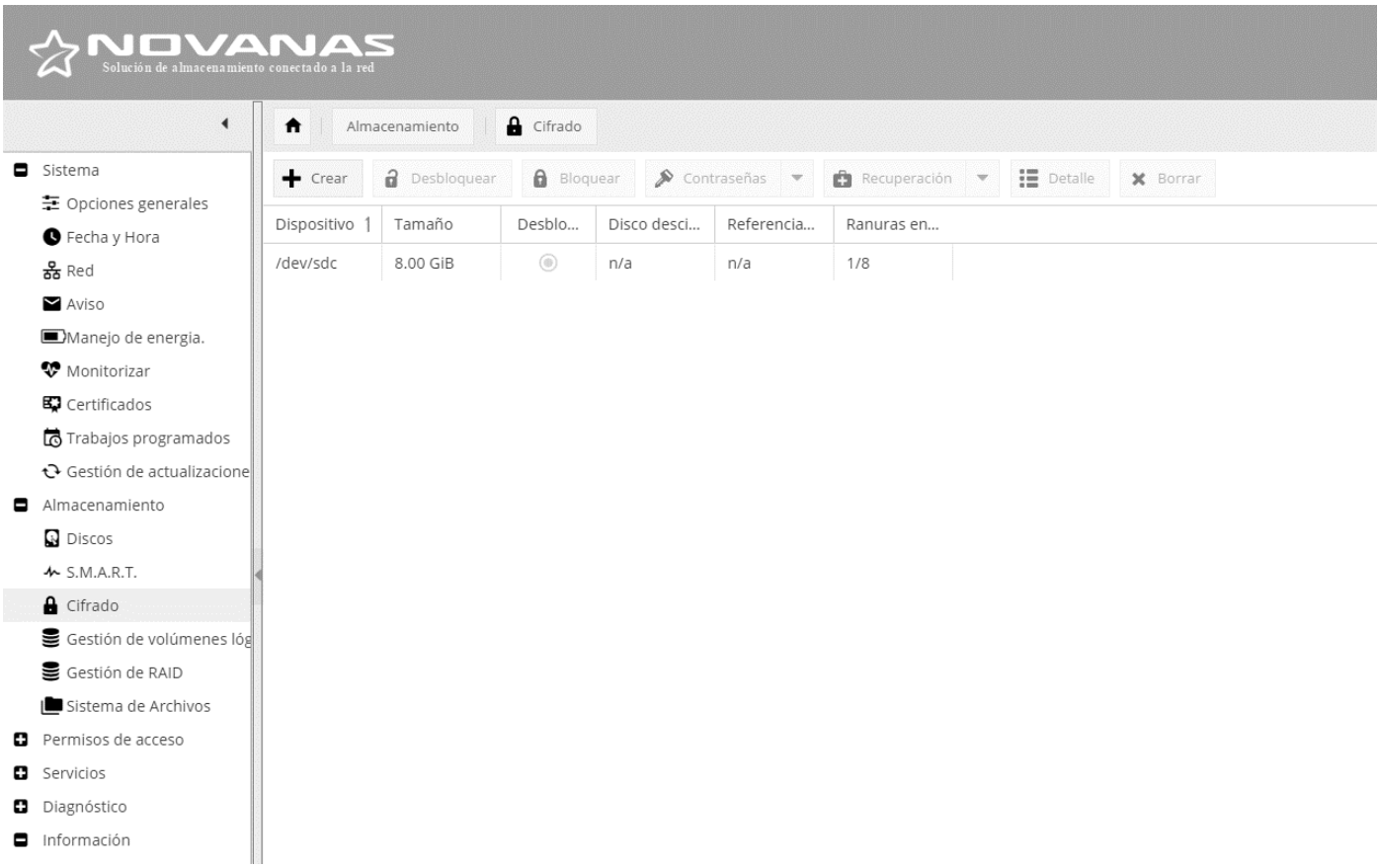


FIGURA 14: INTERFAZ PRINCIPAL DEL MÓDULO (FUENTE: ELABORACIÓN PROPIA)

Anexo 4 Diagrama de clases del diseño

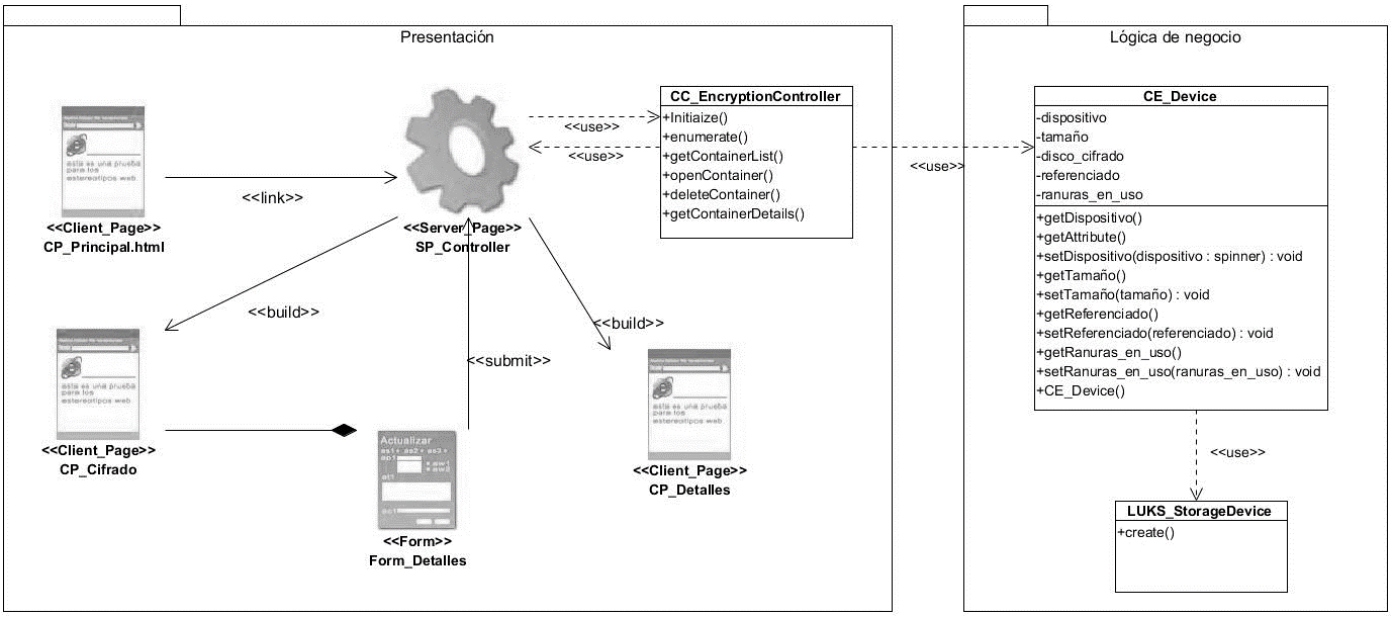


FIGURA 15: DIAGRAMA DE CLASES DE DISEÑO DE LA HU DETALLES (FUENTE: ELABORACIÓN PROPIA)

Anexo 5: Acta de aceptación

Acta de aceptación de productos de trabajo

UCI
Universidad de las Ciencias Informáticas

ACTA DE ACEPTACIÓN DE PRODUCTOS DE TRABAJO

En cumplimiento del **Convenio de colaboración** establecido entre el **Centro de Software Libre (CESOL)** y la estudiante **Shadet Asnay Ariosa Aquia** de la Facultad 1 de la Universidad de las Ciencias Informáticas y en función de la ejecución del proyecto: **Módulo para el cifrado de los datos informáticos almacenados en NovaNAS**, se hace entrega del producto que se relaciona a continuación:

- Módulo para el cifrado de los datos informáticos almacenados en NovaNAS

La parte Cliente, luego de haber revisado el producto de trabajo relacionado anteriormente procede a firmar la aceptación de los mismos en total conformidad.

Entrega	Recibe
Nombre y Apellidos: Shadet Ariosa Aquia	**Nombre y Apellidos:** Hanny Valdes Hernández
Cargo: Estudiante Facultad 1	**Cargo:** Jefe de Departamento
Firma:	**Firma:**

Fecha: 29 de mayo de 2019