TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE ACAPULCO

CRIPTOANÁLISIS A LA FUNCIÓN HASH DE UN SISTEMA OPERATIVO NIVEL C1 EMPLEANDO UN CLÚSTER HPC CON SOFTWARE DE USO LIBRE

TESIS

QUE PARA OBTENER EL TÍTULO DE:
MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA
ING. GADDIEL FREDY FLORES ARTEAGA

DIRECTOR
DR. EDUARDO DE LA CRUZ GÁMEZ

CODIRECTOR
M.T.I. ELOY CADENA MENDOZA

ACAPULCO, GRO., Noviembre de 2019.

El presente trabajo de tesis fue desarrollado en la *División de Estudios de Posgrado e Investigación* del *Instituto Tecnológico de Acapulco*, perteneciente al Programa Nacional de Posgrados de Calidad (PNPC-CONACYT).

Con domicilio para recibir y oír notificaciones en Av. Instituto Tecnológico de Acapulco s/n, Crucero del Cayaco, Acapulco, Guerrero, México. C.P. 39905.

Becario:	Gaddiel Fredy Flores Arteaga.
CVU:	851420.
Núm. de apoyo:	627006.
Grado:	Maestría

Descargo de Responsabilidad Institucional

El que suscribe declara que el presente documento titulado “Criptoanálisis a la función hash de un sistema operativo nivel C1 empleando un clúster HPC con software de uso libre” es un trabajo propio y original, el cuál no ha sido utilizado anteriormente en institución alguna para propósitos de evaluación, publicación y/o obtención de algún grado académico.

Además, se han recogido todas las fuentes de información utilizadas, las cuales han sido citadas en la sección de referencias bibliográfica de este trabajo.

Nombre: Ing. Gaddiel Fredy Flores Arteaga _____ Fecha y firma

Dedicatoria

Este logro se lo dedico a mi toda mi familia, que incondicionalmente siempre está ahí cuando los necesito. A mis Padres, especialmente dedicado a mi Madre que es un ejemplo de esfuerzo para mí, sabe el camino complicado que he tenido que pasar para poder llegar a este gratificante momento, y siempre ha tenido una palabra que me alienta a no desmallar y seguir adelante.

Dedicado también para mi hermano Uriel, porque sin saberlo y a la distancia su apoyo y confianza han sido un pilar fundamental para poder llegar a este momento.

Adrielito, hijo, te dedico este logro por tu paciencia, porque sé que has tenido que pasar tiempo con Papá en la escuela en lugar de estar en casa descansando o jugando después de salir de tus clases, gracias por soportarlo, te amo y espero ser un ejemplo de esfuerzo y perseverancia para ti.

Los amo a todos.

Agradecimientos

Agradezco primero y ante todo a Dios, todo se lo debo a Él, por derramar sus bendiciones sobre mí, por acompañarme en cada paso que doy y haber fortalecido mi corazón e iluminado mi mente. Gracias Padre.

A mi familia, que ha estado ahí, siempre a mi lado en los momentos más oscuros y difíciles creyendo en mí, impulsándome para alcanzar mis metas.

Al Consejo Nacional de Ciencia y Tecnología por haberme apoyado económicamente para la realización de este proyecto.

A mi director de tesis, el Dr. Eduardo de la Cruz Gámez por su tiempo, por su paciencia, por los valiosos aportes, críticas, comentarios y sugerencias que realizó durante este proyecto.

Al Departamento de Posgrado e Investigación del Instituto Tecnológico de Acapulco, así como a toda la plantilla docente, de la cual me llevo conocimiento y experiencias enriquecedoras que sin lugar a dudas marcaron mi vida profesional.

A la segunda generación, muchas gracias Abejorros, sin el apoyo mutuo durante el tiempo que pasamos en las aulas y durante el desarrollo de los proyectos, tal vez no lo hubiéramos logrado.

Resumen

El presente trabajo inicia con una breve introducción sobre la importancia que tiene la seguridad en cualquier aspecto de la vida y sobre todo en el área laboral, específicamente en el área informática. Posteriormente resalta cómo la criptografía ha estado presente en momentos importantes de la vida del ser humano, sobre todo dentro del ámbito militar, en la cual, ha sido fundamental para permitir comunicaciones seguras. Actualmente la criptografía está dentro de nuestro diario vivir cuando realizamos transacciones en internet de manera segura. Como parte de la investigación en este trabajo se describen los tipos de criptografía y criptoanálisis, también se analizan los niveles de seguridad de sistemas operativos, así como los métodos de agrupamiento de computadoras, para implementar con la ayuda de esta herramienta de una manera más rápida criptoanálisis a un sistema operativo con un nivel de seguridad C1.

Palabras clave: clúster, seguridad, criptoanálisis.

Abstract

This work begins with a review of the importance of security in any aspect of life and especially in the workplace, specifically in the computer area. Later this work speaks briefly about how cryptography has been present at important moments in the life of the human being, especially in the military field, in which, it has been fundamental to allow secure communications. Currently, cryptography is inside our daily life when we conduct transactions on the internet in a secure manner. As part of the research in this work, the types of cryptography and cryptanalysis are described, as well as the security levels of operating systems and, above all, the main methods of computers clustering to implement a system with a faster method of cryptanalysis in an operative system with a security level C1.

Keywords: cluster, security, cryptanalysis.

Índice de Contenido

Índice de Figuras

Índice de Tablas

Capítulo 1. Introducción

1.1 Antecedentes del problema

Sin lugar a duda una de las herramientas más importantes producida en el siglo XX ha sido la computadora, la cual ha provocado cambios vertiginosos en la sociedad y en el progreso de la misma. En la actualidad, el entorno está prácticamente controlado por las nuevas tecnologías, que a medida que transcurre el tiempo avanzan sin límites y en ocasiones son utilizadas incorrectamente provocando daños de grandes dimensiones.

Los trascendentales cambios operados en el mundo moderno, caracterizados por su constante desarrollo, la acelerada globalización de la economía, la acentuada dependencia a almacenar grandes volúmenes de información y los sistemas que la proveen, aunado a grandes inversiones de las empresas en conocimiento actual y futuro, en sistemas de información, además del potencial que poseen las tecnologías para cambiar drásticamente las organizaciones y las prácticas de negocio, así como crear nuevas oportunidades, diseñar nuevas estructuras tecnológicas basadas en información la convierte en un activo codiciable. Tales avances hacen permanente el riesgo en las tecnologías y nos llevan a pensar en la problemática sobre la seguridad informática.

La protección de datos, documentos y control de acceso a la información es un tema que cada día toma más fuerza en las grandes compañías, debido a las diferentes prácticas de especialistas tecnológicos como los son los *hackers* y *crackers* que roban información vital para posteriormente sacar provecho de ella.

“Desde la aparición de los grandes sistemas aislados hasta nuestros días, en los que el trabajo en red es lo habitual, los problemas derivados de la seguridad de la información han ido también cambiando, evolucionando, pero están ahí y las soluciones han tenido que ir adaptándose a los nuevos requerimientos técnicos. Aumenta la sofisticación en el ataque y ello aumenta la complejidad de la solución, pero la esencia es la misma” (Mifsud, 2012)

Dentro del mundo actual la información es denominada un activo importante de una institución la cual debe ser protegida para evitar su pérdida, modificación o el uso inadecuado de su contenido.

El extravío o el mal uso de información que para alguien es confidencial genera daños y repercusiones directas relacionadas a la integridad, disponibilidad y confidencialidad de los archivos y a su vez a los dueños de dicha información.

“La seguridad informática se distingue por tener dos propósitos de seguridad, la Seguridad de la Información y la Protección de Datos, estos se diferencian debido a que los datos son valores numéricos que soportan la información mientras que la información es aquello que tiene un significado para nosotros” (González Agudelo, 2014)

La seguridad informática consiste en asegurar que los recursos del sistema de información (material informático o programas) de una organización sean utilizados de la manera que se decidió

inicialmente y que el acceso a la información allí contenida, así como su modificación sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización.

Actualmente hay mucha información circulando en internet, por ejemplo, correos electrónicos, publicaciones en *Twitter*, *Facebook*, videos, fotos, comercio electrónico, secretos de estado, militares o industriales. Lamentablemente dicha red es pública e insegura, dentro de la cual existen usuarios bien intencionados y muchos mal intencionados. Diariamente encontramos noticias de sitios con intrusiones mal intencionadas (hackeados), clonación de tarjeta de crédito, o imágenes adulteradas. Todo lo anterior motiva la necesidad de proteger la información a través de mecanismos que impidan a los usuarios no autorizados acceder a dicha información.

1.2 Planteamiento del Problema

Los sistemas de información actuales están relacionados completamente con la exposición de su información a entidades externas que son ajenas a la organización o delegación a la que pertenecen.

Cuando no se conoce el alcance que tienen los sistemas informáticos, ni lo vulnerables que pueden ser, si son expuestos a Internet por los mismos empleados de la organización, puede darse "fuga" de información hacia el exterior; esto los hace vulnerables a eventos y amenazas que conllevan a riesgos que pueden provocar pérdidas tangibles e intangibles, las cuales se traducen en pérdidas que en muchos de los casos por decir lo menos, serán monetarias.

El portal español de estadísticas *statista*, muestra la cuota de mercado de los principales sistemas operativos a nivel mundial a fecha de febrero de 2017, según el número de instalaciones (statista, 2017). Windows 10 fue el segundo sistema operativo más instalado, con una cuota de mercado de 25.3%. Esto significa que alrededor de una de cada cuatro computadoras está gobernado por este sistema operativo. Por su parte, en el caso de Windows 7, el porcentaje se sitúa por encima del 45% a pesar de que ya no se vende ningún PC con este sistema operativo. Hablando de manera conjunta los sistemas operativos Windows cuentan con una cuota del mercado global de 88.57 %.

La empresa de seguridad informática Británica Avecto informó en su Microsoft *Vulnerabilities Report 2017* (Avecto, 2017) (reporte 2017 de vulnerabilidades de Windows), un aumento de vulnerabilidades publicadas del 111% en los últimos 5 años, de las cuales 235 críticas

fueron mitigadas removiendo permisos de administrador, así como otras muy importantes enfocadas al Administrador de Cuentas de Seguridad (SAM, Security Account Manager, por sus siglas en inglés), la cual es una base de datos almacenada como un archivo del registro en Windows y que almacena las contraseñas de los usuarios en un formato *hash* (Algoritmos que crean a partir de una entrada una salida alfanumérica de longitud fija).

1.3 Objetivos

1.3.1 Objetivo General. A través de la implementación de un procedimiento de fuerza bruta, utilizando para ello un clúster de alto rendimiento, así como software de uso libre, lograr capacidad de procesamiento capaz de vulnerar un sistema de autenticación nivel C1.

1.3.2 Objetivos Específicos

- Analizar los diferentes niveles de seguridad de sistemas operativos.

- Analizar diferentes métodos de criptoanálisis para vulnerar el Administrador de cuentas de seguridad.

- Estudiar las técnicas de agrupamiento de computadoras (clustering) para la obtención de mayor poder de procesamiento.

- Implementar un clúster de alto rendimiento en el Laboratorio de la Maestría en Sistemas Computacionales del Instituto Tecnológico de Acapulco para ejecutar operaciones de cómputo distribuido.

1.4 Justificación

Antes la seguridad de la información se realizaba a través de medios físicos, por ejemplo, una caja fuerte, en la cual las personas resguardaban objetos o información valiosa para ellos. Con la introducción de las computadoras, y el uso de internet, los usuarios comienzan a almacenar información importante dentro de sus equipos por lo que se considera indispensable el uso de herramientas que brinden protección a la información que se encuentra resguardada dentro de éstas. Los usuarios informáticos suelen enviar información por la red, en las empresas al realizar trámites, o enviar informes a socios, jefes, entre otros.

El resguardo de información siempre ha sido importante para los seres humanos, con la diferencia que anteriormente no se contaba con los avances tecnológicos que tenemos actualmente, puesto que, con la introducción de internet y el uso de dispositivos electrónicos, se tiene mayor riesgo de la información que en ellos se maneja.

La protección del contenido de los datos es una necesidad urgente en casi todos los ámbitos de la sociedad, incluyendo los servicios públicos y privados, por lo que la tecnología de la seguridad de los sistemas de cifrado se encuentra de frente a verdaderos retos.

1.5 Hipótesis

A través de un de fuerza bruta como técnica de criptoanálisis y con la ayuda de una mayor capacidad de procesamiento mediante la implementación de agrupamiento de computadoras (clúster) de alto rendimiento, será posible encontrar alguna vulnerabilidad dentro del sistema de administración de cuentas de seguridad de Windows server 2012.

1.6 Alcances y Limitaciones

Esta tesis se centrará en el sistema operativo Windows como objeto de estudio, al cual se le realizará el criptoanálisis.

La tesis se enfocará en el método Beowulf para realizar el agrupamiento de computadoras, así como la distribución de software libre del sistema operativo Linux Rocks.

El número de equipos disponibles para poder realizar el agrupamiento es de 17 computadoras, hasta lograr el poder de cómputo de 800 *Gflops*.

1.7 Impacto Social, Tecnológico, Económico y Ambiental

La sostenibilidad del presente proyecto será abordada a partir de los siguientes aspectos:

1.7.1 Impacto social. A veces se puede pensar que un clúster de alto rendimiento pudiera no tener algún impacto directo en el medio social, pero si nos enfocamos a un punto de vista en donde gracias a la capacidad de procesamiento de estas máquinas, han sido parte fundamental en el desarrollo de proyectos que inciden de manera directa en beneficio de la sociedad en la que vivimos, como lo es la medicina, la física, química.

Este proyecto busca beneficiar a nuestra sociedad proveyendo su capacidad de procesamiento en la búsqueda de alguna vulnerabilidad en el sistema operativo más utilizado hoy en día, como lo es Windows, además de que estará disponible para la comunidad científica y

estudiantil del Instituto Tecnológico de Acapulco para optimizar tiempo de cálculo en futuras investigaciones.

1.7.2 Impacto ambiental. *Green IT*, también conocido como *Green Computing*, o traducido al español como Tecnologías Informáticas Verdes, es definido como el estudio y la práctica de diseñar, desarrollar, usar y disponer de computadoras, servidores y diferentes periféricos tales como monitores, impresoras, dispositivos de almacenamiento, de redes y comunicaciones eficientemente y efectivamente con un impacto mínimo o nulo en el medio ambiente. (Murugesan, 2008)

Bajo la premisa anterior, este proyecto hace uso de los equipos con los que cuenta actualmente el laboratorio de la Maestría en Sistemas Computacionales, lo cual evitará la adquisición de equipo nuevo y por consecuencia desechar el equipo existente, evitando desperdicio electrónico. Dicho equipo cuenta con un procesador Intel I3 de 4ta. generación lo cual según el portal electrónico *Europa press*:

“Esta nueva generación supone el "mayor salto" en cuanto ahorro energético de la historia de Intel, así como una mejora en los gráficos y nuevas experiencias de usuario”. (europa press, 2013)

Teniendo como base estos dos puntos importantes como lo son la reutilización de equipo, y utilizando procesadores de bajo consumo energético, es como este proyecto busca contribuir de manera positiva con el medio ambiente.

1.7.3 Impacto económico. Actualmente existe una gran necesidad de procesar grandes cantidades de datos lo cual hace imprescindible disponer de máquinas con estas características, sin embargo, su alta complejidad, aunado al gran costo, hace que en instituciones de carácter público sea muy complicado conseguir recurso para este fin. Sin embargo, un clúster de alto rendimiento reutilizará equipo propio evitando así gastos por concepto de adquisición de equipo, siendo, de esta manera viable para instituciones de carácter público.

1.8 Metodología a Utilizar.

1.8.1 Análisis. Después de analizar las diferentes metodologías de clustering se optó por desarrollar e implementar un clúster de alto rendimiento (HPC).

1.8.2 Diseño.

1.8.2.1 Economía. Debido a las características del equipo con el cual cuenta el Laboratorios de la Maestría en Sistemas Computacionales, los componentes utilizados para este proyecto son considerados de uso común y de bajo costo, lo cual los hace fácilmente reemplazables en caso de fallo a un costo accesible. El equipo es de la marca Acer Aspire modelos AXC-605-M021, lo cual nos hace tener ventaja y respaldo del fabricante con respecto a hardware genérico.

1.8.2.2 Instalaciones. Un clúster requiere de un ambiente controlado. Esto es, una habitación especial con sistema de enfriamiento, capacidad suficiente de carga eléctrica, control de humedad y un ambiente libre de polvo. Además, se deben considerar el peso del equipo sobre el piso (ya sea falso o piso normal) Se recomienda colocarlo en los sótanos o en la plata baja de edificios. (Rocha Quezada, Botello Rionda, Vargas Félix, & Munguía Torres, 2011)

En el laboratorio de la Maestría en Sistemas Computacionales se cuenta con espacio que reunía la mayoría de las características requeridas para este proyecto.

1.8.2.3 Escalabilidad. Un clúster debe ser fácilmente escalable y a su vez confiable. Por tal motivo se ha utilizado equipo de marca, lo cual nos proporciona más de confiabilidad en su desempeño y a su vez facilidad de aumentar el número de equipos si en un futuro fuese necesario agrandar este proyecto. Se armó un clúster de cinco computadoras con procesadores Intel Core I3 de 64 bits en cada nodo. La memoria incluida y periféricos son fáciles de adquirir en el mercado.

1.8.2.4 Red. En la parte de redes se utilizó un switch fast Ethernet con una velocidad de 1 Gbps, el cual tiene tazas de transferencia rápida y el costo es notablemente más económico que uno de fibra óptica.

El cableado seleccionado fue el estándar UTP de cobre con la norma Gigabit Ethernet de IEEE 802.3ab con la categoría de cableado 6e que soporta tazas de transferencia de datos de 1000Base T (1 Gbps).

1.8.2.5 Software.

"El sistema operativo debe constar con servicios dedicados para compartir y respaldar archivos, así como capacidad de acceder a los datos de trabajo rápidamente. El sistema operativo debe también proveer servicios de monitoreo y reporte de fallas en el sistema." (Rocha Quezada, Botello Rionda, Vargas Félix, & Munguía Torres, 2011)

Existe una gran variedad de software para instalar clúster de alto rendimiento, sin embargo, como se ha comentado con anterioridad el costo de adquisición debe ser accesible, por tal motivo se optó por una distribución con licencia GNU de Linux, Rocks Cluster, la cual es una distribución sencilla de instalar y configurar en comparación con otras distribuciones analizadas, dicha distribución está basada en CentOS que, a su vez, está basada en la conocida distribución de Linux, Red Hat.

1.8.2.6 Almacenamiento. Para este proyecto no se requerirá un gran espacio de almacenamiento, por tal motivo los equipos utilizados, tanto maestro, como esclavos, mantuvieron sus unidades de disco originales, las cuales son de 1 TB de almacenamiento.

1.8.3 Implementación. La implementación del clúster de alto rendimiento fue realizada dentro de las instalaciones del Laboratorio de la Maestría en Sistemas Computacionales del Instituto Tecnológico de Acapulco, construido inicialmente para este proyecto de criptoanálisis, lo cual requerirá suficiente capacidad de procesamiento para poder implementar ataques de fuerza bruta a sistemas operativos. Las características del clúster construido son:

1.8.3.1 Software. El software instalado en cada uno de los nodos es el siguiente:

- Sistema operativo: Rocks Clúster Linux 7.4 (64 bits)
- Librería de manejo de mensajes: Open-MPI 1.5.3
- Software de monitoreo Ganglia

1.8.3.2 Hardware. El clúster está compuesto de 17 equipos con las siguientes características:

• Nodo Maestro y 16 Nodos Esclavos:

- Procesador: Intel Core I3 4150 (4 núcleos), Memoria: 6 GB, Disco Duro: SATA 1 TB, tarjeta de Red Gigabit Ethernet

El hardware adicional es el siguiente:

• Interconexión de Red

• Switch Cisco: 26 puertos, 1Gbps, modelo SG200. Red interna.

1.8.3.3 Recursos totales del clúster.

• Núcleos de unidades de procesamiento (cores): 68

• Capacidad en memoria: 92 GB

• Capacidad en disco: 17 TB

1.9 Aseguramiento Técnico – Material

A continuación, se muestra el desglose del equipo utilizado para la elaboración de un clúster de alto rendimiento, estos componentes consisten en *Hardware y Software* especializado, así como los costos de producción y personal involucrados en la metodología y elaboración del proyecto.

- Costos de *Hardware*.
- Costos de *Software*.

1.9.1 Hardware. En este punto se refleja el costo de los dispositivos y herramientas a nivel físico las cuales sirvieron para la implementación del proyecto clúster de alto rendimiento. Cabe resaltar que el equipo utilizado forma parte del inventario actual del Laboratorio, motivo por el cual, su utilización no causó erogación alguna, sin embargo, la investigación de costos se realizó con la finalidad que el lector pueda establecer comparativas y buscar alternativas en caso de ajuste de presupuesto.

1.9.1.1 Equipo de cómputo. El equipo de cómputo utilizado consta de diecisiete computadoras las cuales servirán, una para Nodo Maestro y las dieciséis restantes, como Nodos Esclavos.

La descripción de estos equipos se muestra en la tabla a continuación, mostrando los costos de todos los equipos utilizados dentro del proyecto

Tabla 1.1. Costo del Proyecto

Modelo	Cantidad	Características	Costos
PC - Acer AXC-605	17	Procesador: Intel Core i3. Pantalla: 15,6”. Disco duro: 1TB HDD RAM: 6GB Salida HDMI. Tarjeta de gráficos INTEGRADA Lector de CD	Los costos en el mercado de una PC de escritorio con estas características tiene un valor aproximado de $6,000.00 pesos (Moneda nacional Mexicana).
Switch Cisco Gigabit Ethernet	1	Modelo SG200 de 26 puertos	Costo en el mercado, alrededor de $20,000.00 pesos M.N.
Total			122,000.00 pesos M.N.

Capítulo 2. Estado del Arte

Título del Trabajo: Introducción a la Criptografía

Autor: Gibrán Granados Paredes

Fecha: Julio 2010

Lugar: Ciudad de México

Publicado en: Coordinación de Publicaciones Digitales. DGSCA-UNAM

http://www.revista.unam.mx/vol.7/num7/art55/int55.htm

Síntesis: Cuando se desea tener seguridad hablando en ámbitos informáticos, la criptografía es una herramienta muy útil; el autor lo entiende también como un medio para garantizar las propiedades de confidencialidad, integridad y disponibilidad de los recursos de un sistema.

Con la criptografía se puede garantizar las propiedades de integridad y confidencialidad, pero hay que saber cómo utilizarla, para ello es importante tener claros los conceptos básicos que están detrás de los sistemas criptográficos modernos. Estos conceptos van desde entender qué es la criptografía, cómo está clasificada, entender el funcionamiento básico de algunos sistemas de cifrado y conocer cómo se forman los documentos digitales como firmas y sobres digitales.

Resultados: El autor realiza un viaje por un gran número de conceptos utilizados en criptografía, el inicio y la clasificación de la misma, así como la criptografía clásica y moderna y analiza de manera breve algunos métodos de cifrado más comunes

Título del Trabajo: Criptoanálisis sobre métodos clásicos de cifrado

Autores: Sebastián Gómez, Juan David Arias, Diego Agudelo

Fecha: 2 de mayo 2013

Lugar: Ciudad de México

Publicado en: Scientia et Technica Año XVII, No 50, abril de 2012.

Síntesis: Los algoritmos criptográficos utilizados desde la antigua Roma hasta nuestros días, son métodos que convierten un mensaje de texto plano a cifrado, al proceso inverso se le conoce como descifrar. Este artículo realiza un estudio de algunas técnicas de encriptación clásicas del siglo pasado como lo son el cifrado de Cesar y el cifrado de Vigenére, así como muestra técnicas básicas y modernas de criptoanálisis basadas en la teoría de la información y estadística como lo son la entropía y el ataque de máxima correlación. Las técnicas de criptoanálisis descritas anteriormente pueden ser usadas en otros métodos de cifrado siempre y cuando estén presentes las mismas debilidades.

Resultados: Los autores implementaron en lenguajes de programación Python un algoritmo para el cálculo de entropía para diferentes longitudes de clave. Para la implementación de los algoritmos de cifrado y descifrado de Vigenére y de máxima correlación utilizaron lenguaje de programación C++. Se probaron los métodos anteriormente descritos sobre textos planos de diferentes longitudes logrando un resultado de 93.85 % de efectividad en promedio.

Conclusión: Este artículo aporta para mi investigación el conocimiento de dos de los algoritmos de encriptación más antiguos y básicos, pero sobre todo que conociendo los métodos de criptoanálisis me permitirán analizar debilidades y fortalezas de algoritmos.

Título del Trabajo: ULTRACOM: Computación de Alto Rendimiento para Criptoanálisis

Autor: Antonio Castro Lechtaler, Alejandro Repetto, Martín Bianchi, Marcelo Cipriano, Alejandro Arroyo Arzubi, César Cicerchia, Eduardo Malvacio

Fecha: octubre 2014

Lugar: Buenos Aires, Argentina.

Publicado en: XX Congreso Argentino de Ciencias de la Computación (Buenos Aires, 2014)

Síntesis: La generación de números primos, fundamentales para la producción de claves, el análisis de secuencias pseudo-aleatorias, los procesos de verificación y validación de algoritmos de seguridad, en un sentido genérico, requieren enormes cantidades de procesamiento. Los procesos relacionados con la matemática criptográfica se caracterizan por la alta necesidad de cómputo. El objetivo de ULTRACOM es desarrollar una computación distribuida multipropósito que permita, como primera aplicación real, el análisis, la validación y la ejecución de pruebas de estrés sobre sistemas criptográficos. El presente trabajo de investigación detalla el proceso de desarrollo de ULTRACOM y muestra un ejemplo de su implementación en una versión Beta. Para ello, toman como entrada dos versiones del algoritmo Trivium. Utilizando la misma infraestructura, y sin desarrollo extra, se prueban grandes volúmenes de claves y vectores de inicialización de los algoritmos en busca de secuencias débiles, de menor longitud que la de la buscada.

Resultados: El equipo implementó una versión beta de la plataforma ULTRACOM, comprobando la factibilidad y viabilidad técnica de la generalización de la infraestructura de computación distribuida en grilla (malla) BOINC. Este avance, permite extender los horizontes de BOINC y habilita la posibilidad de obtener la capacidad de cómputo de alto rendimiento a bajo

costo. Concluyeron que ULTRACOM es un proyecto viable, escalable y de alto impacto para la evaluación y validación de algoritmos criptográficos.

Conclusión: Este artículo aporta para mi investigación el conocimiento y la implementación de una arquitectura de computación distribuida y una vez siendo factible su desarrollo, utilizar su poder de cómputo atacando a través de un algoritmo un sistema criptográfico.

Título del Trabajo: Criptografía Cuántica

Autor: M. Baig

Fecha: 2010

Lugar: Barcelona, España.

Síntesis: En este artículo el autor realiza una breve introducción a la criptografía clásica y la relación que existe con la teoría de la información de Shannon, la cual asegura que el cifrado digital simétrico es secreto y perfecto siempre y cuando se cumplan los tres requisitos:

1. La llave ha de ser aleatoria

2. La llave debe usarse sólo una vez

3. La llave ha de ser tan larga como el mensaje y de un solo uso, debe de estar en posesión tanto del emisor como del receptor.

El punto 3 es el más complicado de cumplir debido a la forma de compartir la llave entre el emisor y receptor.

Más adelante en el artículo se hace un análisis sobre otra técnica de criptografía que es el cifrado digital asimétrico en la que se estudia cómo funcionan las llaves privadas y públicas, así como el método de criptográfico RSA

Resultados:

Conclusión: Este artículo aporta para mi investigación el conocimiento más profundo de los métodos de criptografía simétricos y asimétricos, específicamente el algoritmo asimétrico RSA.

Título del Trabajo: La Criptografía Maderista en la Revolución Mexicana (1910-1911) Criptoanálisis de una carta cifrada por Gabriel Leyva Solano

Autor: Roberto Narváez

Fecha: 2011

Lugar: México, DF.

Síntesis: Este trabajo aborda temas relacionados con las comunicaciones secretas del movimiento revolucionario comandado por Francisco I. Madero a partir de 1910. Se compone de dos partes: la primera es una breve reseña crítica de ejemplos criptográficos del maderismo que fueron generados entre 1910 y 1911, la segunda describe paso a paso el criptoanálisis que aplicó el autor Roberto Narváez para desencriptar una carta parcialmente cifrada de Gabriel Leyva Solano a Francisco I. Madero.

Resultados: El propósito general último es contribuir al conocimiento de la criptografía mexicana en el siglo XX. En lo particular se trata de poner a disposición de los estudiosos el contenido completo de la carta de Leyva Solano, por primera vez después de 100 años, y al mismo tiempo someter a la crítica técnica e histórica el procedimiento criptoanalítico puesto en operación para recuperar el texto plano.

Conclusión: Este artículo aporta para mi investigación el conocimiento de la técnicas y estudios realizados en nuestro país sobre formas y métodos de encriptación durante la Revolución Mexicana; incluso mucho antes de la 2ª. Guerra mundial, momento donde se registró un auge importante dentro de la criptografía moderna.

Título de Tesis: Análisis de la programación concurrente sobre la cpu y gpu en el desarrollo de fractal build.

Autor: José Vicente Anilema Guadalupe

Fecha: 2012

Lugar: Riobamba, Ecuador.

Síntesis: Este trabajo aborda temas relacionados con la geometría euclidiana debido a que el autor realiza una aplicación que crea paisajes fractales, lo que significa crear un paisaje con una misma figura geométrica. Lo anterior se traduce en una necesidad de gran poder de cómputo para realizar de una manera eficaz la tarea. Dentro de la tesis el autor realiza diversos estudios comparativos entre el procesamiento serializado realizado por una CPU (unidad central de procesamiento) y procesamiento paralelizado realizado por una GPU (Unidad grafica de procesamiento) a través de diversas herramientas, para obtener el mejor aprovechamiento de los recursos.

Resultados: El autor concluye que es necesario examinar los algoritmos a utilizar para que estos puedan ser utilizados en cómputo paralelo. Además de que es necesario aprender ciertas herramientas a utilizar para aprovechar al máximo el cómputo distribuido y no desperdiciar el poder de procesamiento muy superior que las GPU's nos pueden brindar.

Conclusión: Este artículo aporta para mi investigación el conocimiento de la técnicas y estudios realizados para medir y comparar la velocidad de procesamiento de la unidad central de procesamiento (CPU) y la unidad gráfica de procesamiento (GPU). Así como la medida en Flops (Operaciones de punto flotante por segundo).

Título del Trabajo: Arquitectura clúster de alto rendimiento utilizando herramientas de software libre.

Autores: Leonardo Chiquiguanca, Edyson Malla, Freddy Ajila, Rene Guamán-Quinché

Fecha: mayo, 2015

Lugar: Loja, Ecuador.

Síntesis: Este trabajo aborda temas relacionados con la gran cantidad de datos que tienen que procesar las universidades públicas dentro de sus investigaciones realizadas en diferentes áreas como la ingeniería, medicina, etc. y la carencia de recursos económicos para obtener supercomputadoras que procesen dicha información. Los autores proponen dentro de su investigación, la utilización de equipo de cómputo existente dentro de su universidad y por medio de software libre realizar agrupamiento de computadoras para poder lograr el procesamiento de grandes volúmenes de datos de una manera accesible.

Resultados: Los autores concluyen que la mejor arquitectura para el agrupamiento (clúster) de computadoras en ambientes universitarios es la arquitectura Beowulf, debido a que la implementaron en hardware convencional existente dentro de su campus universitario y que además usaron software libre, mencionando que el poder de cómputo logrado fue muy cercano al de una supercomputadora.

Conclusión: Este artículo aporta para mi investigación, el conocimiento de las técnicas utilizadas en ambientes universitarios para la realización de agrupamiento de computadoras y sobre todo la optimización de costos para realizar dicha tarea, y tener como finalidad el lograr mayor capacidad de procesamiento.

Título de Tesis: Desarrollo de un algoritmo de cifrado simétrico de resumen

Autores: Jonathan Dave Orjuela Navarrete

Año: 2008

Lugar: Bogotá, D.C.

Síntesis: Esta tesis realiza un estudio de varios métodos de criptografía simétrica, para posteriormente, utilizando diferentes fórmulas matemáticas crear un algoritmo criptográfico nuevo. Dentro de la tesis utiliza dos métodos para realizar pruebas a su algoritmo; criptoanálisis diferencial y criptoanálisis basado en fallos de hardware. Dentro de los cuales hace un estudio detallado de las técnicas y herramientas de criptoanálisis realizado a su algoritmo.

Resultados: El autor concluye de manera positiva con la realización del algoritmo y de manera satisfactoria con la realización de criptoanálisis a su algoritmo, pudiendo demostrar la fortaleza del algoritmo a través de ejemplo de cifrado con textos claro muy parecido y su conversión a cadenas completamente diferentes.

Conclusión: Este artículo aporta para mi investigación, el conocimiento a fondo de los métodos de criptoanálisis diferencial y criptoanálisis basado en fallos de hardware para la evaluación de algoritmos de cifrado simétrico de textos claros.

Título del Trabajo: Firt step in a Pc cluster development with openMosix

Autores: Javier Bilbao, Gorka Garate

Año: 2008

Lugar: Bilbao, Spain.

Síntesis: En este trabajo se afirma que es posible hacer máquinas de alta capacidad de cálculo mediante la interconexión de equipos con redes de alta velocidad. A este tipo de sistemas se les llama sistemas distribuidos y hoy en día es la tecnología más utilizada para resolver problemas que requieren una gran capacidad de cálculo. La agrupación de computadoras permite que varias computadoras trabajen juntas para resolver los problemas de informática común. Este artículo centra su estudio en uno de los diferentes métodos de agrupamiento de computadoras, openMosix. El trabajo también evalúa el clúster por dos casos de estudio, donde se mejora el tiempo de ejecución de los problemas resueltos.

Resultados: El autor del artículo concluye con la realización satisfactoria de agrupamiento de computadoras bajo el método openMosix, el cual únicamente permite agrupamiento de la forma SSI (Single System Image) que se refiere a que únicamente se puede realizar el agrupamiento con todos los nodos ejecutando un mismo sistema operativo.

Conclusión: Este artículo aporta para mi investigación, el conocimiento a fondo de los métodos de agrupamiento de computadoras openMosix, ventajas, desventajas y dificultades para su implementación.

Título del Trabajo: Diseño e implementación de un clúster tipo Beowulf para el desarrollo de cómputo científico avanzado.

Autores: Edgar Rubén González Ramírez, Abimael Rodríguez Sánchez

Año: 2008

Lugar: Instituto Politécnico Nacional. México, D.F.

Síntesis: Este trabajo de tesis ha desarrollado e implementado una herramienta computacional, llamada "Cluster Beowulf", para mejorar la programación paralela. Comenzando con la memoria compartida y distribuida, el autor implemento el agrupamiento tipo Beowulf pensando en procesamiento paralelo: la optimización de las fuentes informáticas. El diseño se construye mediante una distribución Linux Fedora Core 6 a 64 bits, utilizando todas las ventajas que ofrece el estándar Gigabyt Ethernet. La programación paralela fue desarrollada en lenguaje C tanto para MPI como para OPENMP.

Resultados: El autor del artículo concluye que, de acuerdo con los resultados, el tiempo de procesamiento se redujo con el uso múltiples núcleos, mejoró hasta ocho veces gracias al multihilo.

Conclusión: Este artículo aporta para mi investigación, el conocimiento a fondo de los métodos de agrupamiento de computadoras Beowulf, y su implementación en sistemas operativos de uso libre Linux distribución Fedora 6.

Marco Teórico

2.1 Seguridad de la Información

La Seguridad de la Información consiste y hace referencia a la protección de cualquier amenaza para salvaguardar los activos fijos y en especial toda información de la cual está basada la continuidad de las operaciones de la empresa sean estas ocasionadas dentro o fuera de la misma, con esto se logra disminuir los daños y perjuicios que estas amenazas causarían a la organización y a la vez aumentar las oportunidades de servicio hacia otras empresas.

La Seguridad de la información se consigue mediante la implementación de controles efectivos, que pueden ser normas, manual de funciones, buenas prácticas, procedimientos, estructuras organizativas, funciones de software y hardware, planes de contingencia y herramientas que se encargan de proteger la privacidad e integridad de la información que se almacena en un sistema informático. Estos controles necesitan continuamente ser establecidos, monitoreados, revisados y mejorados en donde sea necesario, para asegurar que se cumplan los objetivos específicos de seguridad en cada área de negocio de la organización.

La seguridad informática es en realidad una rama de la seguridad de la información, aunque de forma práctica algunos autores suelen utilizar de manera indistinta ambos términos.

2.1.1 Objetivo de la seguridad de la información. En la seguridad de la información sabiendo que dichos procesos de seguridad permiten proteger el recurso más importante y valioso de una organización, partiendo como principio fundamental los procesos de aseguramiento el

objetivo del proceso de seguridad informática es obtener un nivel aceptable de seguridad, ya que la información es lo más valioso que tiene una institución o empresa, por lo tanto, esta debe ser protegida para que no sea vulnerable y utilizada para fines maliciosos.

2.1.2 Principios de la seguridad informática

- Confidencialidad: Los datos solo deben ser conocidos y accedidos por quienes estén autorizados durante su almacenamiento, procesamiento o trasmisión. Verificar y certificar que solo los usuarios con accesos autorizados puedan acceder a la información.
- Integridad: Los datos solo pueden ser modificados y eliminados por quienes estén autorizados para ello, es decir los sistemas y aplicaciones solo deben ser operados por personal autorizado.
- Disponibilidad: Los sistemas que almacenan datos e información deben garantizar su acceso, cuando así se requiera, por quienes tengan derecho a ello.

2.1.3 Características de seguridad informática

- Control: Solo los usuarios autorizados deciden cuándo y cómo permitir el acceso a la información.
- Auditoría: Establece las acciones, cuándo las implementa y quién las realiza sobre el sistema.
- Autenticidad: Define que la información necesaria es correcta, utilizable y válida en tiempo y forma.
- No Repudio: Su función es evitar que alguna entidad que envió o recibió información alegue, que no lo hizo.

2.1.4 Formas de Autenticación

- Contraseña: Una contraseña es un tipo de seguridad informática, ya que es una serie de caracteres secretos que permiten a un usuario tener acceso a un archivo, equipo de cómputo o programa y negar el acceso a usuarios no autorizados.
- Certificado digital: Es un tipo de seguridad informática que permite la firma digital electrónica que garantiza técnica y legalmente la identidad de una persona en Internet.
- Autenticación HTTP: Tipo de seguridad informática que delega la autenticación de usuarios a un servidor a través de *Kerberos*.

2.1.5 Ciber-seguridad. Con los avances tecnológicos y la demanda que ha causado en ello, existe un elevado índice de información personal y empresarial que circula y se mantiene en la red por la utilización del internet, esto permite que toda información quede expuesta a cualquier ataque de ciber-delincuentes los cuales buscan la manera de vulnerar cualquier sistema para espiar, robar información o hacer daño de cualquier tipo por lo que se ha visto la necesidad de tomar medidas de seguridad para salvaguardar la información ya que es muy importante.

La ciberseguridad es la protección de los sistemas conectados a Internet, incluidos el hardware, el software y los datos, de los ciberataques. En un contexto informático, la seguridad comprende la ciberseguridad y la seguridad física; ambas son utilizadas por las empresas para proteger contra el acceso no autorizado a los centros de datos y otros sistemas de cómputo. El objetivo de la ciberseguridad es limitar el riesgo y proteger los activos de Tecnologías de Información de los atacantes con intenciones maliciosas. La seguridad de la información, está

diseñada para mantener la confidencialidad, integridad y disponibilidad de los datos, es un subconjunto de la ciberseguridad.

2.2 Criptografía

La Criptografía también llamada “escritura oculta”, tradicionalmente en el ámbito de criptografía es donde se ocupan las técnicas de cifrado o codificación que permiten alterar el contenido de cierta información o mensajes con el fin de hacerlos ininteligibles a receptores o personas no autorizadas. (Menéndez-Barzanallana, 17)

Actualmente la criptografía se encarga de analizar algoritmos, protocolos y sistemas que son utilizados para proveer seguridad a las comunicaciones, información y a las entidades que se comunican entre sí.

2.2.1 Tipos de Criptografías

2.2.1.1 Tablas Hash. Un hash o función hash toma un grupo de caracteres (llamado clave) y lo asigna a un valor de cierta longitud (llamado valor hash o hash). El valor hash es representativo de la cadena de caracteres original, pero normalmente es más pequeño que el original.

El hash también es utilizado en el cifrado y descifrado de firmas digitales. La función hash transforma la firma digital, luego tanto el valor hash como la firma se envían al receptor. El receptor utiliza la misma función hash para generar el valor hash y luego lo compara con el recibido

en el mensaje. Si los valores hash son los mismos, es probable que el mensaje se haya transmitido sin errores.

Un ejemplo de una función hash se llama plegado(folding). Esta toma un valor original, lo divide en varias partes, luego agrega las partes y usa los últimos cuatro dígitos restantes como valor o clave hash.

2.2.1.2 Tipos de algoritmos Hash

2.2.1.2.1 SHA - 1 (Secure Hash Algorithm). Es un algoritmo criptográfico de Hash publicado por el gobierno de los Estados Unidos. Genera un valor Hash de 160 bits a partir de una secuencia de longitud arbitraria. Es considerado lo suficientemente seguro para aplicaciones prácticas, pero hay disponibles versiones más robustas, SHA - 256, SHA - 384 y SHA - 512 que generan valores Hash de 256, 384 y 512 bits respectivamente, estas versiones reemplazarán a SHA - 1 mientras se siga trabajando sobre ellas. (Borghello, 2009)

2.2.1.2.2 MD5 (Message Digest Algorithm). En criptografía, MD5 es una función hash criptográfica ampliamente utilizada con un valor hash de 128 bits. Como estándar de Internet (RFC 1321), MD5 se ha empleado en una amplia variedad de aplicaciones de seguridad, y también se usa comúnmente para verificar la integridad de los archivos. Un hash MD5 generalmente se expresa como un número hexadecimal de 32 dígitos.

MD5 es una versión reforzada de MD4. Al igual que MD4, el hash MD5 fue inventado por el profesor Ronald Rivest del Massachusetts Institute of Technology (MIT). Además, MD5 se

utilizó como modelo para SHA-1, ya que comparten muchas características comunes. MD5 y SHA-1 son los dos algoritmos hash más utilizados en la actualidad, aunque el uso de MD5 ha disminuido ya que ahora se considera roto. (Borghello, 2009)

2.2.1.2.3 RIPEMD – 160. Es una función hash criptográfica basada en la construcción Merkle-Damgård (MD). Se usa en el estándar de Bitcoin. Es una versión reforzada del algoritmo RIPEMD que produce un resumen de hash de 128 bits, mientras que el algoritmo RIPEMD-160 produce una salida de 160 bits. La función de compresión consta de 80 etapas formadas por 5 bloques que se ejecutan 16 veces cada uno. Este patrón se ejecuta dos veces con los resultados combinados en la parte inferior utilizando la adición del módulo 32. (Borghello, 2009)

2.2.1.2.4 Función Merkle-Damgård. La construcción MD es una metodología para crear funciones hash resistentes a colisiones a partir de funciones de compresión unidireccionales. Ralph Merkle la propuso en el año de 1979 en "A certified Digital Signature", posterior a este estudio Ivan Damgård demostró de manera independiente que la estructura era sólida. (Merkle & Damgård, s.f.)

2.2.1.3 Cifrado simétrico. Los cifrados simétricos son los cifrados criptográficos más antiguos y más utilizados. En un cifrado simétrico, la clave que descifra el texto cifrado es la misma (o puede derivarse fácilmente) de la clave que cifra el texto claro. Esta clave a menudo se conoce como la clave secreta. Los cifrados simétricos más utilizados son DES y AES.

La problemática que surge es la seguridad de la clave:

- El emisor cifra el mensaje con una clave.
- Envía el mensaje cifrado, de tal manera que nadie puede descifrarlo sin dicha clave.
- El receptor obtiene la clave y entonces puede descifrar el mensaje con la misma clave que recibió del emisor.

Algunos algoritmos de este tipo son:

2.2.1.3.1 RC5 (Cifrado de Rivest). Es un algoritmo de cifrado de bloque, de clave simétrica diseñado por Ron Rivest en 1994. Se destaca por ser simple, rápido (debido a que usa solo operaciones de cómputo primitivas como XOR o shift.) y además por consumir menos memoria.

RC5 es un cifrado de bloques y aborda dos bloques de palabras a la vez. Dependiendo del tamaño del bloque de texto sin formato, número de rondas y tamaño de clave, se pueden definir varias instancias de RC5 y cada instancia se denota como RC5-w / r / b donde w = tamaño de palabra en bits, r = número de rondas y b = tamaño de la clave en bytes. (Pawar, 2016)

Como se ha mencionado, RC5 utiliza bloques de 2 palabras, el tamaño del bloque de texto sin formato puede ser de 32, 64 o 128 bits.

2.2.1.3.2 IDEA (International Data Encryption Algorithm). Básicamente es un algoritmo de cifrado de bloque simétrico. El algoritmo asimétrico utiliza la misma clave para el cifrado y descifrado. Un cifrado de bloque divide el mensaje en bloques, cada uno de una longitud fija, y luego cifra cada bloque de forma independiente.

El tamaño de bloque típico es de 16 bytes de 128 bits. Un cifrado de bloque generalmente operará en bloques o rondas donde se aplica parte de la clave a la ronda. Después de un cierto número de rondas, digamos entre 10 y 16, se termina con el texto cifrado para ese bloque.

El bloque de texto cifrado tiene exactamente el mismo tamaño que el bloque de texto sin formato, 16 bytes. Para cada ronda, se trabaja en el bloque utilizando una parte de la clave de cifrado a la que se le llama la clave redonda. Múltiples claves redondas se derivan de la clave de cifrado utilizando para esto un programa de claves.

La programación de claves es un algoritmo que cambia, XOR, multiplica y realiza otros tipos de operaciones en la clave de cifrado original para obtener estas claves redondas. Bueno, si tengo un bloque de 16 bytes y tengo una clave de 128 bits, que también es de 16 bytes. (Educba, 2019)

2.2.1.3.3 DES (Data Encryption Standard). El algoritmo DES es el algoritmo de seguridad más popular. Es un algoritmo simétrico, lo que significa que se utilizan las mismas claves para cifrar / descifrar datos confidenciales. La longitud de la clave es de 8 bytes (64 bits). Entonces, para cifrar / descifrar datos, el algoritmo DES utiliza una clave de 8 bytes, pero 1 byte (8 bits) para la verificación de paridad. Es un algoritmo de cifrado de bloque, por lo tanto, el tamaño del bloque de datos del algoritmo DES es de 64 bits. Para cifrar / descifrar datos, el algoritmo DES utiliza la estructura Feistel. Entonces, utiliza una ronda para cifrar / descifrar datos. Aunque el tamaño del

bloque de datos es de 64 bits, el número de rondas será de 16 rondas, así mismo, utilizará diferentes subclaves para cada ronda, por lo tanto, el número de subclaves será de 16 subclaves.

Como se puede observar la clave efectiva es de 56 bits, por consiguiente, se tiene 2^{56} combinaciones posibles, por lo que la fuerza bruta se hace casi imposible. (Sanjoy, 2015)

2.2.1.4 Cifrado asimétrico. Los cifrados asimétricos también se conocen como cifrados con claves públicas y privadas. Usan dos claves, una para el cifrado de mensajes y la otra durante el descifrado.

Un intruso puede encriptar cualquier mensaje usando la clave pública conocida. Los cifrados asimétricos son, por lo tanto, vulnerables a los ataques de texto sin formato. Los cifrados con encriptación de clave pública deben proporcionar seguridad contra tales ataques. Después de encriptar dos mensajes con la misma clave pública, el intruso no puede distinguir qué texto cifrado está conectado con qué texto plano. Además, un observador que analiza dos mensajes encriptados usando el mismo algoritmo y la misma clave pública, no puede distinguir sus textos cifrados. (Kowalczyk, 2015)

Los cifrados asimétricos son mucho más lentos que los cifrados simétricos (generalmente mil veces más lentos). Es una práctica común usar el cifrado de clave pública solo para establecer la conexión segura y negociar la nueva clave secreta, que luego se utilizará para proteger la comunicación mediante cifrado simétrico. Ejemplos de este cifrado son:

2.2.1.4.1 DSA (Digital Signature Algorithm - Algoritmo de Firma digital). Es un estándar del Gobierno Federal de los Estados Unidos para firmas digitales. Fue propuesto por el Instituto Nacional de Estándares y Tecnología (NIST) en agosto de 1991 para ser usado en su Estándar de Firma Digital (DSS), en el año de 1993.

La primera parte del algoritmo DSA es la generación de clave pública y clave privada, que puede describirse como:

- Se elige un número primo **q**, que se llama divisor primo.
- Se elige otro número primo **p**, de modo que **p**–1 mod **q** = 0. p será llamado módulo primo.
- Ahora se elige un número entero **g**, tal que 1 <**g** <**p**, **g** ** **q** mod **p** = 1; **g** = **h** ** ((**p** – 1) / **q**) mod **p**. **q** también se llama módulo de orden multiplicativo de **g** módulo **p**.
- Se elige un número entero, tal que 0 <x <q.
- Se calcula **y** como **g** ** **x** mod **p**.
- Se empaqueta la clave pública como {**p**, **q**, **g**, **y**}.
- Por último, se empaqueta la clave privada como {**p**, **q**, **g**, **x**}.

La segunda parte del algoritmo DSA es la generación de firma y la verificación de firma. Para generar una firma de mensaje, el remitente puede seguir estos pasos:

Se genera el resumen del mensaje **h**, utilizando un algoritmo hash como SHA1.

Se genera un número aleatorio **k**, tal que 0 <k <q.

Se calcula **r** como (**g** ** **k** mod **p**) mod **q**. Si **r** = 0, seleccione una **k** diferente.

Ahora se calcula **i**, de modo que **k** * **i** mod **q** = 1. **i** se llama el inverso multiplicativo modular de **k** módulo **q**.

Se calcula **s** = **i** * (**h** + **r** * **x**) mod **q**. Si **s** = 0, se selecciona una **k** diferente.

Por último, se empaqueta la firma digital como {r, s}. (Yang, 2019)

2.2.1.4.2 RSA. Encriptación RSA, encriptación Rivest-Shamir-Adleman, tipo de criptografía de clave pública ampliamente utilizada para la encriptación de datos de correo electrónico y otras transacciones digitales a través de Internet. RSA lleva el nombre de sus inventores, Ronald L. Rivest, Adi Shamir y Leonard M. Adleman, quienes lo crearon mientras trabajaban en la facultad del Instituto Tecnológico de Massachusetts. (Simmons, 2019)

En el sistema RSA, un usuario elige de manera secreta un par de números primos **p** y **q** tan grandes que factorizar el producto **n** = **pq** está mucho más allá de las capacidades informáticas proyectadas para la vida útil de los cifrados. A partir del año 2000, los estándares de seguridad del gobierno de EE. UU. exigen que el módulo tenga un tamaño de 1.024 bits, es decir, **p** y **q** deben tener un tamaño de aproximadamente 155 dígitos decimales, por lo que **n** es aproximadamente un número de 310 dígitos. Dado que los números duros más grandes que actualmente se pueden factorizar son solo la mitad de este tamaño, y dado que la dificultad de factorizar se duplica aproximadamente por cada tres dígitos adicionales en el módulo, se cree que los módulos de 310 dígitos están a salvo de factorizar durante varias décadas.

2.2.1.5 Cifrado híbrido. Es una técnica común que combina diferentes funciones criptográficas como el cifrado de bloque. Con el cifrado híbrido, se pueden cifrar datos, firmarlos o agregar un MAC y enviarlos en un objeto seguro (como una cadena o archivo) a través de un canal de comunicación inseguro (sin un acuerdo clave entre dos partes).

En el sistema de cifrado híbrido, el texto plano (sin formato) se divide en *tokens*. Luego, se calcula la longitud de cada ficha. Los *tokens* y sus longitudes se almacenan en una matriz. Posteriormente, se aplica el algoritmo César anidado sugerido. El texto producido se almacena en una matriz **n** x **m**, llamada A. Luego, la transposición de esta matriz, A^t, se calcula utilizando nuestra versión revisada del cifrado de transposición para resolver el problema de secuenciación. A continuación, el algoritmo RSA revisado se aplica en la matriz producida. En el paso final, se aplica el formato de datos textuales usando el código ASCII para producir el texto cifrado. El proceso de descifrado se realiza de la misma manera, pero en orden inverso. (Nahar & Abu Abbas, 2010)

2.3 Criptoanálisis

El criptoanálisis es el proceso de estudiar sistemas criptográficos para buscar debilidades o fugas de información. En general, se considera que el criptoanálisis explora las debilidades de las matemáticas subyacentes de un sistema criptográfico, pero también incluye la búsqueda de debilidades en la implementación, como ataques de canal lateral o entradas de entropía débiles.

Su función es la de conseguir, obtener el significado de mensajes creados por medio de criptografía. El objetivo principal del criptoanálisis es opuesto al de la criptografía. Tiene como finalidad encontrar la debilidad de las diferentes técnicas criptográficas para explotarla y de esta manera reducir o eliminar la seguridad de la técnica criptográfica. Cualquier intento de criptoanálisis puede ser llamado ataque. Una vez que un atacante logra romper la seguridad que una técnica criptográfica aportaba al sistema, se dice que dicho sistema ha sido roto y por lo tanto dicho ataque tuvo éxito.

El criptoanálisis es la investigación de sistemas, texto cifrado y cifrados para revelar el significado oculto o los detalles del sistema mismo. El objetivo de este tipo de estudio es descubrir los aspectos ocultos incluso si la clave o el algoritmo principal no se pueden descifrar. Se necesita una combinación exitosa de persistencia, matemáticas, intuición, curiosidad y una computadora que funcione para hacer un buen criptoanalista. Este tipo de ruptura de código es extremadamente importante, especialmente en el mundo tecnológicamente dependiente de hoy. El criptoanálisis es constantemente realizado por criptógrafos que intentan obtener información valiosa y por aquellos que intentan descifrar el código para revelar secretos. (Security Degree Hub, 2017)

2.4 Niveles de seguridad

Los niveles de seguridad de sistemas operativos son otorgados de acuerdo con el sistema operativo que esté utilizando la empresa o institución ya sea pública, privada, gubernamental o no gubernamental. En el manual “Trusted Computer System Evaluation Criteria” (United States

Government Department of Defense, 1983) el Departamento de Defensa de los Estados Unidos define los siguientes niveles.

2.4.1 Nivel D1. Este nivel es el más bajo de seguridad, por lo tanto, la norma establece que el sistema no es confiable. Esta división contiene solo una clase. Está reservado para aquellos sistemas que han sido evaluados pero que no cumplen con los requisitos para un nivel de evaluación superior.

2.4.2 Nivel C1. Un sistema de nivel C1 cumple nominalmente los requisitos de seguridad discrecionales al proporcionar separación de usuarios y datos. Incorpora alguna forma de controles creíbles capaces de imponer limitaciones de acceso de manera individual, es decir, aparentemente adecuados para permitir que los usuarios puedan proteger la información privada o del proyecto y evitar que otros usuarios lean o destruyan accidentalmente sus datos. Se espera que el entorno de nivel C1 sea uno de los usuarios cooperantes que procesan datos con el mismo nivel de sensibilidad. Los siguientes son requisitos mínimos para los sistemas asignados a una clasificación de nivel C1 son:

- Control de acceso discrecional, por ejemplo, listas de control de acceso (ACL), usuario / grupo.
- Por lo general, para usuarios que están en el mismo nivel de seguridad.
- Protección de nombre de usuario y contraseña y base de datos de autorizaciones seguras (ADB).
- Sistema operativo protegido y modo de operaciones del sistema.
- Comprobación periódica de integridad de TCB.

- Mecanismos de seguridad probados sin derivaciones obvias.
- Documentación para la seguridad del usuario.
- Documentación para la seguridad de la administración de sistemas.
- Documentación para pruebas de seguridad.
- Documentación de diseño TCB.
- Típicamente para usuarios en el mismo nivel de seguridad
- La certificación C1 es rara. Los sistemas de ejemplo son versiones anteriores de Unix, IBM RACF.

2.4.3 Nivel C2. Los sistemas de esta clase imponen un control de acceso discrecional más fino que los sistemas (C1), lo que hace que los usuarios sean individualmente responsables de sus acciones a través de procedimientos de inicio de sesión, auditoría de eventos relevantes para la seguridad y aislamiento de recursos. Los siguientes son requisitos mínimos para los sistemas asignados a una clasificación de nivel C2:

- La protección de objetos puede ser para un solo usuario. Por ejemplo, a través de una base de datos ACL.
- La autorización de acceso solo puede ser asignada por usuarios autorizados.
- Protección de reutilización de objetos (Por ejemplo, para evitar la reasignación de objetos eliminados de manera segura).
- Procedimientos obligatorios de identificación y autorización para los usuarios. (Por ejemplo, Usuario Contraseña).
- Auditoría completa de eventos de seguridad (es decir, fecha / hora, evento, usuario, éxito / falla, ID de terminal)

- Modo de operación del sistema protegido.
- Protección adicional para la autorización y los datos de auditoría.
- Documentación como C1 más información sobre el examen de la información de auditoría.
- Esta es una de las certificaciones más comunes. Los sistemas operativos de ejemplo son: VMS, IBM OS / 400, Windows NT, Novell NetWare 4.11, Oracle 7, DG AOS / VS II.

2.4.4 Nivel B1. Los sistemas de este nivel requieren todas las características requeridas para el nivel C2. Además, debe estar presente una declaración informal del modelo de política de seguridad, etiquetado de datos y control de acceso obligatorio sobre sujetos y objetos nombrados. La capacidad debe existir para etiquetar con precisión la información exportada. Cualquier falla identificada por la prueba debe ser eliminada. Los siguientes son requisitos mínimos para los sistemas asignados a una clasificación de nivel B1:

- Etiquetado obligatorio de seguridad y acceso de todos los objetos. (Por ejemplo, archivos, procesos, dispositivos.)
- Comprobación de integridad de etiquetas (Por ejemplo, mantenimiento de etiquetas de sensibilidad cuando se exportan datos).
- Auditoría de objetos etiquetados.
- Control de acceso obligatorio para todas las operaciones.
- Capacidad para especificar el nivel de seguridad impreso en salida legible para humanos (Por ejemplo, impresoras).

- Capacidad para especificar el nivel de seguridad en cualquier salida legible por máquina.
- Auditoría mejorada.
- Protección mejorada del sistema operativo.
- Documentación mejorada.
- Los sistemas operativos de ejemplo son: HP-UX BLS, Cray Research Trusted Unicos 8.0, Digital SEVMS, Harris CS / SX, SGI Trusted IRIX.

2.4.5 Nivel B2. En los sistemas de nivel B2, el TCB se basa en un modelo de política de seguridad formal claramente definido y documentado que requiere que la aplicación de control de acceso discrecional y obligatoria que se encuentra en los sistemas de nivel B1 se extienda a todos los sujetos y objetos en el sistema. Además, se abordan los canales encubiertos. El TCB debe estructurarse cuidadosamente en elementos críticos para la protección y no críticos para la protección. La interfaz TCB está bien definida y el diseño y la implementación TCB le permiten someterse a pruebas más exhaustivas y una revisión más completa. Se fortalecen los mecanismos de autenticación, se proporciona una gestión confiable de las instalaciones en forma de soporte para las funciones de administrador y operador del sistema, y se imponen estrictos controles de gestión de la configuración. El sistema es relativamente resistente a la penetración. Los siguientes son requisitos mínimos para los sistemas asignados a una clasificación de nivel B2:

- Notificación de cambios en el nivel de seguridad que afectan a los usuarios interactivos.
- Etiquetas de dispositivos jerárquicos.

- Acceso obligatorio a todos los objetos y dispositivos.
- Comunicaciones de ruta confiables entre usuario y sistema.
- Rastreo de canales de almacenamiento encubiertos.
- Modo de operaciones del sistema más estricto en unidades independientes de varios niveles.
- Análisis de canales encubiertos.
- Pruebas de seguridad mejoradas.
- Modelos formales de TCB.
- Versión, actualización y análisis y revisión de parches.
- Los sistemas de ejemplo son: Honeywell Multics, Cryptek VSLAN, Trusted XENIX.

2.4.6 Nivel B3. El TCB debe satisfacer los requisitos del monitor de referencia para mediar todos los accesos de los sujetos a los objetos, ser a prueba de manipulaciones y ser lo suficientemente pequeño como para ser sometido a análisis y pruebas. Con este fin, el TCB está estructurado para excluir el código que no es esencial para la aplicación de la política de seguridad, con una importante ingeniería del sistema durante el diseño y la implementación del TCB dirigido a minimizar su complejidad. Se admite un administrador de seguridad, los mecanismos de auditoría se amplían para señalar eventos relevantes para la seguridad y se requieren procedimientos de recuperación del sistema. El sistema es altamente resistente a la penetración. Los siguientes son requisitos mínimos para los sistemas asignados a una clasificación de clase (B3):

- ACL adicionalmente basadas en grupos e identificadores.
- Acceso de ruta confiable y autenticación.
- Análisis automático de seguridad.
- Modelos TCB más formales.
- Auditoría de eventos de auditoría de seguridad.
- Recuperación confiable después de la caída del sistema y documentación relevante.
- Cero defectos de diseño en TCB y defectos mínimos de implementación.
- El único sistema operativo con certificación B3 es Getronics / Wang Federal XTS-300.

2.4.7 Nivel A1. Los sistemas en el nivel A1 son funcionalmente equivalentes a los del nivel B3 en que no se agregan características arquitectónicas adicionales ni requisitos de políticas. La característica distintiva de los sistemas en este nivel es el análisis derivado de la especificación formal del diseño y las técnicas de verificación y el alto grado resultante de seguridad de que el TCB se implementa correctamente. Esta garantía es de naturaleza de desarrollo, comenzando con un modelo formal de la política de seguridad y una especificación formal de alto nivel del diseño. Independientemente del lenguaje de especificación particular o del sistema de verificación utilizado, existen los siguientes criterios importantes para la verificación del diseño de nivel A1:

- Métodos formales y prueba de integridad de TCB.
- Estos son los únicos sistemas con certificación A1: Boeing MLS LAN, Gemini Trusted Network Processor, Honeywell SCOMP. (United States Government Department of Defense, 1983)

2.5 Sistemas Operativos

Un sistema operativo actúa como intermediario entre el usuario de una computadora y el hardware de la computadora. El propósito de un sistema operativo es proporcionar un entorno en el que un usuario pueda ejecutar programas de manera conveniente y eficiente.

Un sistema operativo es un software que administra el hardware de la computadora. El hardware debe proporcionar mecanismos apropiados para garantizar el correcto funcionamiento del sistema informático y para evitar que los programas del usuario interfieran con el correcto funcionamiento del sistema. (Carretero Pérez, de Miguel Anasagasti, García Carballeira, & Pérez Costoya, 2001)

Un sistema operativo es un programa que controla la ejecución de programas de aplicación y actúa como una interfaz entre el usuario de una computadora y el hardware de la computadora.

Una definición más común es que el sistema operativo es el único programa que se ejecuta en todo momento en la computadora (generalmente llamado *kernel*), y todo lo demás son programas de aplicación.

Un sistema operativo se ocupa de la asignación de recursos y servicios, como memoria, procesadores, dispositivos e información. El sistema operativo incluye los programas para administrar estos recursos, como un controlador de tráfico, un programador, un módulo de administración de memoria, programas de E / S y un sistema de archivos. (Pawar, 2016)

2.5.1 Windows. Sistema operativo de la empresa Microsoft, es un conjunto de programas que administra los recursos de una computadora. Windows empieza a trabajar cuando se enciende o inicializa el equipo para gestionar el hardware empezando desde los niveles más básicos.

Es importante conocer que los sistemas operativos funcionan tanto en las computadoras como en diferentes dispositivos electrónicos que usan microprocesadores. En este caso Windows, su versión estándar funciona con computadoras. Microsoft domina el mercado de los sistemas operativos debido a su comodidad, se puede decir, que éste sistema operativo se encuentra instalado en el 88.7% de los equipos de cómputo que cuentan con acceso a Internet en todo el mundo.

Se pueden mencionar algunas aplicaciones principales (que éstas también pueden ser desinstaladas por los usuarios o actualizadas sin que el sistema operativo se dañe o deje de funcionar), se encuentran, el procesador de texto WordPad, el navegador Internet Explorer, el reproductor multimedia Windows Media y el editor de imágenes Paint.

La principal novedad que aportó Windows desde sus orígenes fue la facilidad para usarlo y el atractivo visual. De hecho, su nombre (“ventanas”) se debe a la forma en que el sistema presenta a los usuarios los recursos con los que cuenta su computadora, lo que facilita las tareas cotidianas. Windows también suele recibir varias críticas por sus problemas de seguridad y por otros fallos.

2.5.2 Registro de Windows. El registro es una base de datos jerárquica que contiene datos que son críticos para el funcionamiento de Windows y las aplicaciones y servicios que se ejecutan en Windows. Los datos están estructurados en un formato de árbol. Cada nodo en el árbol se llama clave. Cada clave puede contener subclaves y entradas de datos llamadas valores. A veces, la presencia de una clave es el único dato que requiere una aplicación; otras veces, una aplicación abre una clave y usa los valores asociados con la clave. (Microsoft, Inc., 2019) Una clave puede tener cualquier número de valores, y los valores pueden tener cualquier forma. El registro es utilizado por:

- Autenticación
- Núcleo (Kernel).
- Controladores de dispositivos.
- Servicios.
- SAM.

Las carpetas del sistema se almacenarán en distintas rutas de acceso dependiendo de la versión de Windows con la que se esté trabajando, a continuación, cuatro de los cinco archivos que más interesan a esta investigación se encuentran en la siguiente ubicación: %SystemRoot% \ System32 \ Config \ dentro de las siguientes sub llaves:

- SAM - HKEY LOCAL MACHINE \ SAM
- SECURITY - HKEY LOCAL MACHINE \ SECURITY
- SOFTWARE - HKEY LOCAL MACHINE \ SOFTWARE
- SYSTEM - HKEY LOCAL MACHINE \ SYSTEM

2.5.3 Administrador de cuentas de seguridad. El administrador de cuentas de seguridad (SAM por sus siglas en inglés) es una base de datos en donde se encuentran las cuentas de seguridad. Esto es utilizado por Windows para administrar las cuentas de usuario y contraseñas en el formato *hash*.

Las contraseñas nunca se almacenan en formato de texto claro. Se almacenan en formato de *hash* para protegerlas de los ataques. La base de datos *SAM* se implementa como un archivo de registro y las obtiene del *kernel* de Windows y mantiene un bloqueo exclusivo al archivo *SAM*, esto quiere decir que el archivo tiene cierta medida de seguridad para el almacenamiento de las contraseñas. (Hernandez, 2013)

No es posible copiar el archivo *SAM* a otra ubicación en el caso de los ataques en línea. Dado que el archivo *SAM* está bloqueado con un bloqueo exclusivo al sistema de archivos, no puede ser copiado o movido mientras se ejecuta Windows. El bloqueo no iniciará hasta que la excepción de la pantalla azul haya sido inicializada o el sistema operativo haya sido apagado. Sin embargo, los archivos en formato *hash* de contraseñas fuera de línea están disponibles para los ataques de fuerza bruta, el contenido del archivo *SAM* puede ser copiado o descargado utilizando diversas técnicas.

Un identificador de seguridad (SID) es un valor único de longitud variable utilizado para identificar a un usuario. Cada cuenta tiene un SID único emitido por una autoridad, como un controlador de dominio de Windows, y almacenado en una base de datos de seguridad. Cada vez

que un usuario inicia sesión, el sistema recupera el SID para ese usuario de la base de datos y lo coloca en el *token* de acceso para ese usuario. El sistema usa el SID en el *token* de acceso para identificar al usuario en todas las interacciones posteriores con la seguridad de Windows. Cuando se ha usado un SID como identificador único para un usuario o grupo, no se puede volver a usar para identificar a otro usuario o grupo.

La seguridad de Windows usa SID en los siguientes elementos de seguridad:

- En descriptores de seguridad para identificar al propietario de un objeto y grupo primario.
- En las entradas de control de acceso, para identificar al usuario para quien el acceso está permitido, denegado o auditado.
- En *tokens* de acceso, para identificar al usuario y los grupos a los que pertenece.

2.5.3.1 Identificador de Seguridad (SID. Security Identifier)

S - 1 - 5 - 21 - 3983109318 - 554656936 - 3066752241 - 1000

S - indica que es un string de SID

1 - el nivel de revisión

5 - el identificador de valor de autoridad o *authority value* no siempre será 5. Entre otros valores que se pueden encontrar están los siguientes:

- 0 - sin autoridad
- 1 - autoridad global
- 2 - autoridad local
- 3 - autoridad de creador
- 4 - autoridad no única
- 5 - autoridad NT
- 9 - autoridad de administrador de recursos

21 - 3983109318 - 554656936 - 3066752241 - es el dominio en el que se encuentra el sistema o el identificador local de la computadora.

1000 - ID único para cada cuenta, aquí se encuentra el *RID* (*Relative Identifier*). Número de longitud variable, cualquier grupo o usuario que se cree de manera diferente a la predeterminada recibirá un *RID* de 1000 o superior. En el caso de que el valor sea 500, responde a que este *SID* representa la cuenta de un administrador, el valor de 500 siempre será el mismo en cualquier cuenta de administrador, a su vez, el valor 501 representa el *SID* de una cuenta Invitado (*Guest*). (Hernandez, 2013)

2.5.4 Autenticación NTML. NTLM (*NT LAN Manager*) es un protocolo empleado por muchos productos de Microsoft para realizar la autenticación desafío/respuesta, por lo tanto, es el esquema de autenticación que por defecto usa *Firewall* de Microsoft y también los productos del servidor *proxy*.

Este software fue desarrollado para afrontar el problema del trabajo con Tecnologías Java en un entorno orientado a Microsoft. Puesto que no se basa en ninguna especificación oficial de protocolo, por lo tanto, no hay garantía de que funciona correctamente en todos los casos. También ha estado en algunas instalaciones de Windows, donde funcionó con éxito.

La Autenticación *NTLM* se compone de dos protocolos: protocolo de autenticación *NTLM* y protocolo de autenticación *LM*. Estos protocolos utilizan una metodología de *hash* diferente para almacenar contraseñas de los usuarios en la base de datos *SAM*. (Hernandez, 2013) (Menéndez-Barzanallana, 17)

2.6 LINUX (ROCKS)

Rocks es una distribución para clúster de Linux de código abierto que permite a los usuarios finales construir de manera más amigable clústeres computacionales y muros de visualización en mosaico. Cientos de investigadores de todo el mundo han utilizado Rocks para implementar su propio clúster.

Desde mayo de 2000, el grupo Rocks ha abordado las dificultades de implementar clústeres manejables. Encaminados hacia un objetivo: crear clústers de manera más sencilla. Por sencilla significa fácil de implementar, administrar, actualizar y escalar. Este objetivo impulsa a los creadores de esta herramienta a ayudar a entregar el poder computacional de los clústeres a una amplia gama de usuarios científicos. Está claro que poner a disposición de una amplia gama de

científicos plataformas de computación paralelas estables y manejables ayudará enormemente a mejorar el estado del arte en herramientas paralelas. (Rocksclusters, 2010)

2.6.1 Escritorio. De todos los escritorios disponibles para el sistema operativo Linux, GNOME ha logrado convertirse en el más eficiente, estable y confiable, sin dejar de ser sencillo de usar. De hecho, la mayoría de los usuarios, independientemente de su experiencia, pueden trabajar con GNOME casi sin esfuerzo.

Los usuarios que son nuevos tanto en Linux como en GNOME harían mejor en conocer los entresijos del escritorio que hacen que Linux no solo sea fácil, sino también divertido.

Un entorno de escritorio agrupa programas que trabajan juntos para formar un espacio de trabajo informático cohesivo, unificado, transparente e interactivo para los usuarios. O, para decirlo de manera más simple: el entorno de escritorio es la interfaz de usuario para el sistema operativo. En el caso de GNOME, ese sistema operativo es Linux. Sin un entorno de escritorio, su única alternativa es una sesión de *Shell* (línea de comandos) basada en texto. (Wallen, 2019)

2.6.2 Paquete de Oficina. Una suite ofimática es una colección de software creada por el mismo proveedor y diseñada para ser utilizada para tareas rutinarias dentro de una organización. Por lo general, una suite ofimática incluye aplicaciones como procesamiento de texto, hojas de cálculo, presentación, correo electrónico, toma de notas, base de datos, colaboración y otros tipos de software relacionados. En la mayoría de los casos, cada aplicación en la suite ofimática se puede instalar por separado y todas las aplicaciones dentro de la suite admiten la interoperabilidad entre

sí. Microsoft Office Suite, Lotus Live Notes y LibreOffice son todas suites de oficina de uso común.

2.6.3 Organizador de archivos. Es una función avanzada diseñada para agrupar y renombrar archivos, usando sus atributos del sistema o meta atributos.

Cada archivo tiene atributos del sistema como la fecha (Acceso, Creado y Modificado), el tipo de archivo (definido por extensión) y asociado con la aplicación registrada de ese tipo de archivo. Estos atributos se pueden usar para generar un nuevo nombre de archivo o carpeta (grupo) para cada archivo con el mismo atributo. Además de los atributos del sistema, algunos archivos, en su mayoría medios o imágenes, pueden contener metacampos como el nombre del artista, el título, el nombre del álbum y otros. El Organizador de archivos también usa estos metacampos para agrupar archivos en una carpeta con el mismo atributo. Por lo tanto, el organizador de archivos opera reglas de organización de archivos que definen la jerarquía de agrupación de carpetas y las reglas de cambio de nombre de archivos.

Así como el conocido *"explorador de Windows"*, Linux integra un organizador de archivos y carpetas llamado NAUTILUS, para realizar tareas de manejo de archivos, como mover, copiar, renombrar, crear carpetas y de esta manera administrar nuestros dispositivos de almacenamiento.

2.7 Agrupamiento (Clúster) de Computadoras

El agrupamiento (clúster) de computadoras está conformado por una colección de computadoras autónomas interconectadas trabajando unidas como un solo recurso de computación integrado. El concepto de agrupamiento de computadoras nació cuando los pioneros de la supercomputación intentaban difundir diferentes procesos entre varias computadoras, para luego poder recoger los resultados que dichos procesos debían producir. Con un hardware más barato y fácil de conseguir se pudo perfilar que podrían conseguirse resultados muy parecidos a los obtenidos con aquellas máquinas mucho más costosas. (Chirinov, 2003)

Un agrupamiento de computadoras se puede definir como el trabajo realizado por dos o más computadoras que en conjunto se encargan de proveer una determinada solución. Tiene como finalidad agrupar el poder de cómputo de los nodos implicados para proporcionar una mayor escalabilidad, disponibilidad y fiabilidad. (Branch Bedoya & Mesa Múnera, 2008)

La escalabilidad es la capacidad de un equipo para hacer frente a volúmenes de trabajo cada vez mayores sin, por ello, dejar de prestar un nivel de rendimiento aceptable.

La disponibilidad y la fiabilidad, se encuentran bastante relacionadas, aunque difieren ligeramente en algunos aspectos. La disponibilidad es la calidad de estar presente y listo para su uso, mientras que la fiabilidad es la probabilidad de un correcto funcionamiento.

Entre los tipos de agrupamiento más comunes se pueden encontrar, de alta disponibilidad (ambiente a prueba de fallas), cómputo de alto desempeño (los procesos o las tareas dadas se presentan como un solo sistema virtual) y escalamiento horizontal (se utiliza para proporcionar una sola interfaz a un conjunto de recursos que pueden aumentar o disminuir arbitrariamente su tamaño en un cierto periodo de tiempo). (González Ramírez & Rodríguez Sánchez, 2008)

2.7.1 Beowulf. *E*s una clase de computadora masivamente paralela de altas prestaciones principalmente construida a base de agrupamiento de componentes hardware estándar (Llorens & Peña, 2002). Consta de un conjunto de nodos minimalistas, unidos por un medio de comunicaciones económico. Esto quiere decir que tienen lo mínimo para ejecutar su función, de hecho, los nodos por sí solos no son capaces de ejecutar siquiera un sistema operativo (Nievas, Pino, & Arroyo). Se puede ver como una supercomputadora paralela construida con hardware comercial de fácil adquisición, que posee como sistema operativo Linux. Los clústers Beowulf son extremadamente poderosos, pero no son para todas las personas. Su principal desventaja es que requieren software diseñado para poder aprovechar los recursos del clúster. (Lizárraga, 2002)

2.7.2 Mosix. Según Pérez (2001), la arquitectura Mosix está basada en la misma ideología de la arquitectura *Beowulf*. Se basa en un conjunto de parches aplicados al núcleo (kernel) de Linux y que asignan a todo el grupo de nodos un espacio de direcciones y de procesos común, gracias al cual los procesos migran de uno a otro con el fin de equilibrar favorablemente la carga del sistema global. Es una herramienta diseñada para realizar balanceo de carga en el agrupamiento de forma totalmente transparente de manera tal que los nodos se comportan como una sola máquina, y así incrementar el aprovechamiento de cada uno de los nodos (Chávez, y otros, 1999). La principal

ventaja de Mosix frente a *Beowulf* se basa precisamente en esto. Mosix da mejores respuestas que *Beowulf* frente a la caída o inserción de nodos. A parte de esto, Mosix permite un cambio constante de aplicación, o incluso, el correr aplicaciones independientes de forma simultánea.

2.7.3 OpenMosix. OpenMosix es una extensión del proyecto Mosix. Según Chirinov (2003), la idea de este modelo es que la distribución de tareas en el clúster la determina OpenMosix de forma dinámica, conforme se van creando tareas. Cuando un nodo está demasiado cargado, las tareas que se están ejecutando pueden migrar a cualquier otro nodo del clúster. Así desde que se ejecuta una tarea hasta que ésta muere, podrá migrar de un nodo a otro, sin que el proceso sufra mayores cambios.

Además, OpenMosix funciona a nivel de *kernel* por tanto puede conseguir toda la información que necesite para decidir que tanto está cargado un sistema y qué pasos se deben seguir para aumentar el rendimiento, además puede realizar más funciones que cualquier aplicación a nivel de usuario.

Después de elegir el tipo de clúster a utilizar, se procede a elegir los elementos básicos que se emplearán para implementar el clúster: capacidad de procesamiento, memoria, espacio del disco de cada nodo, así como también el ancho de banda de la comunicación entre los nodos. Se deberá decidir cuáles son importantes basándose en la variedad de aplicaciones que se piensen ejecutar en el clúster, y de la cantidad de dinero que se disponga para construirlo.

Bajo estos mismos parámetros, se debe escoger también el tipo de red que interconectará los nodos. (Turner, 2004)

Luego se debe escoger el sistema operativo a utilizar. Esta elección dependerá mucho de la máquina que se escoja. Linux es siempre una opción en cualquier máquina, y es la elección más común. Muchas de las herramientas necesarias para la implementación de un clúster han sido desarrolladas bajo Linux. Pero existen clústers corriendo bajo Windows NT, UNIX, Solaris y AIX.

Capítulo 3. Implementación de Clúster

3.1 Requisitos previos

Antes de iniciar la instalación del FrontEnd, es necesario asegurarse que las conexiones de red de los equipos junto con la configuración de la BIOS de cada uno sean las correctas.

3.1.1 Conexión de los equipos y acceso a la red

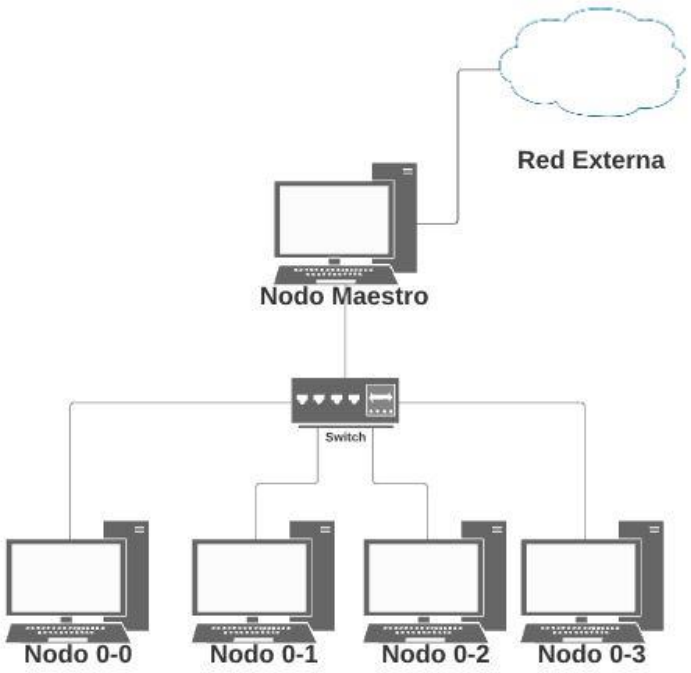


Figura 3.1. Esquema típico de configuración de un clúster. (Fuente: Elaboración propia.)

Un clúster de alto desempeño funciona como un único equipo, el cuál comparte los recursos de los nodos esclavos con el nodo maestro, para ello es esencial la existencia de una red interna de alta velocidad por medio de la cual se realice la comunicación entre el nodo maestro y los nodos esclavos como se muestra en la Figura 3.1. El nodo maestro necesitará 2 tarjetas de red de alta

velocidad, una para la comunicación interna y otra para la comunicación con redes externas. El sistema operativo Rocks asigna automáticamente la interface “eth0” a la red interna del clúster y la interface “eth1” a la red externa. En caso de que Rocks sea incapaz de determinar cuál es la red externa debido a la falta de conexión a internet, estas podrían ser asignadas de manera incorrecta, por tal motivo se sugiere asegurar el nombre de cada una de las interfaces.

Dado que Rocks no permite volverlas a asignar una vez finalizada la instalación, es necesario volver a reinstalarlo para poder solucionar este inconveniente.

3.1.2 Configuración de la BIOS. Para evitar problemas durante la instalación, conviene configurar correctamente la Bios.

3.1.2.1 ATA/IDE Mode. Algunos sistemas operativos, puede que no sean capaces de reconocer el controlador del disco duro como es en el caso de esta distribución de Rocks usada. Para evitar complicaciones de tener que añadir el controlador a mano, lo mejor es habilitar el modo “Legacy” de controlador de disco duro:

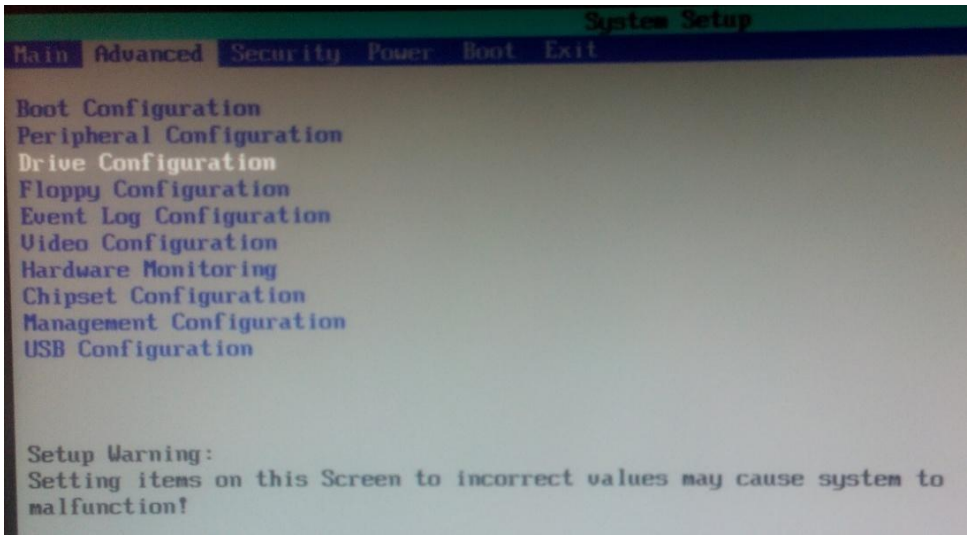


Figura 3.2. Imagen del Bios del equipo, en la opción "drive configuration". (Fuente: Elaboración propia.)

Para ello se accede a la Bios pulsando F2 al encender el equipo, se sitúa en la pestaña "Advanced" y se selecciona "Drive Configuration", como se muestra en la Figura 3.2.

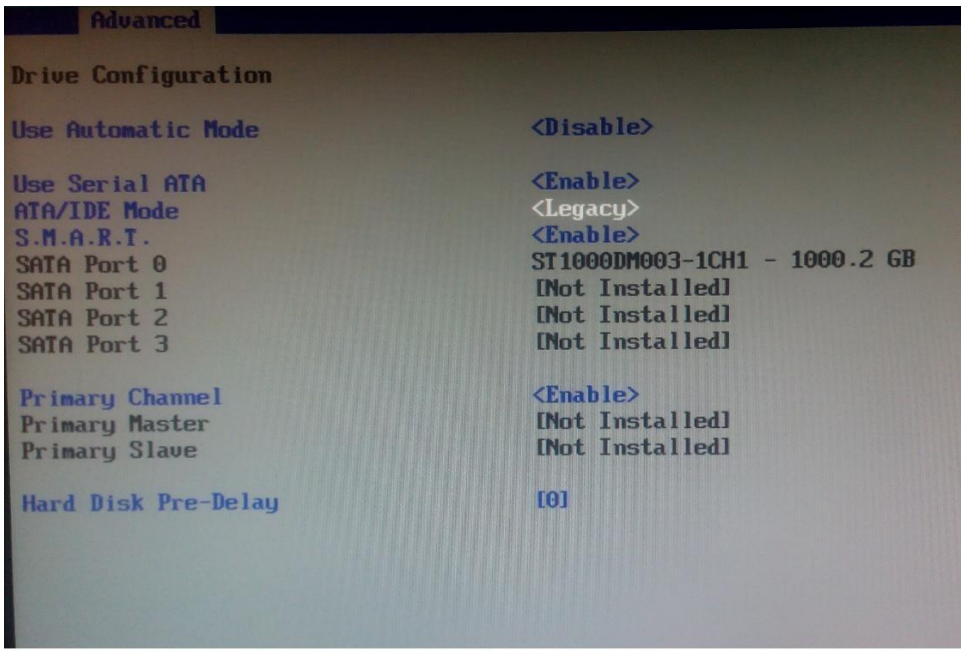


Figura 3.3. Bios. ATA/IDE Legacy Mode. (Fuente: Elaboración propia.)

Dentro del menú, se tiene que seleccionar el Modo “Legacy” en la opción “ATA/IDE Mode” como se muestra en la Figura 3.3 y para salir se guardan los cambios con F10.

3.1.2.2 PXE. Al momento de instalar un nodo, puede ocurrir falla al arrancar y aparecer bootsplash (Reboot and Select proper device). Para evitarlo, es necesario habilitar la opción de PXE para que tome los archivos de instalación del nodo maestro a través de la red interna.

En la Bios, se selecciona la pestaña “Boot” y dentro de ese mismo menú, se deshabilita “Silent boot” y se habilita la opción de “PXE boot to lan” como se muestra en la Figura 3.4.

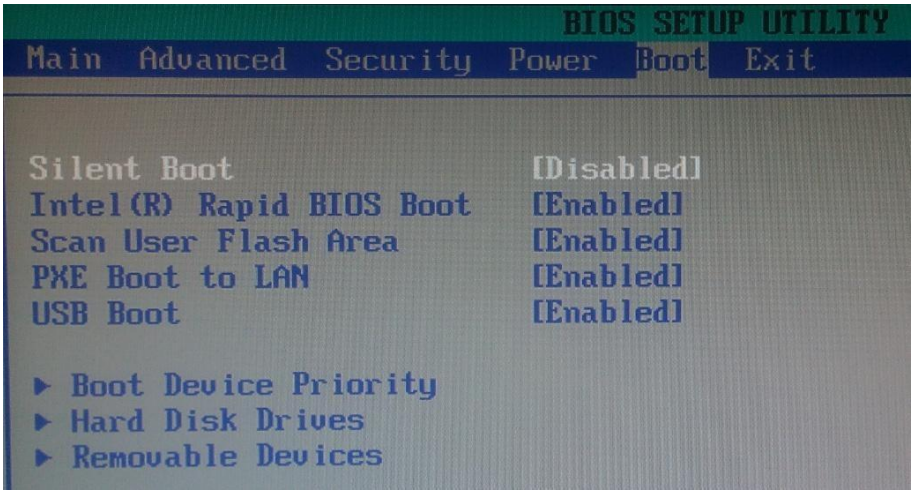


Figura 3.4. Bios. Boot PXE. (Fuente: Elaboración propia.)

Finalmente, hay que verificar que en la secuencia de arranque (Figura 3.5) como primer dispositivo sea desde red, para que de esta manera inicie desde el PXE:

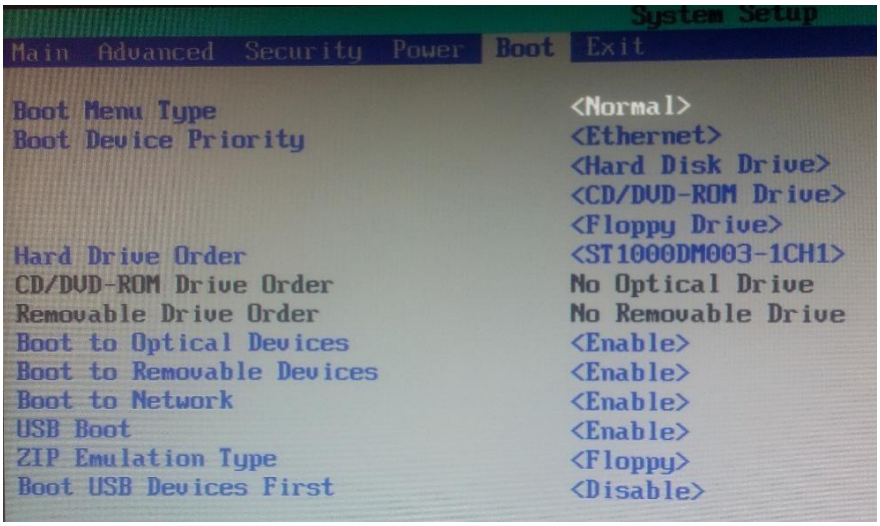


Figura 3.5. Bios. Boot device prioriry. (Fuente: Elaboración propia.)

Se guardan los cambios antes de salir.

3.2 Instalación de Sistema Operativo Rocks

Una vez configurados los equipos, se enciende el equipo que será el nodo maestro (FrontEnd), en el cuál previamente se ha modificado la secuencia de inicio del boot desde la Bios, de modo que al insertar el CD de Rocks, éste procederá a iniciar la instalación con la siguiente pantalla que se muestra a continuación:

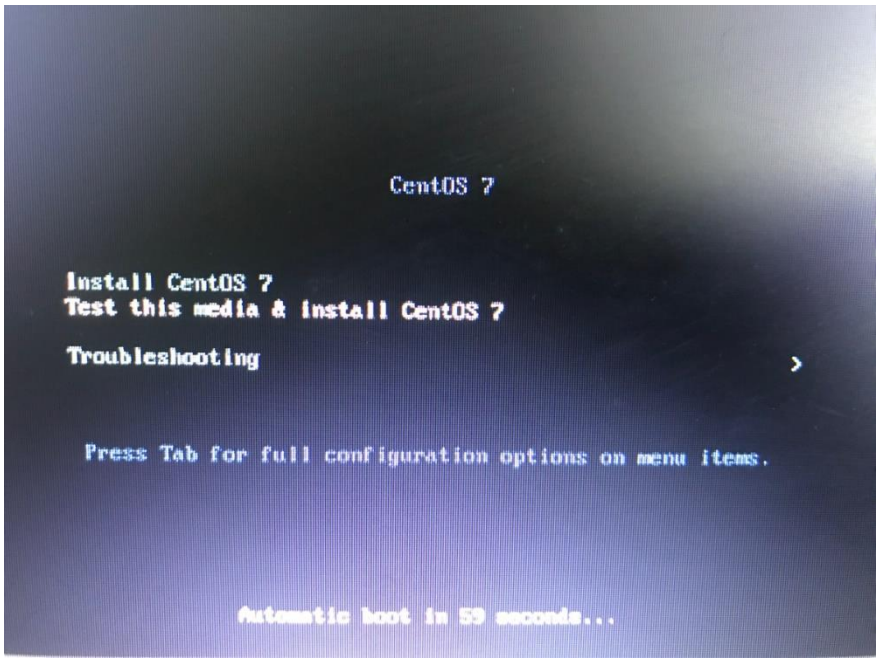


Figura 3.6. Pantalla de Inicio de instalación de Rocks. (Fuente: Elaboración propia.)

Para iniciar la instalación, únicamente se selecciona la opción “Install CentOS 7” como se muestra en la Figura 3.6, dando así comienzo al proceso de instalación. Tras finalizar un proceso de carga en memoria, Rocks mostrará un formulario en entorno gráfico, mediante el cual procederemos a configurar el FrontEnd.

Esta primera pantalla, se muestra cómo se desea realizar la instalación, puede realizarse por medio de la descarga del sistema operativo (se requiere internet). En este caso, como se muestra en la Figura 3.7, dado que previamente se descargó el DVD completo, se seleccionó “CD/DVD based rolls”.

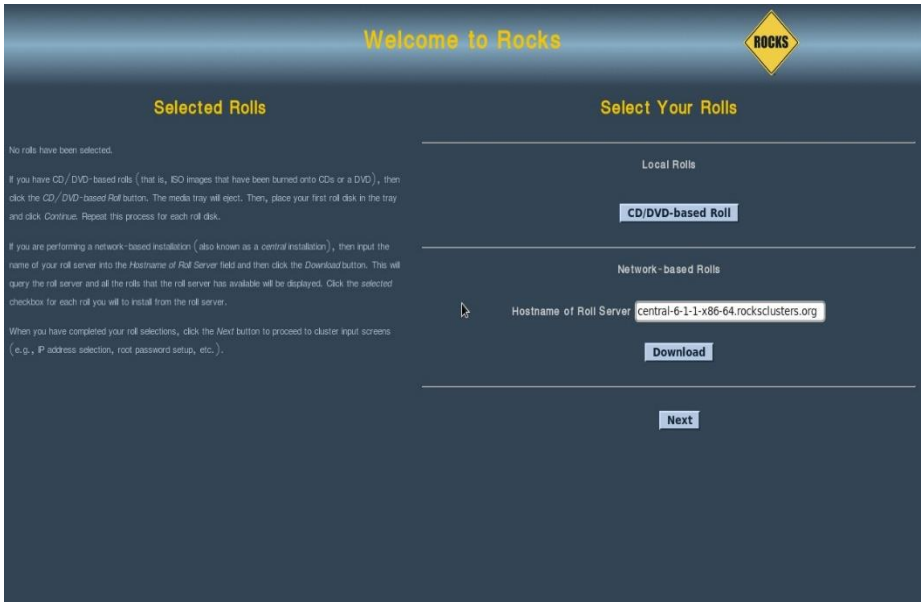


Figura 3.7. Pantalla de selección de Rolls. (Fuente: Elaboración propia.)

A continuación, Figura 3.8, se selecciona de la lista de Rolls aquellos que se desean instalar:

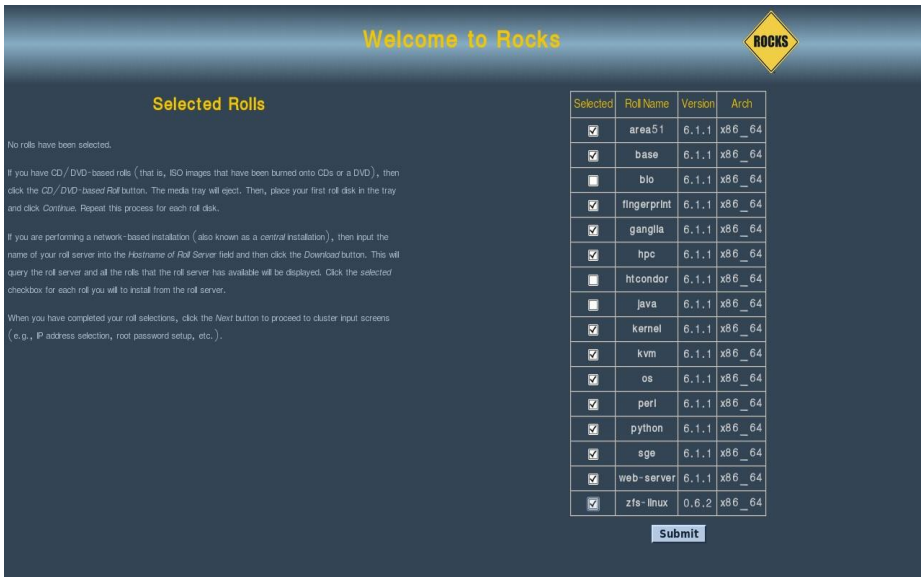


Figura 3.8. Listado de opciones de rolls para instalar. (Fuente: Elaboración propia.)

Tras presionar el botón “Submit”, la siguiente pantalla (Figura 3.9) muestra formulario en el que hay que indicar la información básica del clúster entre las cuales aparecen:

- Fully-Qualified Host Name: Nombre con el que se conocerá en la red externa el clúster
- Cluster Name: Nombre utilizado para ser identificado dentro de las herramientas del clúster.
- Certificate Organization, Locality, State and Country: Organización a la que pertenece el clúster, la localidad, el estado y la ciudad.
- Contact, URL and Lattitude/Longitude: Contacto, dirección web y posición GPS.

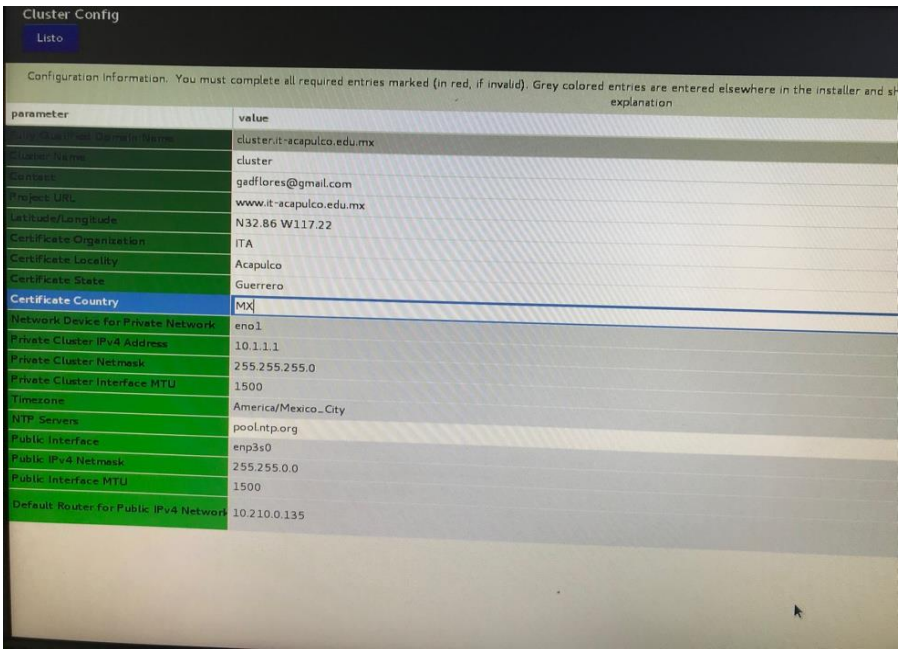


Figura 3.9. Información básica del clúster a instalar. (Fuente: Elaboración propia.)

Una vez que se ingresa la información al clúster, se presiona el botón “Listo” y posteriormente se podrá cambiar la configuración básica de la red interna del clúster. En este caso, Figura 3.10, se dejó el valor por defecto, una IP de clase B (10.1.1.1/16).

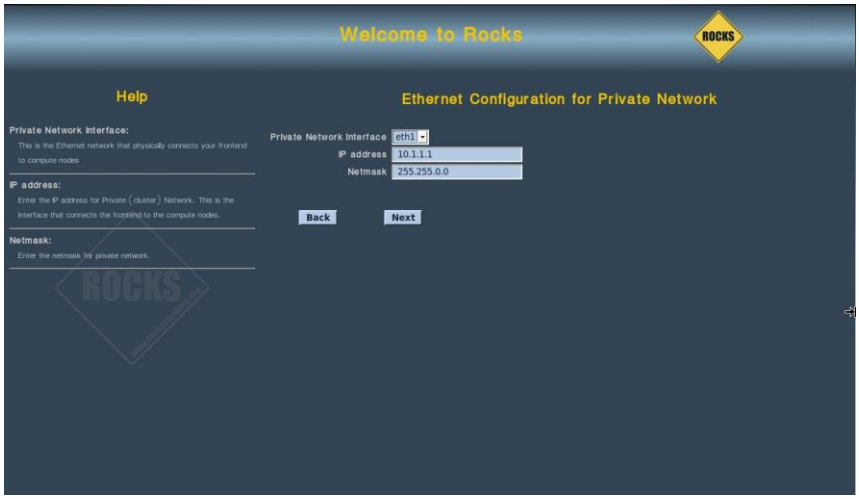


Figura 3.10. Configuración Ethernet de la red privada. (Fuente: Elaboración propia.)

El siguiente paso es la configuración de la red externa (para tener acceso a internet). En este punto, se introducen los datos de la red externa a la que se encuentra conectado el nodo maestro, como se muestra en la Figura 3.11.

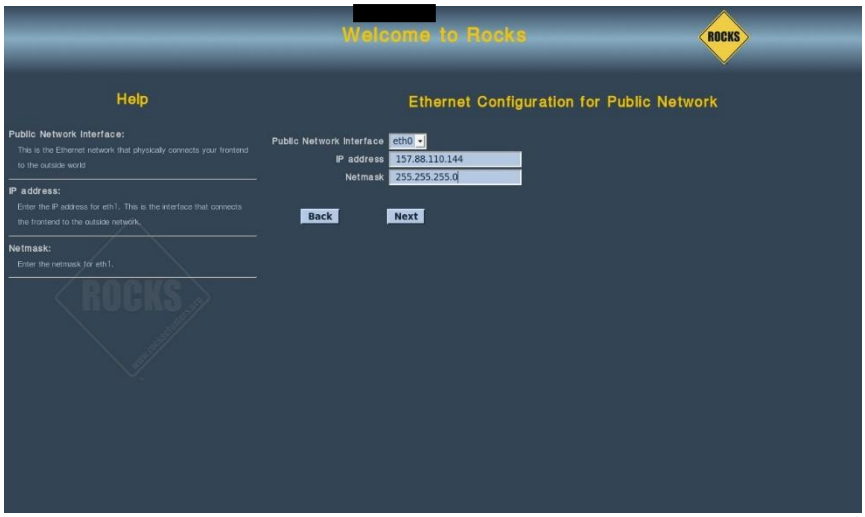


Figura 3.11. Configuración Ethernet de la red pública. (Fuente: Elaboración propia.)

Para terminar de configurar la red externa, como se muestra en la Figura 3.12 se ingresa la dirección IP del Gateway y los DNS (Servidor de nombre de dominio) que va a usar para el acceso a internet.

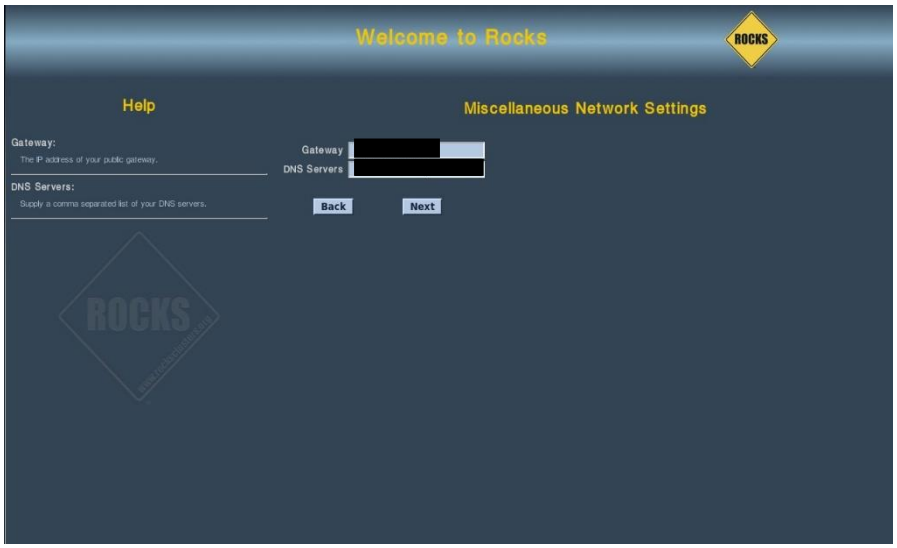


Figura 3.12. Configuración de DNS y Gateway. (Fuente: Elaboración propia.)

Se continúa con el resto del formulario donde se introduce la contraseña para el usuario *root*, tal y como se muestran en la Figura 3.13.

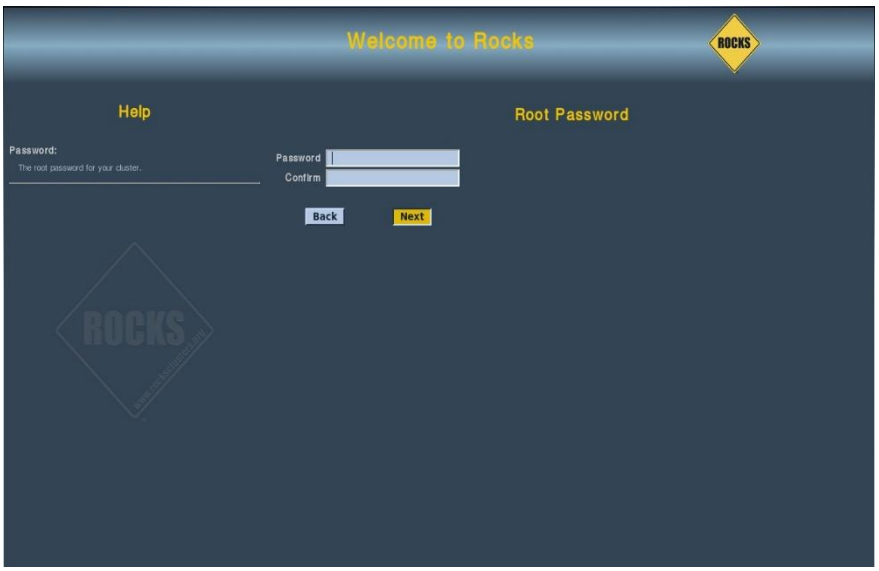


Figura 3.13. Ingresar contraseña de usuario root. (Fuente: Elaboración propia.)

Paso siguiente, se procede a seleccionar la zona horaria como se muestra en la Figura 3.14 donde se encuentra ubicado el clúster, así como un servidor de donde podrá obtener la zona horaria automáticamente, en este caso se dejó el valor por default.

Figura 3.14. Configuración horaria. (Fuente: Elaboración propia.)

En el formulario de la Figura 3.15 se selecciona la forma de particionar el disco duro (automática, manual), una vez seleccionado dará comienzo al proceso de instalación de manera

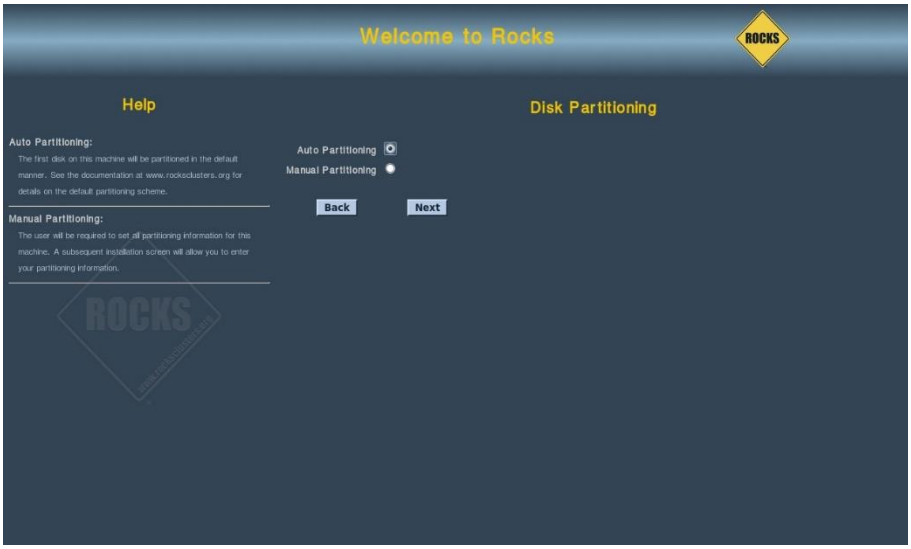


Figura 3.15. Particiones del disco duro. (Fuente: Elaboración propia.)

desatendida. Cuando ha concluido la instalación, el equipo se reiniciará, y se requerirá iniciar con el usuario *root* y la contraseña establecida previamente.

3.2.1 Configuración del teclado. Debido a que el teclado por defecto está en idioma inglés, podría provocar problemas al momento de escribir la contraseña u otros comandos.

Según las pruebas realizadas, aunque se cambie la configuración del teclado desde el menú de preferencias, únicamente se mantiene dentro de la sesión desde la que se ha cambiado dicha opción (lo cual implica que la contraseña de usuario hay que introducirla según la distribución inglesa).

3.3 Instalación de los nodos

Previo a la instalación de los nodos esclavos, cada uno de los equipos fueron configurados desde la Bios para que arranquen desde la red (sección 3.1.2.2). Para ir añadiendo los nodos, solo se tiene que introducir el siguiente comando desde nuestro FrontEnd:

```
$insert-ethers
```

A continuación, como se muestra en la Figura 3.16, aparecerá una ventana en la pantalla, en la cual se seleccionará la opción “Compute”.

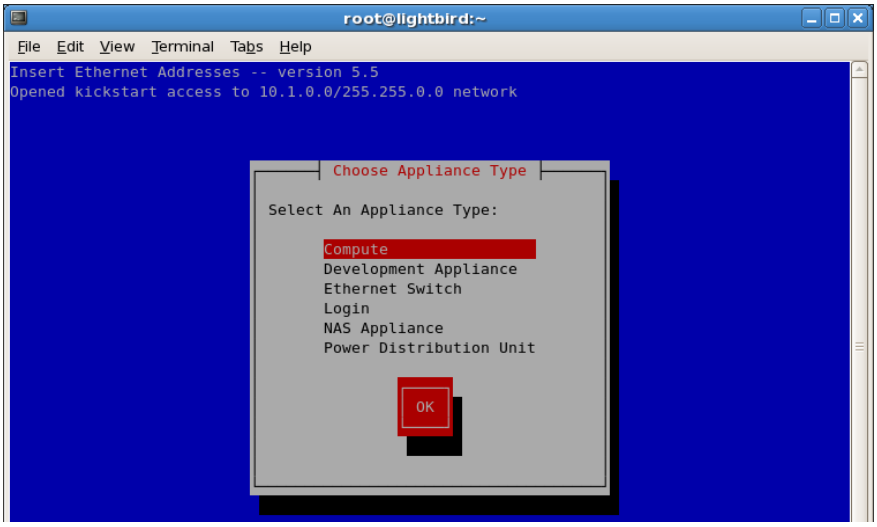


Figura 3.16. Pantalla para ingresar nodos. (Fuente: Elaboración propia.)

Una vez realizado este paso, se procede a encender los equipos que funcionarán como nodos esclavos uno a uno.

Se sugiere esperar a que cada nuevo equipo encendido, esté marcado con un asterisco antes de proceder a encender el siguiente, como se muestra en la Figura 3.17. De esta manera se asegura que cada nuevo nodo añadido posea el nombre y numeración deseada.

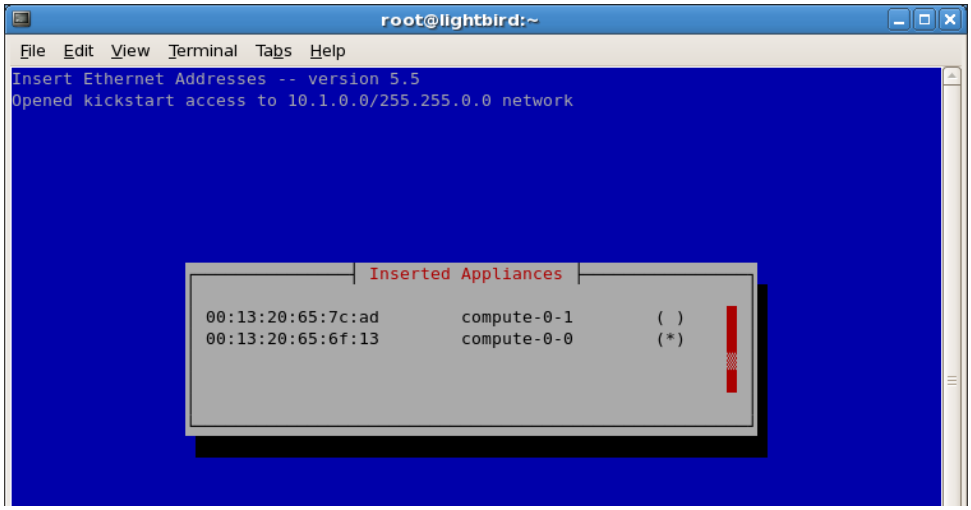


Figura 3.17. Indicador de inserción de nuevos nodos. (Fuente: Elaboración propia.)

Puede darse el caso de que un nodo no sea detectado automáticamente, eso indica que se ha omitido algún paso o parte de la configuración realizada hasta el momento es incorrecta. En ese caso, comprobar lo siguiente:

- Que el nodo esclavo este correctamente conectado a la red interna.
- Que el FrontEnd (nodo maestro) tenga correctamente configurada la red interna y externa
- Que el PXE esté habilitado en la Bios del nodo esclavo.

3.3.1 Administración de nodos. Algunos de los comandos más útiles para para la gestión de los nodos son los siguientes:

Eliminar un nodo de la lista:

```
$rocks remove host nombre_nodo
```

Reinstalar un nodo:

```
$rocks set host boot nombre_nodo action=install
```

Listar los nodos ya instalados:

```
$rocks list host
```

Acceder a un nodo:

```
$ssh nombre_nodo
```

Ejecutar un comando de manera remota a un nodo:

```
$ssh nombre_nodo ‘comando’
```

3.3.2 Creación de cuentas de usuario. Para agregar a un usuario hay que realizar los siguientes pasos:

1. Creación de la cuenta:

```
$useradd nombre_usuario
```

2. Asignación de una contraseña:

```
$passwd nombre_usuario
```

3. Sincronización de la cuenta de usuario con el resto de nodos:

```
$rocks-user-sync
```

3.3.3 Configuración del directorio compartido. El directorio /share es compartido por el FrontEnd y el resto de nodos, se localiza dentro del FrontEnd en la carpeta /exports, siendo la carpeta /share un enlace simbólico a esta última.

Al principio es posible que no se sincronicen correctamente los archivos introducidos en el directorio debido a que la carpeta inicialmente está vacía, por lo que para solucionar este inconveniente puede usarse el comando siguiente para reiniciar dicho servicio:

```
$cd /
$export fs –rv
/service nfs restart
```

3.4 Operaciones básicas de Rocks

Comandos y aplicaciones esenciales para el manejo del clúster:

Ver que rolls están instalados:

$rocks list roll

Visualizar el estado del sistema, Figura 3.18.:

$qstat –f

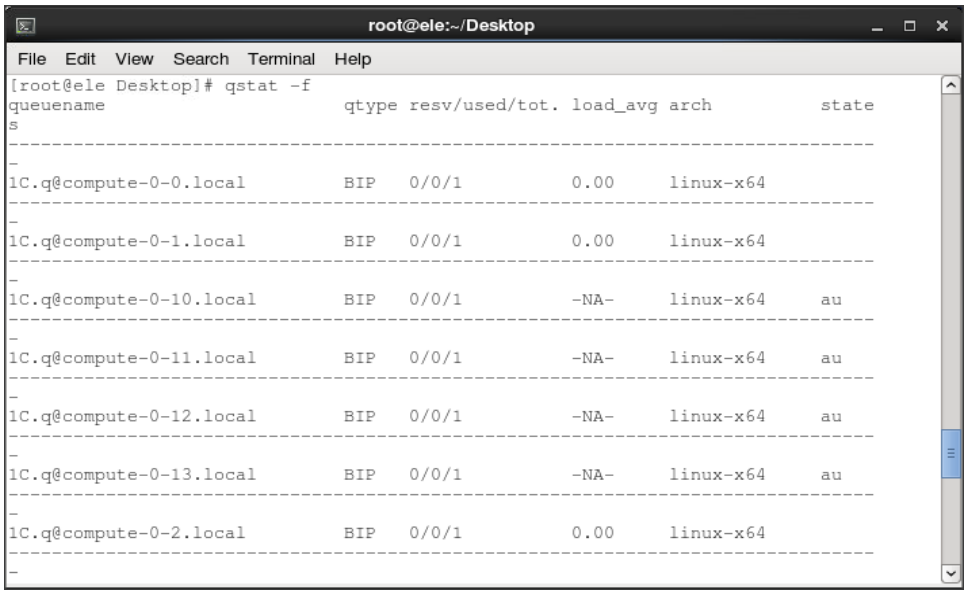


Figura 3.18. Comando qstat –f. (Fuente: Elaboración propia.)

- Probar el sistema de colas para distribuir tareas a los nodos.

Para ello, se prueba que el SGE del clúster funcione, enviándole una serie de trabajos mediante el comando qsub.

En primer lugar, se crea un fichero (prueba.pl) que tenga permisos de ejecución, con el siguiente contenido:

```
#!/usr/bin/perl

for(my $i=0;$i<30;$i++){ sleep(2) } print "ok!\n";
```

Y se ejecuta con el siguiente comando:

```
$ for ((i=1;i<=10;i+=1)); do qsub prueba.pl; done
```

Al ejecutarse, se puede comprobar que los trabajos se han mandado, Figura 3.19.

```
[root@ele Desktop]# for ((i=1;i<=10;i+=1)); do qsub prueba.pl; done
Your job 1 ("prueba.pl") has been submitted
Your job 2 ("prueba.pl") has been submitted
Your job 3 ("prueba.pl") has been submitted
Your job 4 ("prueba.pl") has been submitted
Your job 5 ("prueba.pl") has been submitted
Your job 6 ("prueba.pl") has been submitted
Your job 7 ("prueba.pl") has been submitted
Your job 8 ("prueba.pl") has been submitted
Your job 9 ("prueba.pl") has been submitted
Your job 10 ("prueba.pl") has been submitted
```

Figura 3.19. Envío de trabajos por medio del comando qsub. (Fuente: Elaboración propia.)

Y al realizar de nuevo un "qstat –f", Figura 3.20, se observa cómo se van realizando:

```
############################################################################
 - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS
############################################################################
      8 0.55500 prueba.pl  root         qw    06/22/2015 01:37:28     1
      9 0.55500 prueba.pl  root         qw    06/22/2015 01:37:28     1
     10 0.55500 prueba.pl  root         qw    06/22/2015 01:37:28     1
[root@ele Desktop]#
```

Figura 3.20. Estado de los trabajos pendientes mostrados por qstat –f. (Fuente: Elaboración propia.)

Para ver la carga de trabajo en el clúster de manera global y de forma gráfica, se tiene que ingresar en un navegador web en la siguiente dirección: http://localhost/ganglia, aparecerá la Figura 3.21.

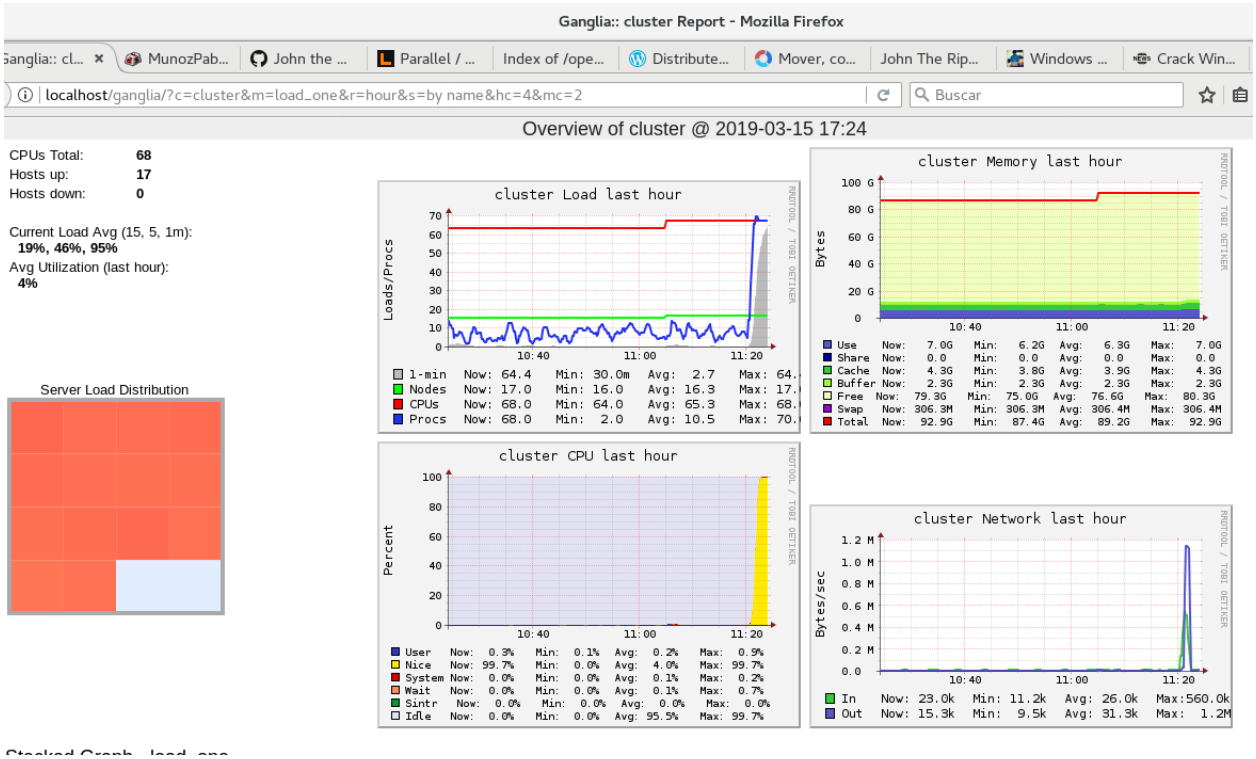


Figura 3.21. Visualización de la distribución de cargas en el clúster, usando la herramienta Ganglia. (Fuente: Elaboración propia.)

Capítulo 4. Metodología y Resultados

Una vez finalizada la instalación y puesta en operación del equipo, se analizaron diferentes metodologías de trabajo, lo que implicó un trabajo arduo debido a que existen complicaciones cuando se buscan temas relacionados a vulnerar sistemas operativos. La realización de este proyecto, está basado en la metodología de Joshua Picolet "Cracking Methodology" (Picolet, 2017).

4.1 Obtener Archivo hash

Se pueden vulnerar contraseñas de Windows de diferentes maneras. Una de ellas es mediante el acceso al archivo del Administrador de cuentas de seguridad (*SAM*), el cual se detalla en el apartado 2.5.3, a través de la obtención de las contraseñas del sistema en su forma *hash*, por medio de herramientas diferentes. Alternativamente, las contraseñas se pueden leer desde la memoria, lo que tiene el beneficio adicional de recuperar las contraseñas en texto plano y evitar el requisito de descifrado. Para comprender los formatos que se observan cuando se descarguen *hashes* del sistema de Windows, se requiere una breve descripción de los diferentes formatos de almacenamiento.

Originalmente, las contraseñas de Windows de menos de 15 caracteres se almacenaban en el formato *hash* de *Lan Manager* (*LM*). Algunos sistemas operativos como Windows 2000, XP y Server 2003 continúan usando estos *hashes* a menos que estén deshabilitados por el usuario. Ocasionalmente, un sistema operativo como Windows Vista puede almacenar el *hash LM* para la compatibilidad con otros sistemas. Según analistas, incluye varias vulnerabilidades, como dividir

la contraseña en dos bloques y permitir que cada uno se descifre de forma independiente. Mediante el uso de tablas *Rainbow*, es trivial descifrar una contraseña almacenada en un *hash LM*, independientemente de la complejidad. Este *hash* se almacena con la misma contraseña calculada en el formato de *hash NT*, como el que se muestra en el siguiente ejemplo.

Ejemplo de un *hash NTLM* con el componente *LM* y *NT*.

Administrator:500:611D6F6E763B902934544489FCC9192B:B71ED1E7F2B60ED5A2EDD28379D45C91:::

Los sistemas operativos Windows más recientes como el Windows 8 y 10 usan el *hash NT*. En términos simples, no hay una debilidad significativa en este *hash* que lo diferencie de cualquier otra función hash criptográfica. Se requieren métodos de descifrado como lo es *la fuerza bruta* o *ataque de diccionario* para recuperar la contraseña, si únicamente se almacena en el formato *NT*.

Ejemplo de un *hash NTLM* únicamente con el componente *NT* (como se ve en los sistemas Windows mas recientes.

Administrator:500:NOPASSWORD**********:EC054D40119570A46634350291AF0F72:::

Es importante señalar que la cadena "NO PASSWORD" es variable según la herramienta que se utilice. Otras pueden presentar esta información como ceros rellenos, o comúnmente puede

ver la cadena “AAD3B435B51404EEAAD3B435B51404EE” en lugar de ninguna contraseña. Esto significa que el *hash LM* está vacío y no está almacenado.

El archivo *hash* se encuentran en el directorio “Windows \ System32 \ config” y se necesitan ambos archivos *SAM* y *SYSTEM*. Además, también se encuentran en el archivo de registro HKEY_LOCAL_MACHINE \ SAM al que no se puede acceder durante el tiempo de ejecución. Finalmente, las copias de seguridad se pueden encontrar a menudo en “Windows \ Repair”. (Peleus, 2019)

Para extraer el archivo *hash* de un sistema operativo con un nivel de seguridad C1, existen diferentes métodos, en esta ocasión se procederá a iniciar el equipo objetivo con una distribución de Linux desde una memoria USB, como se muestra en la figura 4.1, para de esta manera ingresar al directorio C:/WINDOWS/System32/Config, directorio de ubicación de los archivos *SAM* y *SYSTEM* que son los necesarios para obtener el archivo *hash*. (Caballero, 2018)

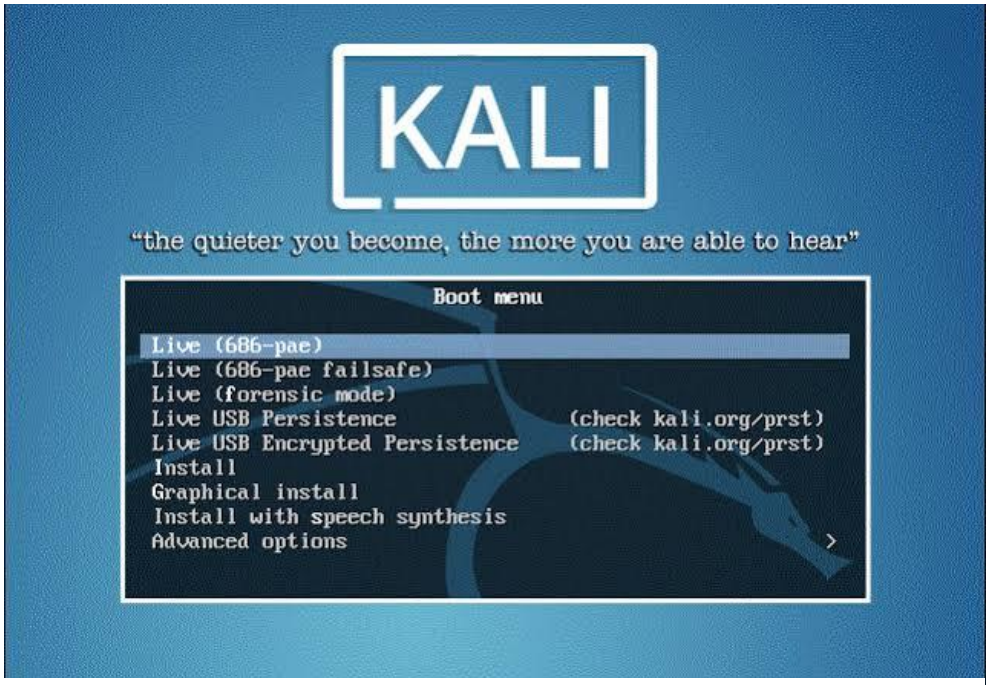


Figura 4.1. Arranque con LiveUSB KaliLinux. (Fuente: Elaboración propia.)

4.2 Dar Formato al Hash

En este punto es donde se tiene que identificar el algoritmo con el que fueron encriptados los archivos *SAM* y *SYSTEM* que se han obtenido en el punto anterior, con la finalidad de seleccionar el tratamiento que se les dará. El algoritmo de encriptación del archivo *SAM* de Windows 10 es *NTLM* y además está almacenado en un formato binario (ByteBleeder, 2017), por lo que fue necesario instalar la herramienta con licencia *GNU samdump2* para obtener la *syskey* y de esta manera poder extraer los *hash* desde el archivo SAM de Windows.

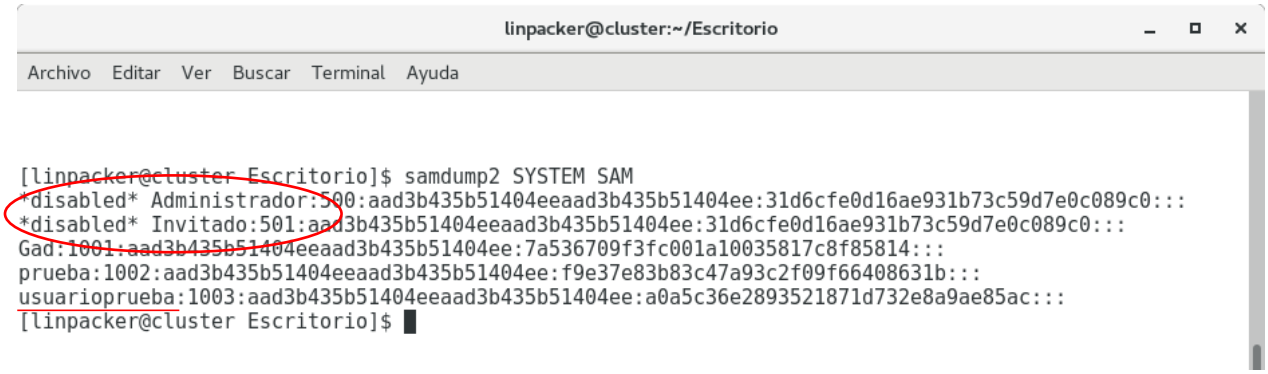


Figura 4.2. Resultado de la ejecución de la herramienta samdump2. (Fuente: Elaboración propia.)

En la figura 4.2 se han obtenido los *hashes* del archivo *SAM*, donde se puede observar que los usuarios Administrador e Invitado están deshabilitados, además de que existen tres cuentas de usuario activas, las cuales son: **Gad**, **prueba** y **usuarioprueba**, cada una con su contraseña cifrada en formato *NTLM* a la derecha del nombre de usuario; la cuenta **usuarioprueba** fue a la que se le realizó el análisis que se detalla posteriormente.

4.3 Evaluar la fuerza del Hash

En criptografía se dice que un algoritmo criptográfico se rompe si por algún método se consigue reducir su fortaleza, es decir reducir por ejemplo su entropía de los 2^{256} bits teóricos a

tan sólo 2^{255} por ejemplo. El hecho de que se encuentre una colisión no implica necesariamente que un algoritmo se haya roto, pues podría haber sido realizado mediante *fuerza bruta*, pero puede ser una causa.

Un ataque de colisión significa que encuentro dos entradas diferentes que producen el mismo resultado de un hash particular. Este es generalmente el tipo de ataque más fácil. Es el tipo de ataque que se conoce desde hace varios años contra MD5.

Dado que una utilidad importante de las funciones hash está en las firmas digitales, encontrar colisiones puede facilitar falsificar firmas digitales. Es decir, dos mensajes distintos que tengan un mismo resumen, tienen además la misma firma y un atacante puede reemplazar un mensaje por el otro. Un ejemplo de este ataque fue demostrado en 2008 por un grupo de investigadores en Ámsterdam los cuales falsificaron un certificado raíz mediante el uso de un ataque de colisión contra el algoritmo MD5. (Preukschat, 2015)

Un ataque previo a la imagen generalmente es mucho más difícil: es encontrar otra entrada que produzca el mismo resultado que un hash que ya se conoce.

Para tratar de aclarar la diferencia aquí: en un ataque de colisión, encuentro dos entradas A y B que producen el mismo resultado. En un ataque de preimagen, comienzo a partir de algún resultado conocido X, y encuentro una entrada que producirá X como resultado. (Coffin, 2015)

Para realizar esta evaluación se tomó como base la tabla del apéndice "Hash craking benchmark" (Picolet, 2017); el objetivo es observar el comportamiento y velocidad del Hash mencionado cuando fue sometido a pruebas de criptoanálisis. Si se trata de un Hash lento, se deberá ser más selectivo en qué tipos de diccionarios y métodos de criptoanálisis se le debe aplicar. Si es un Hash rápido, se puede ser más libre con la estrategia de criptoanálisis. Este estudio busca realizar una prueba de criptoanálisis mostrando las diferencias entre el poder de procesamiento de un clúster de 17 nodos, contra una computadora personal, por lo que no necesariamente se implementará el método óptimo para la realización del criptoanálisis.

4.4 Calcular la capacidad del equipo.

Con la información de la evaluación de la fuerza del Hash obtenida en el apartado anterior, en este apartado se tiene que establecer la capacidad de la plataforma o plataformas que realizarán el criptoanálisis.

Para conocer la capacidad de procesamiento del clúster de alto rendimiento se utilizó la herramienta Linpack Benchmark. HPL es una implementación portátil del Benchmark de alto rendimiento LINPACK para equipos de cómputo distribuido. Actualmente se utiliza para obtener varios resultados en la lista actual de Top500 de HP clústers. (Dongarra, 2010)

El Linpack Benchmark (HPL del Inglés High-Performance Linpack Bench-mark) es un software que resuelve un sistema lineal aleatorio de doble precisión aritmética (64 bits) en equipos de memoria distribuida. El rendimiento muestra el número de operaciones de punto flotante por segundo logradas por el sistema. (Petitet, Whaley, Dongarra, & Cleary, 2008)

El HPL fue desarrollado en el Argone National Laboratory por Jack Don-garra en 1976, y es uno de los más usados en sistemas científicos y de ingeniería para el cálculo de prestaciones. Las principales características del algoritmo utilizado por el HPL son:

- Distribución cíclica de datos de bloques de dos dimensiones.
- Variante derecha de la factorización LU con pivoteo parcial. (Técnica de pivoteo. Dicta que el elemento pivote que debe escogerse es el mayor absolutamente de cada columna.)
- Factorización recursiva con pivoteo y reenvío de columna.
- Distintas topologías virtuales de reenvíos.
- Algoritmo de reducción de reenvío para ancho de banda. (Castelló Gimeno, 2014)

La resolución del sistema de ecuaciones se realiza del siguiente modo:

1. Se crea un sistema de ecuaciones:

$$Ax = b \tag{1}$$

$$A \in R^{nxn} \tag{2}$$

$$x, b \in R^{n} \tag{3}$$

HPL calcula que la solución para un sistema de ecuaciones lineales se puede generalizar como: $Ax = b$, donde A es una matriz $N\ x\ N$ cuyos valores se generan aleatoriamente, $x\ y\ b$ son vectores de tamaño N. Como se muestra en la Figura 4.3.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Figura 4.3. Sistema de Ecuaciones tipo $Ax = b$. (Fuente: Elaboración propia.)

2. Primero se calcula la factorización LU con pivoteo parcial de los coeficientes n y $n + 1$ de la matriz $[A, b]$:

(4)

$$P_r[A, b] = [[LU], y]$$

(5)

$$P_r, L, U \ \in \ R^{nxn}$$

(6)

$$y \ \in \ R^n$$

El primer paso es la factorización de la matriz A en una matriz triangular superior, U, y una matriz triangular inferior, L tal que $A \ = \ LU$. La factorización se usa para calcular el vector solución, x.

3. Una vez el pivoteo (representado por la permutación de la matriz P_r) y la factorización inferior se aplican sobre b, la solución se obtiene en un paso resolviendo el sistema superior triangular:

(7)

$$Ux = y$$

La matriz triangular inferior izquierda y el conjunto de pivotes no se devuelven en el resultado. Para la estabilidad numérica, el algoritmo de factorización utiliza un pivote parcial de fila.

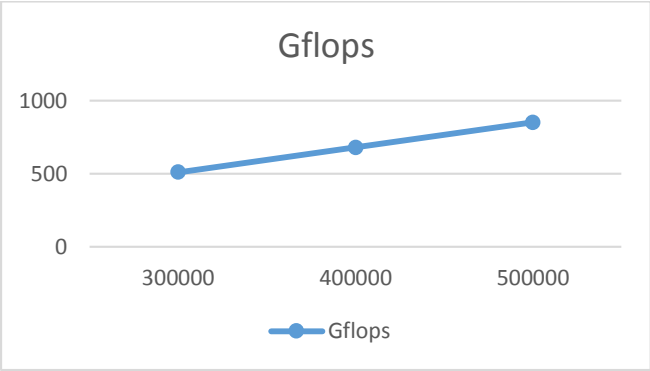
La implementación paralela de Linpack divide la matriz $A: N\ x\ N$ en una cuadrícula de procesos $P\ x\ Q$ bidimensional y luego la divide en subgrupos de mosaicos $NB\ x\ NB$. (Petitet, Whaley, Dongarra, & Cleary, 2008)

Utilizando la herramienta HPL en esta investigación se obtuvieron los siguientes resultados:

La prueba abordó el ajuste de N y la eficiencia obtenida en comparación con el rendimiento máximo. Se determinó un valor para $N = 800{,}000$ aleatoriamente para ejecutar HPL en todo el clúster. Sin embargo, el tiempo fue excesivo, por lo tanto, se determinó que este tamaño del problema fue demasiado grande para el clúster. Al realizar otros experimentos con valores más pequeños de N se obtuvieron mejores resultados de rendimiento. La Tabla 4.1 muestra el rendimiento de HPL en GFlops para $N = 300{,}000$, $400{,}000$ y $500{,}000$. Como se puede observar en la Tabla 4.1, se obtuvo el mejor resultado de rendimiento, 850 GFlops para $N = 500{,}000$; para valores mayores de N la distribución de cargas del clúster empieza a sobrepasar su capacidad como se muestra en la Figura 4.4, y la vez, su rendimiento disminuye. Esto ocurre porque la cantidad de memoria utilizada con las memorias intermedias de comunicación MPI es mayor de lo esperado. Por lo tanto, es necesario dejar más del 20% de la memoria para el sistema. (Petitet, Whaley,

Dongarra, & Cleary, 2008) También se realizaron pruebas usando solo un nodo de cómputo y el rendimiento fue de 34 GFlops para $N = 20{,}000$.

Tabla 4.1. Rendimiento de HP clúster. (Fuente: Elaboración propia.)



Analizando la eficiencia obtenida para la ejecución de un solo nodo, cuando $N = 20{,}000$, se obtuvo el 82.3% del rendimiento máximo (el rendimiento máximo para un solo nodo es 47.7 Gflops). Para todo el clúster, el rendimiento máximo sería: $Tmax = 47.7\ x\ 17 = 810.9\ Gflops$ aproximadamente.

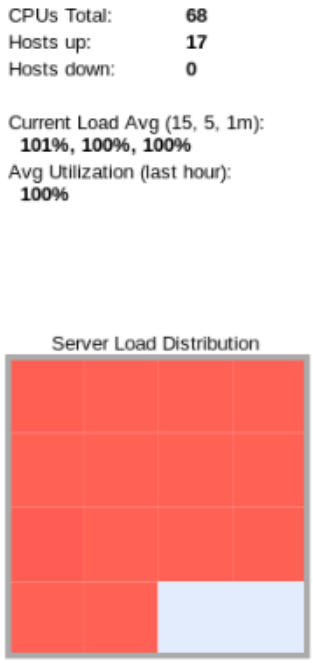


Figura 4.4. Distribución de cargas del clúster.

La Figura 4.4 titulada “Distribución de cargas del clúster”, muestra la imagen de la herramienta Ganglia de manera gráfica, a través de cuadros que varían de color dependiendo de la carga que tenga cada uno de los nodos al momento de implementar la herramienta HPL. De manera inicial indica que el clúster tiene un total de 17 equipos, de 4 cores cada uno, eso da un total de 68 cores trabajando, sin equipos inactivos. Muestra también una sobrecarga en el clúster y un porcentaje promedio de 100 % de uso en la última hora.

4.5 Establecer Estrategia

Esto significa únicamente que en este apartado se realizará una guía básica para procesar los *hashes* obtenidos, sin perder de vista que cada escenario a criptoanalizar es único en función de las circunstancias externas. Como ya se ha mencionado anteriormente, esta estrategia de criptoanálisis se ha planteado implementar dos métodos, los cuales son *Fuerza Bruta* y *ataque de diccionario*, sin importar si son o no los óptimos, debido a que se realizará una comparativa de capacidad de procesamiento.

4.5.1 Fuerza Bruta. Una vez que se han intentado varios métodos y han fallado, se puede iniciar un ataque de fuerza bruta estándar, poniendo especial atención en la cantidad de símbolos del alfabeto que puede usar el equipo a analizar. Por encima de 8 caracteres, esto generalmente no tiene sentido debido a limitaciones de hardware y entropía (complejidad de la contraseña). Una contraseña de Windows según Microsoft (Microsoft, Inc., 2019) una contraseña de Windows debe está formada por:

26 Caracteres mayúsculas de la “A” a la “Z”

26 Caracteres minúsculas de la “a” a la “z”

10 Dígitos de base 10 (0 a 9)

32 Caracteres especiales (~!@#$%^&*_-+=|(){}\[]:;"'<>,.?/)

94 Caracteres alfanuméricos en total.

4.5.1.1 Cantidad de combinaciones posibles. Si se usa una clave numérica (números del 0 al 9) existen 10 posibilidades de 4 dígitos de longitud, que es la extensión por defecto de los teléfonos celulares. Para 4 dígitos, las combinaciones posibles serían 10 mil. $10^4 = 10{,}000$. (Labaca Castro, 2014)

Por otro lado, si tomamos un alfabeto de 26 letras mayúsculas y 26 minúsculas, 10 números y 10 caracteres especiales, en total tenemos 72 diferentes posibilidades para un único carácter. Eso significa que las combinaciones para una contraseña de 4 caracteres serían: $72^4 = 26{,}873{,}856$. Básicamente, más de 26 millones de combinaciones en una clave de 4 caracteres contra 10 mil en una clave de 4 dígitos numéricos. En este estudio se utiliza la contraseña de bloqueo descrita en el apartado anterior que es de 94 combinaciones para cada carácter, teniendo una longitud de 7 caracteres, por lo tanto, serán: $94^7 = 64{,}847{,}759{,}000{,}000$ de posibles combinaciones que nuestros equipos deben explorar para obtener la contraseña correcta.

Como se mencionó en el apartado 4.2 el usuario sobre el cual se realizó el criptoanálisis fue **usuarioprueba**, el cual constaba de una contraseña de longitud 7 caracteres, la cual, después de realizado el análisis, y como se puede observar en la Figura 4.5:

1. Análisis a usuario: **usuarioprueba**
2. Número de nodos en operación: 17
3. Número de procesos ejecutándose de forma paralela: 68
4. Tiempo de duración del análisis en el clúster: 12:27:27 horas
5. Contraseña criptoanalizada: ***Pass#7**.

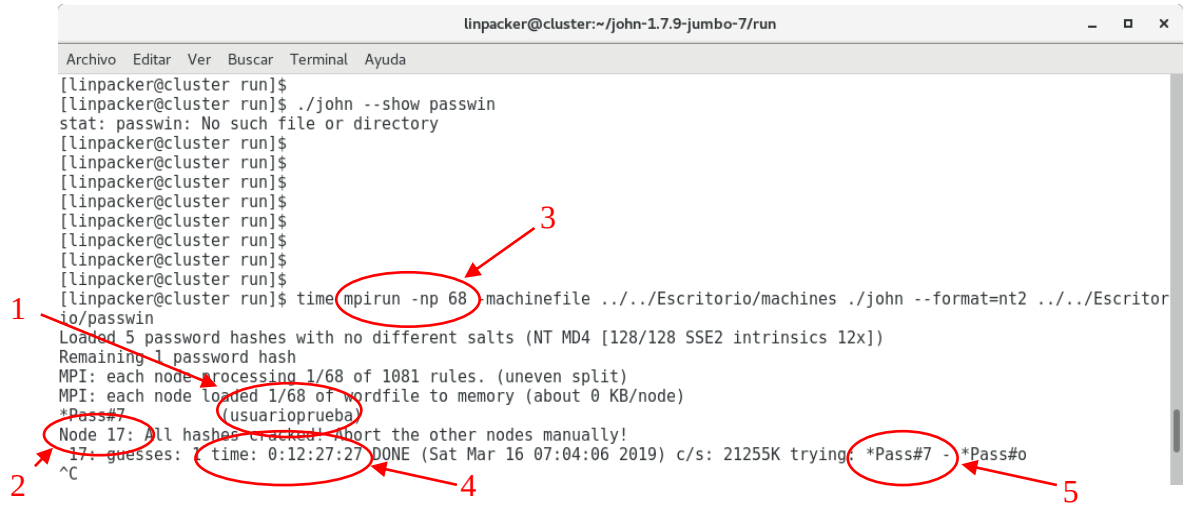


Figura 4.5. Resultado de criptoanálisis realizado con el clúster. (Fuente: Elaboración propia.)

Así mismo en la Tabla 4.2 se muestra el resultado final y el tiempo de duración del criptoanálisis realizado al archivo SAM con el clúster, y la computadora personal.

Tabla 4.2. Resultados de criptoanálisis de archivo SAM. (Fuente: Elaboración propia.)

Nodos	Núcleos	Tiempo de Procesamiento (Horas)
1 (PC)	4	107:53:16
17 (Clúster)	68	12:27:27

4.5.2 Diccionario. Método empleado para romper la seguridad de los sistemas basados en contraseñas (password) en la que el atacante intenta dar con la clave adecuada probando todas (o casi todas) las palabras posibles ingresadas previamente en un diccionario idiomático. Generalmente se emplean programas especiales que se encargan de ello.

Este método suele ser más eficiente que el método de fuerza bruta, debido a que muchas usuarios casi siempre suelen utilizar contraseñas con una o varias palabras existentes en su lengua, para que la contraseña no sea olvidada fácilmente y sea sencilla de recordar, esto no es una práctica recomendable. (Alegsa, 2018)

Para realizar este método se utilizó un diccionario descargado de internet el cual contiene 10,000,000 de posibles contraseñas. (g0tmi1k, 2018).

Además, se ejecutó un programa creado en lenguaje C (Anexo 1), a través de la implementación de las librerías MPI y open MPI propias de la arquitectura de clúster de computadoras. Este programa fue ejecutado en los mismos equipos descritos anteriormente para obtener todos los números primos encontrados dentro de un intervalo dado. En este caso el intervalo dado fue de 10,000,000, con los siguientes resultados mostrados en la Tabla 4.3.

Tabla 4.3. Resultados de ejecución de programa en C para la obtención de números primos. (Fuente: Elaboración propia.)

Nodos	Núcleos	Tiempo de Procesamiento (Segs.)
1 (PC)	4	5.078
17 (Clúster)	68	0.0013

4.5.3 Analizar los resultados. Posterior a realizar el criptoanálisis, y descifrar con éxito una cantidad suficiente de hashes, se analizan los resultados en busca de pistas o patrones. Este análisis puede ayudar a tener éxito en cualquier hash restante.

4.5.4 Personalizar ataques. Tomando como base el análisis de resultados realizado previamente, se diseñarán diversas estrategias personalizadas que aprovechen las vulnerabilidades conocidas o patrones. Los ejemplos serían ataques de máscara personalizados o reglas para adaptarse al comportamiento o las preferencias del usuario objetivo.

Conclusión

El trabajo de investigación presentado parte de la necesidad de buscar alternativas para explotar al máximo la capacidad de procesamiento del equipo de cómputo que existe actualmente dentro del inventario del Instituto Tecnológico de Acapulco, así como proveer una plataforma que coadyuve a la investigación dentro del Instituto, a través de implementar un clúster de cómputo con capacidad para realizar proyectos en el área de ingeniería y matemáticas con el fin de optimizar el tiempo de ejecución, lo cual en un futuro se verá reflejado en ventajas para el desarrollo de la región.

La revisión y análisis de información en relación a clústers, es muy interesante, pero a la vez es compleja, dado que involucra diferentes aspectos durante su evolución, así como diversas metodologías para su formación e implementación. Se considera de suma importancia para este trabajo; las revisiones efectuadas, ya que son la base para la propuesta entregada.

De acuerdo a las especificaciones técnicas de los equipos informáticos existentes dentro del laboratorio de la Maestría en Sistemas Computacionales, y una vez que se realizó el análisis de varios casos de éxito, se propuso la metodología para implementación de un clúster homogéneo tipo *Beowolf*. Este método está basado en otros que ya se han implementado bajo realidades académicas similares, rescatando las etapas más importantes de cada uno y complementándolo con los aportes personales; se consideró que fue el óptimo debido a que se implementó con hardware convencional y herramientas de uso libre.

El hecho de haber seleccionado un método adecuado, permitió superar gran parte los problemas que presentaban las metodologías que fueron analizadas, simplificó la implementación del clúster, y permitió el poder alcanzar una mayor estabilidad en su desempeño desde su etapa inicial, fue importante conocer el resultado en cada una de las etapas de implementación para asegurar su desenvolvimiento en el futuro.

Para integrar los equipos independientes a la arquitectura clúster, se realizó el diseño de una topología de red Gigabit Ethernet tipo estrella con un direccionamiento IP estático, la configuración de protocolos de comunicación remota sin autenticación, la compartición de archivos en red, y la utilización de aplicaciones de licencia GPL que permiten obtener el funcionamiento similar a la de una supercomputadora.

Haciendo uso de librerías de paso de mensajes (MPI) y el software *John the Ripper* para el descifrado de contraseñas, la arquitectura clúster implementada alcanzó una eficiencia de 88.5% de rendimiento, en comparación con los recursos computacionales de un solo equipo con las características de un nodo. Además, otros de los proyectos ejecutados en la arquitectura clúster para probar el rendimiento, fue la ejecución de un programa para la detección de números primos dentro de un archivo de 10,000,000 de números; obteniendo una eficiencia de más de 99 %, gracias a la distribución de tareas que proporcionan las librerías de paso de mensajes (MPI) dentro de la arquitectura clúster.

Con la implementación de la arquitectura clúster, el laboratorio de la Maestría en Sistemas Computacionales podrá dar soluciones potentes, eficientes y económicas en los entornos de

investigación, mejorando la capacidad de procesamiento en aplicaciones como evaluación de algoritmos, compresión de archivos, renderización de imágenes, análisis de datos (big data), en un tiempo relativamente bajo, ideal para las necesidades educativas.

La experiencia obtenida al realizar esta investigación y propuesta ha sido enriquecedora, tanto por la obtención de conocimiento en base a la historia, evolución y arquitectura de clústers, así como la programación paralela y sus ventajas en diferentes proyectos desarrollados en todo el mundo. Igualmente, la comparación entre cada uno de los modelos y métodos resulta interesante dado que existe un sinnúmero de ejemplos con características diferentes, pero básicamente con un mismo objetivo final.

Trabajo Futuro

El clúster implementado en el Laboratorio de la Maestría en Sistemas computacionales del Instituto Tecnológico de Acapulco puede ser utilizado en gran cantidad de aplicaciones dentro de las diferentes áreas de investigación del Instituto, así como estar disponible para el Departamento de Posgrado e Investigación como apoyo durante la realización de tareas y proyectos dentro de las áreas de Desarrollo de Sistemas Inteligentes, así como de Tecnologías Web.

Propuesta de Trabajo a Futuro

Durante la realización de este proyecto se observó la necesidad que existe dentro del departamento de Investigación y Posgrado de equipos de alta capacidad de procesamiento para realizar de una manera más eficiente las investigaciones que se están llevando a cabo. Una de ellas es el “Sistema de apoyo al diagnóstico oportuno de Retinopatía Diabética” en la cual, a través del entrenamiento de una red neuronal, se clasificarán y evaluarán un número de 3000 imágenes para

demostrar la presencia de dicha enfermedad en pacientes del Instituto Estatal de Oftalmología. Actualmente se ha realizado una prueba de entrenamiento de dicha red neuronal con 70 imágenes, obteniendo como resultado, un tiempo clasificación y evaluación de 30:15.32 horas en un equipo con un procesador Core I7 con 8 GB en memoria RAM.

Bibliografía

Alegsa. (01 de 01 de 2018). *Alegsa, Diccionario Informático*. Recuperado el 07 de 2019, de Alegsa: http://www.alegsa.com.ar/Dic/ataque_por_diccionario.php

Avecto. (2017). *Avecto Limited.* Recuperado el 10 de 08 de 2018, de AVECTO a Bomgar Company: www.avecto.com

Borghello, C. (2009). *Seguridad de la Información*. Recuperado el 11 de 2018, de Criptografía de la A-Z: https://www.segu-info.com.ar/proyectos/p1_hash.htm

Branch Bedoya, J., & Mesa Múnera, A. (Diciembre de 2008). Implementación de un cluster homogéneo para la resolución de problemas de alta complejidad computacional . *Avances en Sistemas e Informática, 5*(3).

ByteBleeder. (02 de 06 de 2017). *Practical guide to NTLM Relaying in 2017*. Obtenido de Github: https://byt3bl33d3r.github.io/practical-guide-to-ntlm-relaying-in-2017-aka-getting-a-foothold-in-under-5-minutes.html

Caballero, A. (26 de 06 de 2018). *Hacking Ético*. Recuperado el 03 de 2019, de John The Ripper: http://www.reydes.com/d/?q=Recuperar_la_Contrasena_del_Administrador_utilizando_samdump2_y_John_The_Ripper

Carretero Pérez, J., de Miguel Anasagasti, P., García Carballeira, F., & Pérez Costoya, F. (2001). *Sistemas Operativos. Una visión aplicada.* (C. F. Madrid, Ed.) Madrid, España: McGRAW-Hill/Interamericana.

Castelló Gimeno, A. (30 de 07 de 2014). *Universitat Jaume.* Recuperado el 10 de 2019, de Evaluación de prestaciones mediante laaplicación HPL de clusters utilizandorCUDA:

http://repositori.uji.es/xmlui/bitstream/handle/10234/115082/TFM_2014_CastelloGimenoA.pdf?sequence=1&isAllowed=y

Chávez, J., Echeverría, C., Correa, M., Vázquez, E., Suárez, M., Mármol, X., . . . Díaz, G. (07 de 1999). *Centro de Cálculo Científico* . Recuperado el 01 de 05 de 2018, de Universidad de Los Andes: http://www.cecalc.ula.ve/documentacion/tutoriales/beowulf/informe.html

Chirinov, R. (2003). *Nfinite9000 S.L.* Recuperado el 26 de 04 de 2018, de Noticias 3D: https://www.noticias3d.com/articulo.asp?idarticulo=248&pag=4

Coffin, J. (23 de 07 de 2015). *Quora*. Recuperado el 15 de 08 de 2019, de https://www.quora.com/What-does-it-mean-when-a-hashing-algorithm-has-been-broken

Dongarra, J. (2010). *Top500 Super computer sites*. Recuperado el 07 de 2019, de http:// www.top500.org

Educba. (01 de 01 de 2019). Recuperado el 10 de 2019, de Educba.org: https://www.educba.com/idea-algorithm/

europa press. (04 de 06 de 2013). *Portal Tic*. Recuperado el 18 de 11 de 2018, de Europa Press: https://www.europapress.es/portaltic/gadgets/noticia-cuarta-generacion-intel-core-mayor-salto-cuanto-ahorro-energetico-20130604174722.html

g0tmi1k. (2018). *github*. Recuperado el 05 de 2019, de github: https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10-million-password-list-top-1000000.txt

González Agudelo, D. (2014). *El riesgo y la falta de politicas de seguridad informática una amenaza en las empresas certificadas basc*. Recuperado el 20 de 05 de 2018, de Universidad Militar Nueva Granada: https://repository.unimilitar.edu.co/bitstream/10654/12251/1/ENSAYO%20FINAL.pdf

González Ramírez, E., & Rodríguez Sánchez, A. (2008). *Diseño e implementación de un cluster tipo BEOWULF para el desarrollo de cómputo científico avanzado.* CdMx, Mexico.

Hernandez, M. (2013). *Forensics Mexico*. Recuperado el 07 de 2018, de La importancia del registro de Windows parte 1: http://forensicsmexico.blogspot.com/2013/09/la-importancia-del-registro-de-windows.html

Kowalczyk, C. (01 de 2015). *Crypto-IT*. Recuperado el 11 de 2018, de Crypto-IT: http://www.crypto-it.net/eng/asymmetric/index.html

Labaca Castro, R. (13 de 06 de 2014). *We live security*. Recuperado el 2019 de 08 de 07, de Welivesecuritybyeset: https://www.welivesecurity.com/la-es/2014/06/13/matematica-claves-numerica-alfanumerica/

Lizárraga, C. (2002). *Departamento de Física*. Recuperado el 12 de 03 de 2018, de Universidad de Sonora: http://clusters.fisica.uson.mx/

Llorens, E., & Peña, M. (2002). Recuperado el 04 de 2018, de Departament d'Arquitectura de Computadors: http://personals.ac.upc.edu/enric/PFC/Beowulf/beowulf.html

Menéndez-Barzanallana, R. (07 de 08 de 17). *Universidad de Murcia*. Recuperado el 11 de 2018, de Informática Aplicada a las Ciencias Sociales: https://www.um.es/docencia/barzana/IACCSS/Criptografia.html

Merkle, R., & Damgård, I. (s.f.). *Numerentur.org*. Recuperado el 06 de 2019, de Para computar - numerentur.org: http://numerentur.org/funcion-merkle-damgard/

Microsoft, Inc. (02 de 05 de 2019). *www.microsoft.com*. Obtenido de Password must meet complexity requirements: https://docs.microsoft.com/es-es/windows/security/threat-protection/security-policy-settings/password-must-meet-complexity-requirements

Mifsud, E. (26 de 03 de 2012). *Introducción a la seguridad informática.* Recuperado el 07 de 05 de 2018, de Ministerio de educación, cultura y deportes, Gobierno de España.: http://recursostic.educacion.es/observatorio/web/es/software/software-general/1040-introduccion-a-la-seguridad-informatica?showall=1

Murugesan, S. (01 de 02 de 2008). *Researchgate.* doi:10.1109/MITP.2008.10

Nahar, K., & Abu Abbas, O. (01 de 2010). *Researchgate*. Recuperado el 07 de 2019, de Researchgate: https://www.researchgate.net/publication/315492522_Hybrid_Cipher_System

Nievas, F., Pino, O., & Arroyo, R. (s.f.). *Los clusters como Plataforma de Procesamiento Paralelo.* Recuperado el 25 de 04 de 2018, de http://usuarios.lycos.es/lacaraoculta/descargas/Clusters_definitivo. pdf

Pawar, S. (2016). *A computer science portal for geeks*. Recuperado el 10 de 2018, de Geeksforgeeks: https://www.geeksforgeeks.org/rc5-encryption-algorithm/

Peleus. (2019). *NETSEC*. Recuperado el 08 de 2019, de https://netsec.ws/?p=314

Pérez Tavera, I. (2014). Obtenido de Universidad Autonoma del Estado de Hidalgo: https://www.uaeh.edu.mx/scige/boletin/prepa4/n2/e4.html

Pérez, M. (2001). Recuperado el 15 de 04 de 2018, de http://www.dragones.org/Biblioteca/Articulos/ArquitecturaParalela2.pdf

Petitet, A., Whaley, R. C., Dongarra, J., & Cleary, A. (2008). *HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers*. Recuperado el 07 de 2019, de http://www.netlib.org/benchmark/hpl/

Picolet, J. (2017). *Hash Crack: Password Cracking Manual* (Vol. 2). Createspace Independent Pub.

Preukschat, A. (23 de 02 de 2015). *Oro y Finanzas*. Recuperado el 07 de 2019, de https://www.oroyfinanzas.com/2015/02/que-es-colision-criptografia-bitcoin/

Robbins, D. (2008). *OpenMosix*. Recuperado el 22 de 04 de 2018, de the openMosix Project: http://openmosix.sourceforge.net/introduction_to_openmosix.html

Rocha Quezada, J. d., Botello Rionda, S., Vargas Félix, J. M., & Munguía Torres, I. A. (Sep-Dic de 2011). Diseño e implementación de un clúster de cómputo de alto rendimiento. *Acta Universitaria, 21*(3), 24-33. Recuperado el 1 de 12 de 2018

Rocksclusters. (01 de 01 de 2010). *http://www.rocksclusters.org*. Recuperado el 01 de 06 de 2018, de http://www.rocksclusters.org: http://www.rocksclusters.org

Sanjoy, K. (01 de 01 de 2015). *DZone*. Recuperado el 15 de 06 de 2019, de DZone: https://dzone.com/articles/security-algorithms-des-algorithm

Security Degree Hub. (01 de 2017). *Security Degree Hub*. Recuperado el 13 de 02 de 2019, de Security Degree Hub: https://www.securitydegreehub.com/what-is-cryptanalysis/

Simmons, G. (01 de 01 de 2019). *Encyclopedia Britannica*. Recuperado el 08 de 2019, de Encyclopedia Britannica: https://www.britannica.com/topic/RSA-encryption

statista. (09 de 2017). *Statista*. Obtenido de Statista Co.: https://es.statista.com

Turner, D. (2004). Recuperado el 12 de 02 de 2018, de Ames Lab: http://www.scl.ameslab.gov/Projects/parrallel_computing/para_into.html

United States Government Department of Defense. (1983). *Trusted Computer System Evaluation Criteria.* Rainbow Series.

Universidad Internacional de Valencia. (09 de Septiembre de 2016). *Nuestros expertos,Tecnología*. Recuperado el 27 de Abril de 2018, de Universidad Internacional de Valencia: https://www.universidadviu.es/la-seguridad-informatica-puede-ayudarme/

Wallen, J. (19 de 01 de 2019). *Lifewire*. Recuperado el 13 de 06 de 2019, de Lifewire: https://www.lifewire.com/gnome-3-desktop-4175640

Yang, H. (01 de 02 de 2019). *Herong Yang*. Recuperado el 15 de 06 de 2019, de Cryptography Tutorials: http://www.herongyang.com/Cryptography/DSA-Introduction-What-Is-DSA-Digital-Signature-Algorithm.html

Anexo 1

Programa en Lenguaje C para la obtención de números primos, utilizando librerías mpi.

```
# include <cstdlib>
# include <iostream>
# include <iomanip>
# include <cmath>
# include <ctime>
# include "mpi.h"

using namespace std;
int main ( int argc, char *argv[] );
int prime_number ( int n, int id, int p );
void timestamp ( );

int main ( int argc, char *argv[] )
{
 int i;
 int id;
 int master = 0;
 int n;
 int n_factor;
 int n_hi;
 int n_lo;
 int p;
 int primes;
 int primes_part;
 double wtime;
 n_lo = 1;
 n_hi = 10000000;
 n_factor = 2;

 MPI::Init ( argc, argv );
 p = MPI::COMM_WORLD.Get_size ( );
 id = MPI::COMM_WORLD.Get_rank ( );

        if ( id == master )
        {
                cout << "\n";
                cout << "'Cuenta primos\n";
                cout << " C++/MPI version\n";
                cout << "\n";
                cout << " Programa para contar cantidad de primos para un N dado.\n";
                cout << " Corriendo en " << p << " procesos\n";
                cout << "\n";
```

```
            cout << " N S Tiempo\n";
            cout << "\n";
        }

        n = n_lo;
        while ( n <= n_hi )
        {
            if ( id == master )
            {
                wtime = MPI::Wtime ( );
            }

            MPI::COMM_WORLD.Bcast ( &n, 1, MPI::INT, master );
            primes_part = prime_number ( n, id, p );
            MPI::COMM_WORLD.Reduce ( &primes_part, &primes, 1, MPI::INT,
MPI::SUM,
            master );

            if ( id == master )
            {
                wtime = MPI::Wtime ( ) - wtime;
                cout << " " << setw(8) << n
                << " " << setw(8) << primes
                << " " << setw(14) << wtime << "\n";
            }

            n = n * n_factor;
        }

        MPI::Finalize ( );

        if ( id == master )
        {
            cout << "\n";
            cout << "PRIME_MPI - Procesos maestro:\n";
            cout << " Finalizacion del calculo normal.\n";
        }
        return 0;
}

int prime_number ( int n, int id, int p )
{
 int i;
 int j;
 int prime;
 int total;
```

```
 total = 0;
        for ( i = 2 + id; i <= n; i = i + p )
        {
                prime = 1;
                for ( j = 2; j < i; j++ )
                {
                        if ( ( i % j ) == 0 )
                        {
                                prime = 0;
                                break;
                        }
                }
                total = total + prime;
        }
 return total;
}

void timestamp ( )
{
 # define TIME_SIZE 40
 static char time_buffer[TIME_SIZE];
 const struct tm *tm;
 size_t len;
 time_t now;
 now = time ( NULL );
 tm = localtime ( &now );
 len = strftime ( time_buffer, TIME_SIZE, "%d %B %Y %I:%M:%S %p", tm );
 cout << time_buffer << "\n";
 return;
 # undef TIME_SIZE
}
```