UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3069 - Computación Paralela y Distribuida

Sección 21

Ing. Miguel Novella Linares

Proyecto #2

Programación paralela con MPI

Davis Alvarez, 15842

Juan Solorzano, 18151

Mario Perdomo, 18029

GUATEMALA, 29 de abril de 2022

Planificación de actividades

No.	Semana	Tarea	Descripción
1	24 abr - 30 abr	Investigación Preliminar	Realizar investigación sobre DES
2		Ejecutar codigo base	Se logra correr el algoritmo base y se prepara para su modificación.
3	1 may - 7 may	Explicar rutinas	Se explica y entiende como funcionan las rutinas del programa base.
4		Diagramas de flujo	Se realizan los diagramas necesarios para explicar el funcionamiento del algoritmo.
5		Describir primitiva MPI	Se explican las primitivas de MPI: MPI_Irecb, MPI_Send y MPI_Wait
5	8 may - 14 may	Paralelizar algoritmo	se inicia con la paralelización del algoritmo
6		Realizar pruebas	Se pone a prueba el algoritmo paralelo y se calcula el speedup
7		Resar fenomeno del speedup	Realizar los cambios y pruebas necesarias para comprobar que el algoritmo este funcionando correctamente.
8		Decifrar 2 textos	Decifrar 2 textos de dos equipos diferente y calcular speedup
9		Completar reporte	Se complementa el resporte, discutiendo los resultados, colocando las conclusiones y recomendaciones.
10		Entregar/Presentar	Realizar entrega en canvas

Investigación del protocolo DES

Es un algoritmo de cifrado de bloques que toma texto plano en bloques de 64 bits y los convierte a texto cifrado utilizando claves de 48 bits. DES da como resultado una permutación entre los 2 ^ 64 posibles arreglos de 64 bits, cada uno de los cuales puede ser 0 o 1. Cada bloque de 64 bits se divide en dos bloques de 32 bits cada uno, un medio bloque izquierdo L y un medio derecho R.

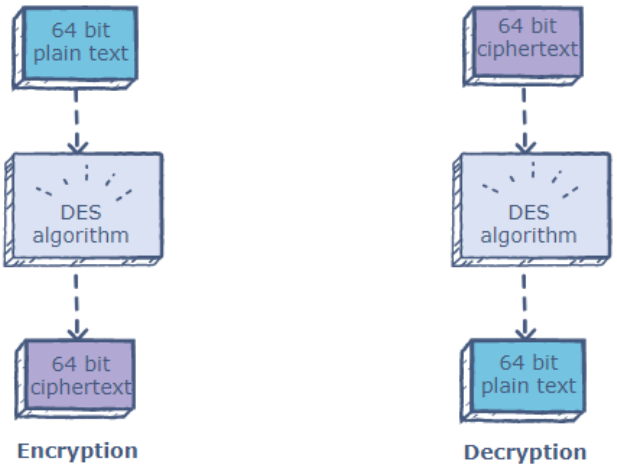


Fig. 1: Diagrama explicando la lógica de DES. (edpresso, 2020).

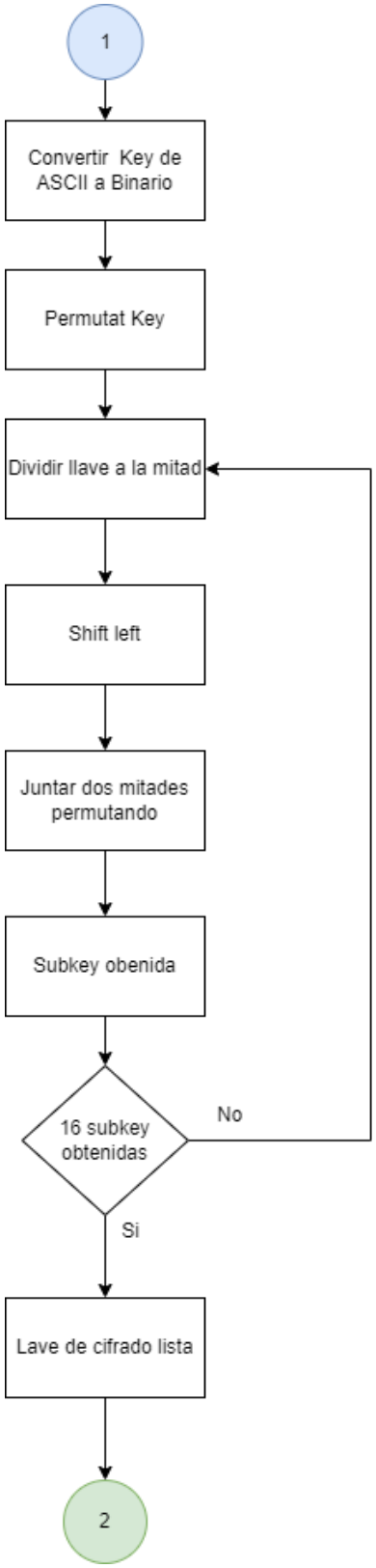
Es un algoritmo de clave simétrica, lo que significa que se utiliza la misma clave para cifrar y descifrar datos. Las keys se almacenan en realidad con una longitud de 64 bits, pero no se utiliza cada octavo bit de la clave (es decir, bits numerados 8, 16, 24, 32, 40, 48, 56 y 64). La longitud real es de 56 bits. (edpresso, 2020).

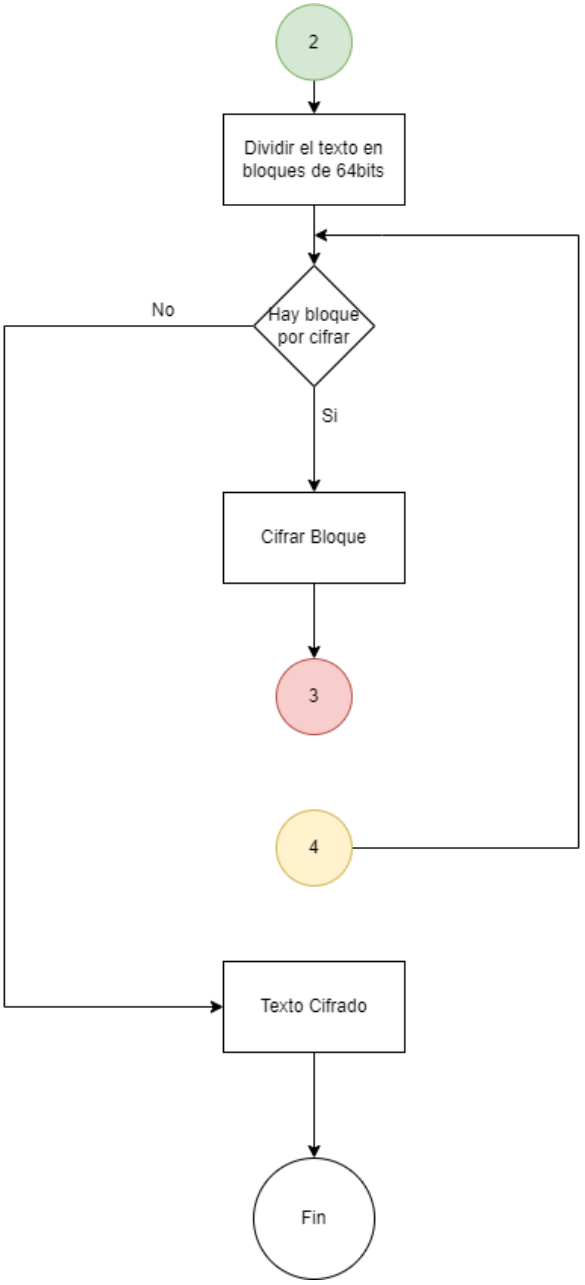
Pseudocódigo de este modelo:

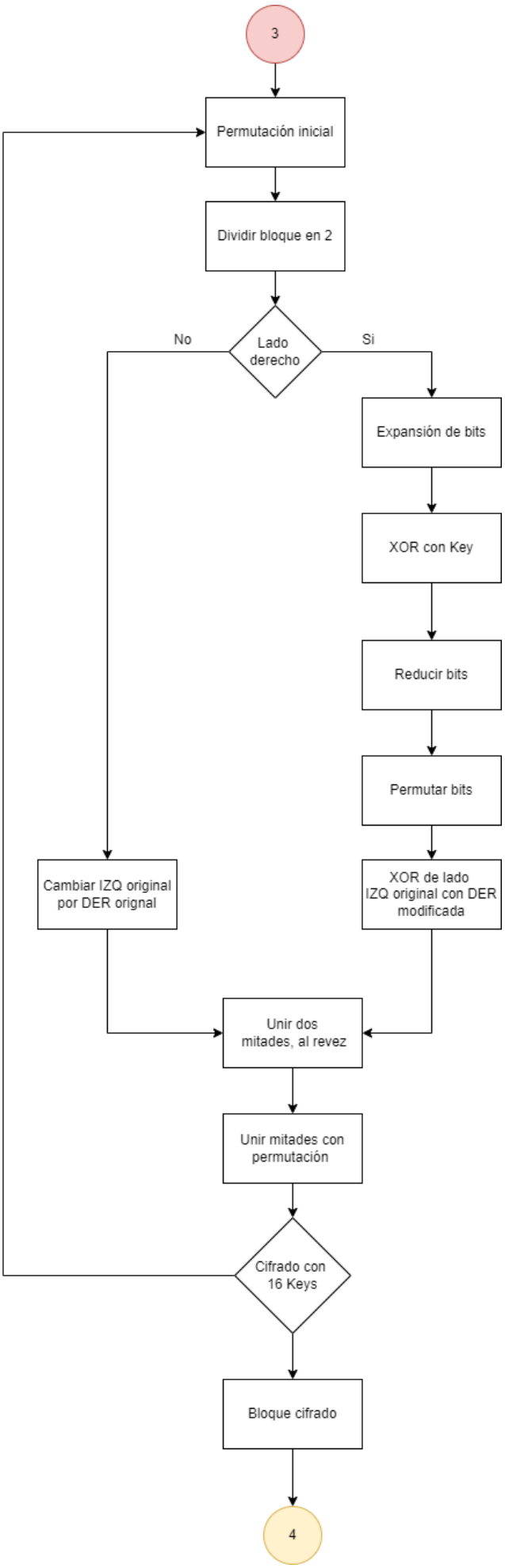
1. En el primer paso, el bloque de texto sin formato de 64 bits se transfiere a una función de Permutación (IP) inicial.
2. La permutación inicial realizada en texto plano.
3. A continuación, la permutación inicial (IP) produce dos mitades del bloque permutado; dice Texto sin formato izquierdo (LPT) y Texto sin formato derecho (RPT).
4. Ahora cada LPT y RPT pasan por 16 rondas de proceso de encriptación.
5. Al final, LPT y RPT se vuelven a unir y se realiza una permutación final (FP) en el bloque combinado
6. El resultado de este proceso produce texto cifrado de 64 bits.

(GeeksforGeeks, 2018).

Diagrama de flujo




2
Dividir el texto en bloques de 64bits
Hay bloque por cifrar
No
Si
Cifrar Bloque
3
4
Texto Cifrado
Fin


3
Permutación inicial
Dividir bloque en 2
Lado derecho
No
Si
Expansión de bits
XOR con Key
Reducir bits
Permutar bits
XOR de lado IZQ original con DER modificada
Cambiar IZQ original por DER orignal
Unir dos mitades, al revez
Unir mitades con permutación
Cifrado con 16 Keys
Bloque cifrado
4

Descripción de rutinas

decrypt()

Función que descifra utilizando la llave, bloques y su tamaño. Este itera en la llave haciendo shift los bits y luego utiliza librerías para setear el parity y descifrar.

tryKey()

Esta función busca si la palabra que se está buscando se encuentra en el texto a través de llamadas a mmcpy, luego decrypt para descifrar la llave y últimamente strstr para determinar si hay una ocurrencia del string.

memcpy()

Esta función recibe como parámetros a dest, src y n. Al ser llamada copia n caracteres del área de memoria src hacia la memoria dest y retorna un puntero al destino.

strstr()

Esta función recibe como parámetros a haystack y needle. Al ser llamada, la función busca la primera ocurrencia del string needle en el string haystack y retorna un puntero a dicha ocurrencia o null si no se encontró.

Uso y Flujo de comunicación de las primitivas de MPI:

MPI_Irecv

MPI_Irecv significa MPI (Receive with Immediate return); no se bloquea hasta que se recibe el mensaje. Para saber si el mensaje ha sido recibido, debe utilizar MPI_Wait o MPI_Test en el MPI_Request lleno.

MPI_Send

MPI_Send es el envío estándar en MPI. Entre bastidores, emitirá un envío con buffer MPI_Bsend o un envío síncrono MPI_Ssend. Esta decisión se basará en si el buffer adjunto para los envíos con buffer contiene suficiente espacio libre para el mensaje a enviar.

MPI_Wait

MPI_Wait espera a que se complete una operación no bloqueante. Es decir, a diferencia de MPI_Test, MPI_Wait se bloqueará hasta que la operación no bloqueante subyacente se complete. Dado que una operación no bloqueante retorna inmediatamente, lo hace antes de que la rutina MPI subyacente se complete. Esperar a que esa rutina se complete es para lo que está diseñado MPI_Wait.

(Rookiehpc, 2022).

Referencias Bibliográficas

edpresso. (2020). What is the DES algorithm. Extraído de: https://www.educative.io/edpresso/what-is-the-des-algorithm
GeeksforGeeks. (2018). Data encryption standard (DES) | Set 1. Extraído de: https://www.geeksforgeeks.org/data-encryption-standard-des-set-1/

Rookiehpc. (2022). MPI documentation. Extraído de: https://www.rookiehpc.com/mpi/docs/index.php

Anexo

Fig. 1: edpresso. (2020). What is the DES algorithm. Extraído de: https://www.educative.io/edpresso/what-is-the-des-algorithm