Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

SISTEMA DISTRIBUIDO DE ALMACENAMIENTO DE ARCHIVOS BASADO EN SECRETO COMPARTIDO

Verónica Agustín Domínguez
Instituto Politécnico Nacional
vagustin@ipn.mx

Axel Ernesto Moreno Cervantes
Instituto Politécnico Nacional
axelernesto@gmail.com

Armando Tonatiuh Ávalos Bravo
Escuela Superior de Ingeniería Química e Industrias Extractivas
avalosarma@hotmail.com

Abstract

En este trabajo se presenta el desarrollo de un sistema de almacenamiento de archivos distribuido capaz de proveer los servicios de seguridad de confidencialidad, integridad, autenticación y disponibilidad a la información almacenada, a través de un esquema de compartición de secretos. El sistema se desarrolló mediante la metodología de desarrollo SDL de Microsoft, la cual considera la seguridad como una parte importante dentro del proceso de desarrollo.

Palabras clave: Criptografía Simétrica, Sistema Distribuido, Secreto Compartido, Repositorio de archivos

Los sistemas de almacenamiento de archivos son un servicio cada vez más utilizado por usuarios comunes y grandes compañías para almacenar y compartir archivos (Geel, 2023). Sin embargo, la información almacenada en estos sistemas es vulnerable a diversos ataques. Una forma de proteger la información es mediante el cifrado, pero es importante gestionar adecuadamente las llaves de cifrado.

La disponibilidad de la información también puede verse afectada por la corrupción de datos o fallas en los servidores donde se almacena la información. Los principales proveedores comerciales de almacenamiento de archivos utilizan el cifrador por bloques AES y la transferencia de información se realiza a través de una conexión SSL. También se utilizan esquemas de seguridad extra, como la verificación en dos pasos. Ejemplos de proveedores son Dropbox, Google Drive y MEGA.

Con la creciente cantidad de datos y archivos en línea, la seguridad se ha convertido en una preocupación cada vez

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

mayor. Los sistemas de almacenamiento tradicionales, a menudo, no son suficientemente seguros para proteger datos sensibles y confidenciales. Una solución para este problema es el almacenamiento seguro de archivos basado en el secreto compartido.

El almacenamiento basado en el secreto compartido es un método criptográfico que se utiliza para proteger archivos confidenciales en línea. Este método implica dividir un archivo en múltiples fragmentos y luego distribuirlos en diferentes ubicaciones de almacenamiento en línea. Para acceder al documento original, se necesita una clave de acceso única que se genera mediante el secreto compartido.

Algoritmos de cifrado simétricos

Data Encryption Standard (DES) es un algoritmo de cifrado simétrico desarrollado en 1975 por IBM. Se basa en una red Feistel y consta de 16 rondas de operaciones de bits, sustituciones de bits no lineales (S-Box) y operaciones lógicas XOR. El algoritmo recibe 2 entradas: el texto en claro en bloques de 64 bits y la llave secreta con longitud de 56 bits.

AES (Advanced Encryption Standard) es otro algoritmo de cifrado simétrico presentado en 1997 que utiliza bloques de datos de 128 bits. Consta de 11 rondas de operaciones y ofrece tres implementaciones con diferentes longitudes de llave AES-128 AES-192 AES-256.

Funciones Hash

Las funciones hash son funciones matemáticas que convierten una cadena de caracteres de longitud aleatoria, en una cadena binaria de longitud fija, conocida como hash-value. Comúnmente se usa en la criptografía para asegurar la integridad de los datos.

Las funciones hash deben cumplir ciertas propiedades para ser efectivas en la autenticación de mensajes, incluyendo la capacidad de aplicarse a bloques de cualquier tamaño, producir una salida de tamaño fijo y ser relativamente fácil de calcular. También deben ser unidireccionales, lo que significa que es imposible encontrar la cadena original a partir del hash, así también como resistentes a la colisión, que significa que es improbable que dos entradas distintas produzcan el mismo hash.

Secreto compartido

El secreto compartido es un método criptográfico, consiste en dividir una clave de acceso en múltiples partes o fragmentos y distribuirlos en diferentes ubicaciones de almacenamiento en un sistema distribuido. Para acceder a los datos, se debe tener acceso a todas las partes del secreto compartido.

La clave solo se puede reconstruir si se tienen todas las partes del secreto compartido, por lo que es altamente seguro. Hace que sea muy difícil para un atacante acceder a los datos.

Esquema de secreto compartido perfecto

Inventado de forma independiente por Adi Shamir en 1979 (Shamir, 1979) y George Blakley (Blakley, 1979). El proceso de inicialización y compartición de las partes se realiza de la siguiente manera:

Sea un secreto $S \in \mathbb{N}$ y $S \in \mathbb{N}_p$ se plantea dividir dicho secreto en n participantes $S_1, S_2, S_3 \ldots S_n$, para lograr esto

1. Se define un umbral t tal que $t \leq n$ este umbral permite establecer con que cantidad de partes es posible reconstruir el secreto S, de manera

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

que contando con $t-1$ partes no es posible reconstruirlo

2. Se escoge una función polinomial de grado $t-1$, es decir $f(x) = \sum_{i=0}^{t-1} a_i\ x^i = a_0 + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}\ (mod\ p)$ donde: $a_0 = S,\ a_k \in \mathbb{Z}_p$ donde $1 \leq k \leq t-1$.

3. Una vez construido dicho polinomio, se escogen de forma aleatoria $x_i \in \mathbb{Z}_p - \{\emptyset\}$ con $1 \leq i \leq n$

4. Teniendo los x_i se evalúan en la función polinomial $y_i = f(x_i)$ obteniendo los pares ordenados $p\big(x_i, f(x_i)\big)$ o, mejor dicho $p(x_i, y_i)$

5. Se distribuyen los pares ordenados $p(x_i, y_i)$ entre los n participantes

Para el proceso de recuperación

1. Sean el conjunto de partes entregadas a los n participantes.

2. Contando con t puntos, se puede reconstruir la función polinomial haciendo uso de la interpolación de Lagrange, es decir que $f(x)$ se puede expresar de la siguiente manera $f(x) = \sum_{j=1}^{t} y_j\ l_j\ donde\ l_j = \prod_{1 \leq k \leq t, k \neq j} \frac{x - x_k}{x_j - x_k}\ (mod\ p)$.

3. AL realizar las operaciones algebraicas de los $y_j\ l_j\ mod\ p$ y reducir términos se obtendrá la función polinomial de la forma: $f(x) = \sum_{i=0}^{t-1} a_i\ x^i\ = a_0 + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}\ (mod\ p)$ recuperando así el termino $a_0 = S$ y finalmente obteniendo el secreto S.

Algoritmo de dispersión de la información (IDA)

Propuesto por Michael O. Rabin en 1989 . Este algoritmo divide un archivo F de tamaño L en n partes F_i con $1 \leq i \leq n$, cada una de estas partes de tamaño $|F_i| = \frac{L}{m}$ de tal manera que sea posible reconstruir el archivo original teniendo m de estas partes. Este método puede ser visto como un método perteneciente al campo de los códigos detectores y correctores de errores en el que se agregan bits adicionales a un mensaje creando un bloque, de modo que después de la ocurrencia de cualquier error k dentro del bloque, el mensaje aún puede ser reconstruido. IDA trata de igual manera dicho problema, además de lograr una eficiencia óptima en la sobrecarga de datos y la máxima simplicidad en los algoritmos de decodificación y de codificación.

Proceso de compartición:

- Se segmenta un archivo en n partes $F = b_1, b_2, \ldots, b_n$
- Cada b_i es considerado un entero tomado del rango $[0 \ldots B]$.
- Se toma un primo p tal que $B < p$.
- Por lo anterior, F se convierte en una cadena de residuos modulo p, es decir que se encuentran en el campo $\mathbb{Z}_p$.
- Se escoge un m tal que $\frac{n}{m} \leq 1 + \varepsilon$ para un $\varepsilon > 0$ fijo.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

- Se escogen n vectores $a_i = \{a_{i1}, a_{i2}, \dots, a_{in}\} \in \mathbb{Z}_p^m$ tal que $1 \leq i \leq n$ de manera que cualquier subconjunto de m vectores en $\{a_1, a_2, \dots, a_n\}$ sean linealmente independientes.

- El archivo es segmentado en secuencias de tamaño, m es decir: $F = (b_1, \dots, b_m), (b_{m+1}, \dots, b_{2m}), \dots$.

- Se denota $S_1 = (b_1, \dots, b_m) \dots$.

- Para $i = 1, \dots, n$ se define $F_i = c_{i1}, c_{i2}, \dots, c_{\frac{N}{m}}$ donde $c_{ik} = a_i S_k = a_{i1} b_{(k-1)m+1} + \cdots + a_{im} b_{km}$

- La suma de los tamaños $|F_i|$ es $\left(\frac{n}{m}\right) L$

Proceso de recuperación

- Si se tienen m piezas es decir $F_1, \dots, F_m$ es posible reconstruir el archivo.

- Sea $A = (a_{ij})$ para $1 \leq i, j \leq m$ una matriz $m \times m$ en la cual la i-esima fila es a_i.

- De la matriz anterior se observa que:

$$A \cdot \begin{bmatrix} b_1 \\ . \\ . \\ . \\ b_m \end{bmatrix} = \begin{bmatrix} c_{11} \\ . \\ . \\ . \\ c_{m1} \end{bmatrix}$$

y de ahí se desprende que:

$$\begin{bmatrix} b_1 \\ . \\ . \\ . \\ b_m \end{bmatrix} = A^{-1} \cdot \begin{bmatrix} c_{11} \\ . \\ . \\ . \\ c_{m1} \end{bmatrix}$$

- Se denota la i-esima fila de A^{-1} por $(\alpha_{i1}, \dots, \alpha_{im})$ luego de manera general $1 \leq k \leq \frac{N}{m}$, $b_j = \alpha_{i1} c_{1k} + \cdots + \alpha_{im} c_{mk}$ para todo $1 \leq j \leq N$ donde $i = j \bmod m$ y $k = \left[\frac{j}{m}\right]$

- Entonces se invierte A y se reconstruye F .

Robust Computational Secret Sharing

El Esquema Computacional Robusto de Secreto Compartido (Robust Computational Secret Sharing - RCSS) (Krawczyk, 1993), es un esquema de secreto compartido que puede recuperar correctamente un secreto, incluso en presencia de un número delimitado de "shares" corruptos, mientras mantiene la secrecía de la información.

El RCSS hace uso de un código de corrección de errores y de un algoritmo de dispersión de información. Un código de corrección de errores es un algoritmo para expresar una secuencia de números tal que cualquier error que se introduce puede ser detectado y corregido (dentro de ciertas limitaciones) tomando como base los números restantes.

Los algoritmos de dispersión de información proporcionan un método para almacenar información en fragmentos dispersos en múltiples ubicaciones, de modo que la redundancia proteja la información en caso de corrupción de datos, además de que el acceso no autorizado a cualquier fragmento no proporciona información utilizable. Sólo una

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

entidad cuente con todos los fragmentos y con el algoritmo de dispersión original puede montar correctamente la información completa.

El algoritmo de Robust Computational Secret Sharing se expresa con el siguiente pseudocódigo.

```
PROCEDURE Share(X)
10  K ←$ {0,1}^k ; C ←$ Encrypt_K(X)
11  K ←$ Share^PSS(K)
12  C ←$ Share^IDA(C)
13  FOR i ← 1 TO n DO
14      H[i] ← Hash(K[i] C[i])
15      S_i ←$ Share^ECC(H[i])
16  FOR i ← 1 TO n DO
17      X[i] ← K[i]C[i] S_1[i] ··· S_n[i]
18  RETURN X

PROCEDURE Recover(X, j)
20  FOR i ← 1 TO n DO
21      K[i]C[i] S_1[i] ··· S_n[i] ← X[i]
22  FOR i ← 1 TO n DO
23      H[i] ← Recover^ECC(S_i, j)
24  FOR i ← 1 TO n DO
25      IF X[i] ≠ ◊ AND Hash(K[i] C[i]) ≠ H[i]
26          THEN K[i] ← ◊ ; C[i] ← ◊
27  K ← Recover^PSS(K, j)
28  C ← Recover^IDA(C, j)
29  X ← Decrypt_K(C)
30  RETURN X
```

Figura 1: Pseudocódigo del algoritmo Robust Computational Secret Sharing

EL algoritmo cuenta con dos procesos principales, “Share” y “Recover”, ambos se explican a continuacion.

Proceso Share

- Línea 10. Se genera una llave K, tomando un vector aleatorio perteneciente del espacio de llaves compuesto por cadenas binarias de longitud k, posteriormente se cifra el secreto X haciendo uso de un cifrado por bloques, con la llave generada aleatoriamente.

- Línea 11. Se genera el vector K, siendo resultado de aplicar la operación de “Share” () bajo el esquema de secreto compartido de Shamir.

- Línea 12. Se genera el vector C, como resultado de aplicar al secreto cifrado C un algoritmo de dispersión de información (IDA, Information Dispersal Algorithm).

- Línea 13-15. Posteriormente se obtiene el digesto de cada elemento de los vectores K y C, almacenándose en el vector H. Además, se genera el vector S_i compuesto por el vector H_i después de aplicar un código de corrección de errores (Error-Correcting Code), esto con el fin de añadir redundancia a la información.

- Líneas 16-17. Finalmente, se genera el vector X, compuesto por los fragmentos de la llave K, el vector C y el vector S. Los fragmentos del vector X son los que serán distribuidos en el sistema (“shares”).

Proceso Recover

- Líneas 20-21. De cada fragmento (“share”) se extraen sus componentes, obteniendo como resultado los vectores K, C, S.
- Líneas 22-23. Para cada elemento del vector S, se aplica el proceso de recuperación con el código de corrección de errores.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

- Líneas 24-26. Si el vector X contiene información (es distinto de nulo) y si el digesto de la llave K_i con el cifrado C_i no coincide con el digesto H_i, entonces ese fragmento se considera como corrupto.
- Línea 27. Se recupera la llave aplicando el proceso de "Recover" (recuperación) al vector K, que es el que contenía la llave.
- Línea 28. De igual forma se recupera el secreto cifrado C, mediante la operación "Recover" del algoritmo de dispersión de información.
- Línea 29. Por último, se descifra el secreto cifrado C, con la llave K, obteniendo el secreto X original.

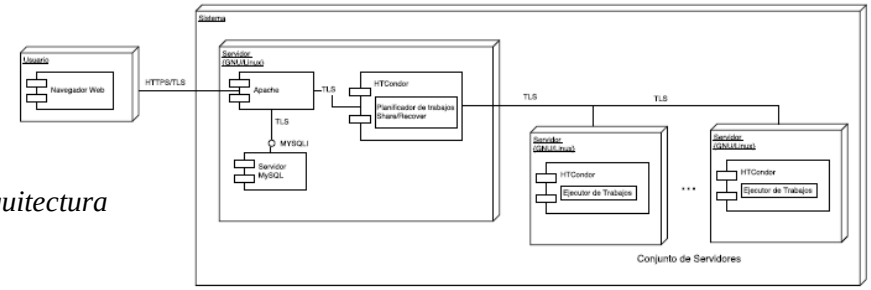

Figura 1. Arquitectura

Metodología

La metodología de desarrollo empleada para este sistema fue Ciclo de Vida del Desarrollo de Seguridad (SDL, *Security Development Lifecycle*) de Microsoft, ya que considera como parte fundamental del desarrollo la seguridad. Esta metodología consta de las siguientes fases:

- **Formación**: formación por parte de los integrantes del desarrollo en componentes básicos de seguridad del sistema (criptografía, secreto compartido, algoritmos de cifrado simétrico, algoritmo IDA, etc.).
- **Requisitos**: se realizó un análisis de requerimientos, dentro de los cuales se consideraron los requisitos de seguridad como parte de la funcionalidad del sistema (confidencialidad, integridad, autenticación y disponibilidad), entre ellos los requisitos de los protocolos de seguridad del sistema (TLS1.2), cifrado y descifrado mediante algoritmo de cifrado por bloques (AES), modos de operación (CTR, CBC), generadores de números pseudoaleatorios (HASH_DRBG, HMAC_DRBG, CTR_DRBG), funciones hash usadas (SHA2 y SHA3), reducción de una superficie de ataques mediante la incorporación de un cortafuegos, restricción de permiso de acceso para archivos de configuración y de registro.
- **Diseño**: Se hizo un análisis de riesgos basado en secreto compartido, entre los riesgos analizados estuvieron los de seguridad, se definieron umbrales de calidad basados en ISO 27001. También se diseñó una arquitectura lógica del sistema propuesto tal como se ve en la figura 1.
- **Implementación**: Se realizó el desarrollo del sistema basado en las etapas de análisis y diseño, para ello, la parte del backend se implementó en su mayoría en lenguaje C++ y el frontend

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

en PHP, para la implementación de los algoritmos criptográficos se usó la biblioteca Crypto++.

- **Comprobación**: Se realizó una revisión de la superficie de ataques (acceso a los servicios del sistema, validación de reglas del cortafuegos, permisos de acceso a los archivos de configuración y registro del sistema, revisión de permisos de ejecución de procesos del sistema).
- **Lanzamiento**: se generó un plan de respuesta a incidentes detectados en la etapa de comprobación.
- **Respuesta**: ejecución del plan de respuesta.

Resultados

Se realizaron pruebas con el objetivo de medir el rendimiento de los procesos de Share y Recover, utilizando diferentes archivos y umbrales de compartición. Las pruebas consisten en medir el espacio consumido por ambos programas tanto en disco como en memoria RAM, medir los tiempos de ejecución en local y en red, además de medir el desempeño de la grid con diferente número de trabajos concurrentes. En cada prueba se especifica el tamaño de los archivos de entrada y los umbrales utilizados.

Las ejecuciones de estas pruebas se realizaron en los equipos participantes del sistema, cada uno de ellos cuenta con el sistema operativo GNU/Linux (x86_64), específicamente con la distribución Ubuntu 16.04, además de tener instalado el software de HTCondor y OpenSSH. Se utilizaron 3 computadoras físicas y 2 máquinas virtuales, las características de cada equipo físico se encuentran listadas en la sección de equipos participantes del sistema.

Espacio consumido

En esta prueba se mide el espacio consumido de los fragmentos de salida generados por la implementación del algoritmo del RCSS. Se utilizaron archivos con un tamaño de 1MB, 10MB, 100Mb y 1.1GB; además de dos umbrales: 2-3 y 3-5.

Tabla 1. Espacio Consumido por los fragmentos para un umbral de 2-3.

Umbral	Tamaño del archivo de entrada	Tamaño total de los fragmentos	Tamaño de *C*	Tamaño de *K*	Tamaño de *S*	Tamaño de cada fragmento *(C+K+S)*
2,3	1 MB	1.5 MB	500.0 kB	24 bytes	264 bytes	500.3 kB
	10 MB	15.0 MB	5.0 MB	24 bytes	264 bytes	5.0 MB
	100 MB	150 MB	50.0 MB	24 bytes	264 bytes	50.0 MB
	1.1 GB	1.7 GB	552.0 MB	24 bytes	264 bytes	552.0 MB

Tabla 2. Espacio Consumido por los fragmentos para un umbral de 3-5.

Umbral	Tamaño del archivo de entrada	Tamaño total de los fragmentos	Tamaño de *C*	Tamaño de *K*	Tamaño de *S*	Tamaño de cada fragmento *(C+K+S)*
3,5	1 MB	1.7 MB	333.3 kB	24 bytes	400 bytes	333.8 kB
	10 MB	16.7 MB	3.3 MB	24 bytes	400 bytes	3.3 MB
	100 MB	166.7 MB	33.3 MB	24 bytes	400 bytes	33.3 MB
	1.1 GB	1.8 GB	368.0 MB	24 bytes	400 bytes	368.0 MB

Mediante esta prueba se comprueba que la implementación del esquema cumple con lo establecido en el Algoritmo de Dispersión de Información IDA, el cual establece que el tamaño de los fragmentos (en este caso C) es igual a L/m, siendo L el tamaño del archivo original y m el umbral de compartición.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

Consumo de memoria

Se utilizó la herramienta massif de valgrind para medir el consumo de memoria de los programas Share y Recover. Valgrind es conjunto de herramientas para realizar pruebas de rendimiento y depuración de memoria en Linux.

Para el caso del programa Share, este consume como máximo de 17.44 MB de memoria, mientras que la implementación de Recover consume como máximo 17.47 MB. En esta prueba se concluyó que el consumo de memoria no es afectado significativamente por el tamaño del archivo ni por el umbral seleccionado.

Tiempo de ejecución

Para estas pruebas, se midió el tiempo de ejecución del programa con archivos de entrada de 1 MB, 10 MB, 100 MB y 1.1 GB, utilizando el comando time de Linux. Se realizaron dos tipos de pruebas: la ejecución en local y la ejecución en red.

Para la ejecución local.

Se realizó la ejecución del algoritmo sin incluir en la medición el tiempo de transferencia de archivos a las demás computadoras. Los archivos tanto de entrada como de salida de los procesos de compartición y recuperación se almacenaron de forma local.

En esta prueba se usaron los umbrales de 2-3, 3-5 y 5-8 para el esquema de secreto compartido. La prueba se realizó en el gestor central de la arquitectura, siendo el tiempo reportado el promedio de 3 mediciones.

Tabla 3. Tiempos de ejecución en local del algoritmo.

Umbral	Tamaño del archivo	Share (tiempo promedio)	Recover (tiempo promedio)
2 - 3	1 MB	0.165 s	0.0653 s
2 - 3	10 MB	1.54 s	0.528 s
2 - 3	100 MB	16.087 s	5.56 s
2 - 3	1.1 GB	3 m 4.215 s	1 m 0.990 s
3 - 5	1 MB	0.189 s	0.0626 s
3 - 5	10 MB	1.798 s	0.526 s
3 - 5	100 MB	17.885 s	5.2973 s
3 - 5	1.1 GB	3 m 38.539 s	1 m 6.860 s
5 - 8	1 MB	0.211 s	0.068 s
5 - 8	10 MB	1.968 s	0.550 s
5 - 8	100 MB	19.756 s	5.698 s
5 - 8	1.1 GB	3 m 39.442 s	1 m 8.092 s

Tiempo de ejecución en red.

Para esta prueba se tomó en cuenta el tiempo necesario para transferir los archivos entre los equipos participantes, tanto de los archivos correspondientes a los fragmentos generados como del archivo recuperado. Esta prueba se realizó en las máquinas físicas del sistema con un umbral de 3-5.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

Tabla 4. Tiempos de ejecución en red del algoritmo.

Tamaño del archivo	Gestor central Share	Esclavo 3 Share	Esclavo 4 Share
1 MB	10.588 s	12.768 s	12.013 s
10 MB	12.344 s	15.303 s	13.974 s
100 MB	30.364 s	31.962 s	31.231 s
1.1 GB	4 m 18.687 s	7 m 28.377 s	6 m 12.785 s

Tamaño del archivo	Gestor central Recover	Esclavo 3 Recover	Esclavo 4 Recover
1 MB	8.839 s	10.210 s	9.766 s
10 MB	9.911 s	12.968 s	11.016 s
100 MB	21.626 s	27.193 s	29.716 s
1.1 GB	1 m 22.134 s	1 m 26.709 s	3 m 7.044 s

Dado los tiempos obtenidos, se observa un aumento en los tiempos de ejecución en red respecto a los tiempos de ejecución en local, esto se debe al tiempo necesario para distribuir y obtener los fragmentos de los demás equipos de la red. Por lo que el tiempo de ejecución del programa en red depende en gran parte de la velocidad de la transferencia de los archivos.

Prueba de ejecución de múltiples trabajos dentro del sistema

En esta prueba se llevó a cabo la ejecución de múltiples instancias del proceso de recuperación en todos los equipos participantes del sistema. La ejecución de estos trabajos se realizó de manera concurrente enviándolos al sistema mediante el planificador de HTCondor. Se utilizaron tamaños de archivo de 1 MB, 10 MB y 100 MB elegidos de manera aleatoria, siendo el tiempo medido el necesario para completar todos los trabajos.

Tabla 5. Tiempos necesarios para completar el número de trabajos especificados.

Número de trabajos	Tiempo
10	1 m 6.823 s
25	1 m 9.531 s
50	2 m 2.721s
75	4 m 48.861 s
100	5 m 56.431 s

Conclusión

Se desarrolló un sistema de almacenamiento de archivos distribuido que provee los servicios de confidencialidad, integridad, autenticación y disponibilidad a la información almacenada en el sistema, a través del uso del esquema de compartición de secretos, conexiones seguras, y un algoritmo de cifrado por bloques simétrico.

El funcionamiento correcto del esquema se comprobó mediante las pruebas realizadas, en las cuales se eliminaban o corrompían fragmentos de la información siendo recuperada correctamente por el esquema, siempre y cuando estuviera dentro de los umbrales establecidos, demostrando así las características de integridad y disponibilidad.

Este sistema es escalable dado que permite añadir nuevos equipos participantes a la grid, adicionalmente la implementación del esquema de Secreto Compartido permite elegir diferentes umbrales de compartición, ajustándose a los equipos disponibles. La escalabilidad del sistema dependerá de la capacidad máxima de los equipos donde se implemente, además de las limitaciones intrínsecas del software de HTCondor y del servidor LAMP.

Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

En el aspecto de seguridad, las comunicaciones entre todas las entidades del sistema se realizan mediante conexiones seguras utilizando TLS/SSL garantizando la confidencialidad del canal y autenticación del servidor; además de seguir las recomendaciones de la metodología SDL como son el uso de funciones seguras, herramientas aprobadas y contraseñas seguras.

Referencias

Blakley, G. R. (1979). Safeguarding cryptographic keys. *1979 International Workshop on Managing Requirements Knowledge (MARK)*, 313-318.

Dropbox. (28 de 2 de 2023). *Dropbox, ¿Cuál es el nivel de seguridad de Dropbox?* Obtenido de https://www.dropbox.com/help/27

Gartner. (28 de 2 de 2023). *Information Dispersal Algorithms*. Obtenido de https://www.gartner.com/it-glossary/information-dispersal-algorithms

Geel, M. (27 de 2 de 2023). *Cloud Storage: File Hosting and Synchronisation 2.0*. Obtenido de https://www.vis.ethz.ch/de/visionen/pdfs/2012/visionen_2012_3.pdf

Google. (28 de 02 de 2023). *Google, Verificación en dos pasos*. Obtenido de https://www.google.com/landing/2step/features.html

Krawczyk, H. (1993). Secret sharing made short. *Advances in Cryptology – CRYPTO'93, 13th Annual International Cryptology Conference*, 136-146.

S, A. T., & Van , M. S. (2008). *Sistemas Distribuidos. Principios y paradigmas*. Mexico: Pearson Educacion.

Shamir, A. (1979). How to Share a Secret. *ACM*, 612-613.

Weisstein, E. W. (28 de 2 de 2023). *Error-Correcting Code. From MathWorld–A Wolfram Web Resource,*. Obtenido de MathWorld--A Wolfram Web Resource: http://mathworld.wolfram.com/Error-CorrectingCode.html