Universidad del Valle de Guatemala
Sección 10 - Computación Paralela y Distribuida
Catedrático: Juan Jose Celada

Proyecto 2

Cristopher Jose Rodolfo Barrios Solis - 18207
Juan Fernando de Leon Quezada - 17822
Jose Amado Garcia Rosales - 181469

Índice

Introducción

A lo largo de la historia el ser humano se ha necesitado comunicarse, poco a poco se han creado varias manera para poder realizar este proceso inclusive comunicándose a distancia, con anterioridad se han usado las cartas y mensajeros, pero en la actualidad se utilizan las computadoras para que las personas puedan realizar dicha labor, pero esto puede llegar a tener muchos riesgos ya que un intruso puede ver lo que contiene, cuando se envía un mensaje y con respecto a un cliente puede dejar mucho que desear de la empresa.

En pocas palabras un mensaje es un conjunto de información que entre en juego entre un emisor y un receptor, la información compartida puede darse de varias maneras en este caso se utilizan ciertas funciones y subrutinas para que pueda ser efectuado lo que se requiere, todas las funciones que se utilicen durante la realización de envio de mensaje encriptan la información para que haya seguridad y privacidad en lo que se está realizando.

Para explotar la mayor parte de las funciones que se contienen en el programa se puede realizar gracias a Message Passing Interface o MPI que contiene la sintaxis para los mensajes que se van a transmitir, y esto toma en cuenta la existencia de múltiples procesadores. Cuando se transmiten los mensajes se puede decir que es una proceso que se realiza mucho, además que no se utiliza memoria compartida para que sea funcional, lo cual tiene sus ventajas.

Se tiene previsto para el proyecto la utilización de Open MPI con el objetivo de tener una buena experiencia del usuario con respecto al traslado de la información utilizando ciertos algoritmos que permitan tener una mejor seguridad y rapidez cuando se esté interactuando con esto.

Antecedentes

Data Encryption Standard (DES)

El estándar de cifrado de datos más conocido como DES en este caso es un método de cifrado de datos de clave simétrica que en la actualidad se encuentra obsoleto, esta se ha usado con anterioridad pero con la tecnología actual este puede ser vulnerado facilmente, la mayoria de veces solo se utiliza para aprender a cómo funcionan las inscripciones de datos, se adoptó en 1977 para que las agencias gubernamentales protegieran los datos confidenciales y se retiró oficialmente en 2005 (Loshin, 2021).

DES fue el primer algoritmo de encriptación que el gobierno de los EE. UU. aprobó para divulgación pública. Este movimiento aseguró que fuera adoptado rápidamente por industrias, como los servicios financieros, que necesitaban un cifrado fuerte. Debido a su simplicidad, DES también se usó en una variedad de sistemas integrados, incluidos los siguientes (Loshin, 2021).

Algunas características clave que afectan el funcionamiento de DES incluyen las siguientes (Loshin, 2021):

- Cifrado en bloque.
- Varias rondas de cifrado.
- Clave de 64 bits.
- Sustitución y permutación.
- Compatibilidad con versiones anteriores.

OPEN MPI

MPI (Message Passing Interface) se utiliza principalmente en computación paralela distribuida. Es un protocolo de comunicación para computadoras paralelas (Gustavus, 2020).

OpenMPI es una de esas implementaciones. La otra implementación notable de MPI es MPICH (Gustavus, 2020).

Algunos de los casos de uso importantes de MPI son los siguientes (Gustavus, 2020).

- Esto es utilizado por muchas supercomputadoras (Gustavus, 2020).
- Hay un aumento en las técnicas de aprendizaje profundo que se aplican a problemas de NLP, visión por computadora, reconocimiento de voz, etc. y los investigadores han logrado avances notables en lo que respecta al

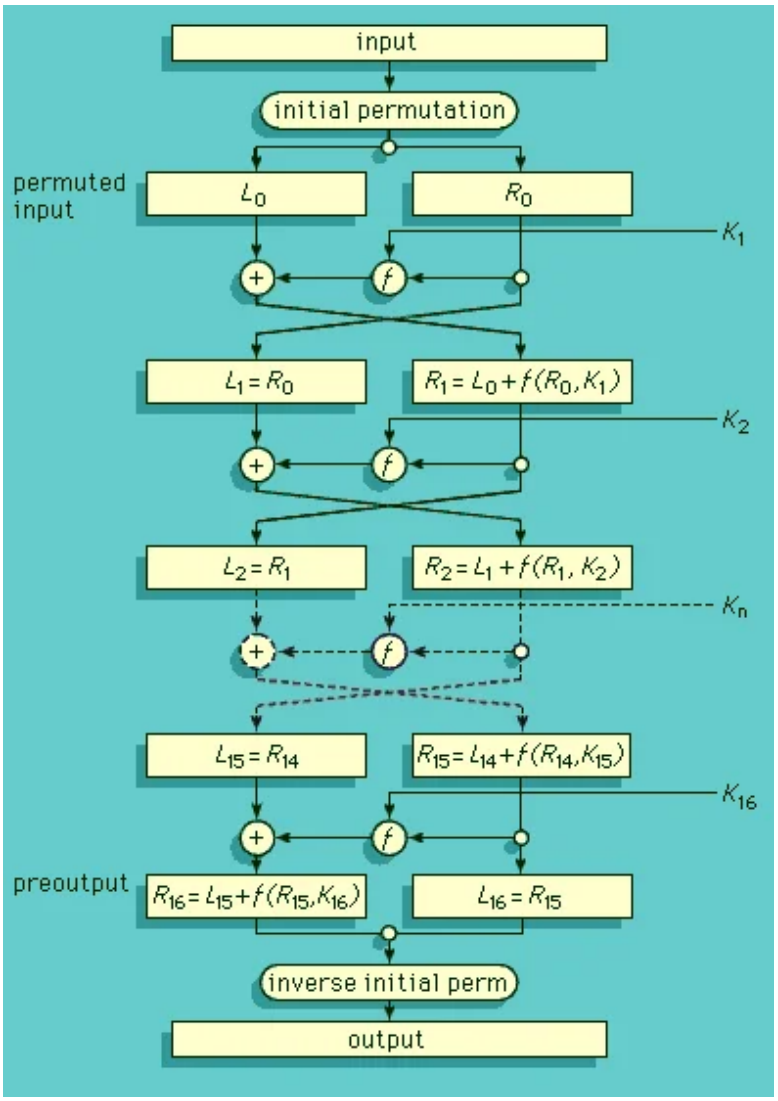
rendimiento y la escalabilidad. Horovod un marco de aprendizaje profundo distribuido; admite TensorFlow, PyTorch junto con el uso de OpenMPI. Con esto, uno puede hacer uso de múltiples GPU en múltiples máquinas para un entrenamiento más rápido de un modelo que entrenarlo usando una sola GPU (Gustavus, 2020).

Diagrama de flujo

Diagrama 1: Cifrado

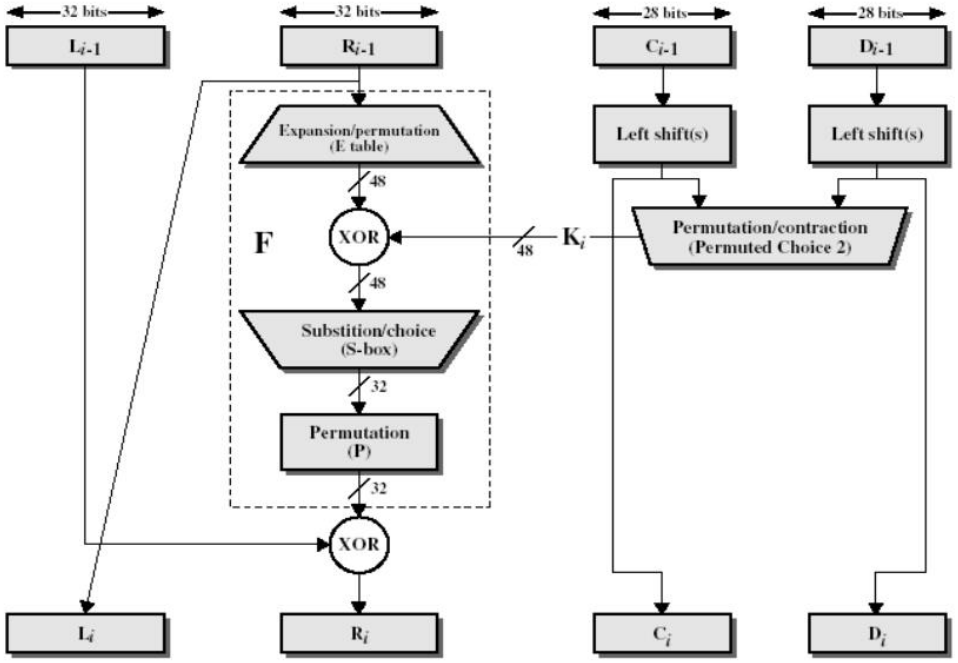
L = izquierda.

R = derecha.



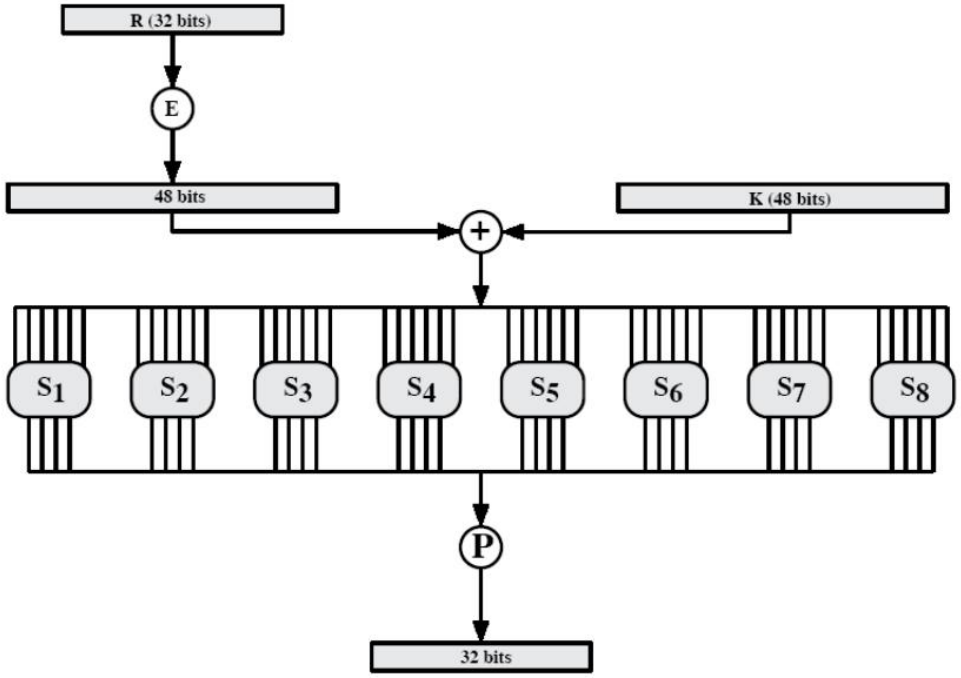
(Owens, 2014).

Diagrama 2: Iteración única del algoritmo DES



(Owens, 2014).

Diagrama 3: Cálculo de F



(Owens, 2014).

Decrypt

Esta rutina que se menciona hace el cifrado que corresponde, establece la paridad de la llave a utilizar, esta recibe la llave (key) la longitud (len) y ciph.

```
void decrypt(long key, char *ciph, int len){
  //set parity of key and do decrypt
  long k = 0;
  for(int i=0; i<8; ++i){
    key <<= 1;
    k += (key & (0xFE << i*8));
  }
  des_setparity((char *)&k);  //el poder del casteo y &
  ecb_crypt((char *)&k, (char *) ciph, 16, DES_DECRYPT);
}
```

TryKey

Este prueba si la llave es válida para que se pueda desencriptar el mensaje, esta recibe como parámetros la llave y la longitud.

```
int tryKey(long key, char *ciph, int len){
  char temp[len+1];
  memcpy(temp, ciph, len);
  temp[len]=0;
  decrypt(key, temp, len);
  return strstr((char *)temp, search) != NULL;
}
```

Memcpy

El propósito de esta rutina es dar como resultado al final una copia binaria de los datos, copia los valores de num bytes desde la ubicación al origen del bloque de memoria que se ve en el destino.

```
  memcpy(temp, ciph, len);
  temp[len]=0;
  decrypt(key, temp, len);
  return strstr((char *)temp, search) != NULL;
}
```

Strstr

Esta rutina como bien se ve no acepta valores nulos, esto nos hará retornar un puntero a la primera posición str2 a str1 o uno nulo si no está str2.

```
  return strstr((char *)temp, search) != NULL;
}
```

Catálogo de Funciones

MPI_Irecv

Bloquea el proceso hasta que se se notifique la llegada de un mensaje, cuando es necesario utilizar un mensaje se utiliza un directiva de MPI para que se detenga la ejecución, en resumen esta función se encarga del recibo del mensaje.

Parámetros

- Count: indica el número máximo de elementos que se recibirán en el buffer
- Datatype: El dato de cada elemento que se recibe
- Source: Rango del proceso de origen, se reciben mensajes que el origen sea específico.
- Tag: El tag queda a disposición del usuario
- Comm: Comunicador para comunicación.

Retorna buf qué es el buffer de entrada y request que tiene la etiqueta de la operación no bloqueante.

MPI_Send

Como su nombre lo indica, esta función se encarga del envío de un mensaje de un proceso inicial a uno final.

Parámetros

- Buf: Dirección del buffer inicial a la hora del envío.
- Count: Número de elementos que se envían.
- Datatype: El dato de cada elemento que se envía.
- Dest: Rango del proceso destino
- Tag: El tag queda a disposición del usuario
- Comm: Comunicador para comunicación.

MPI_Wait

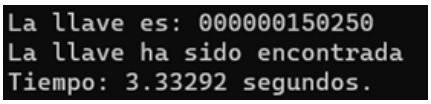
Esta función hace que se espere el proceso hasta que se termine la operación que se le indica, puede ser de envío o de recibo.

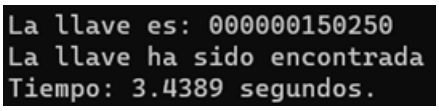
- Request: Se guarda una etiqueta que identifica la operación no bloqueante
- Status: Se tiene datos relevantes sobre el mensaje, la etiqueta y el tamaño

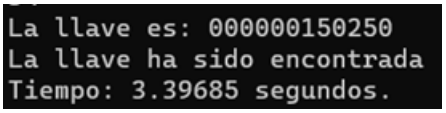
En caso de error se puede tener estas posibles opciones MPI_ERR_REQUEST, MPI_SUCCESS, MPI_ERR_ARG.

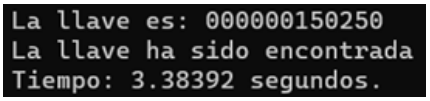
Pruebas

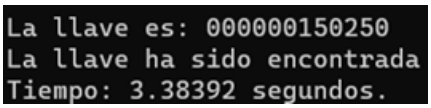
Programa Secuencial



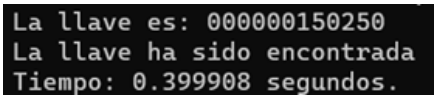


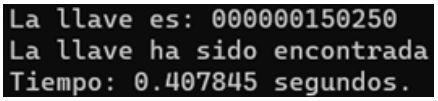


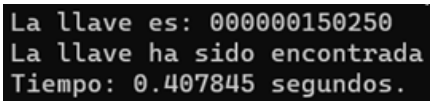


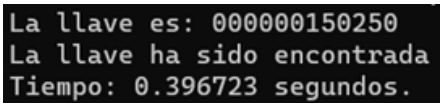


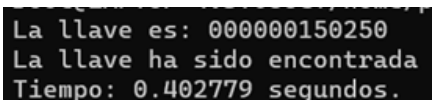
Programa paralelo

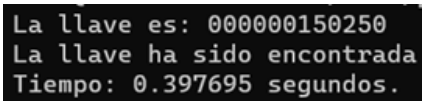












Texto: si no puedes explicar algo de forma sencilla es que ni tú mismo lo has entendido lo suficiente.

Cuadro 1:

Intento	Tiempo Secuencial (s)	Tiempo Paralelo (s)
1	3.24698	3.47958
2	3.34859	3.18575
3	5.31487	7.13257
4	4.35781	5.21465
5	5.74168	8.74682
6	3.24568	2.47365
Promedio	4.20926833333	5.03883666667

Speedup: 0.835365107421067848923183282484 57

Cuadro 2: Llave corta

Intento	Tiempo Secuencial (s)	Tiempo Paralelo (s)
1	3.33292	0.399908
2	3.4389	0.407845
3	3.39685	0.407845
4	3.38392	0.396723
5	3.38392	0.402779
6	3.34543	0.397695
Promedio	3.38032333333	0.4021325

Speedup: 8.40599387846045768496701957688 08

Cuadro 3: Llave larga

Intento	Tiempo Secuencial (s)	Tiempo Paralelo (s)
1	45.47583	77.14975
2	46.21873	78.47146
3	47.47135	79.14274
4	45.17384	82.84317
5	44.35714	76.47814
6	47.13784	77.35874
Promedio	45.972455	78.574

Speedup: 0.585084824496652836816249650011145

Conclusiones

- El tiempo en que tarda en ser ejecutado el proceso puede verse afectado de manera significativa dependiendo del tipo de datos que se han colocado en cada llave.
- Las dos propiedades deseadas, el efecto de avalancha y la integridad, de un cifrado son satisfechas por el algoritmo Estándar de cifrado de datos. Estas dos propiedades hacen que el cifrado sea muy fuerte.
- Un pequeño cambio en el texto sin formato da como resultado un gran cambio en el texto cifrado.cada bit de texto cifrado depende de muchos bits de texto sin formato.
- El tiempo de cifrado y asimismo el de descifrado cambian con respecto al número de núcleos que se le indica en el código.

Recomendaciones

- Realizar el algoritmo de una manera simple ya que en un futuro cuando se quiera volver a utilizar se puede comprender de una manera más fácil e incluso cuando se hagan mejoras se hacen mejoras óptimas..
- Cuando se quiere medir el tiempo de ejecución del algoritmo se debe de realizar varias mediciones ya que solo una no es de mayor relevancia y no se obtiene un resultado concreto.

Bibliografía

- Loshin, P. (2021). Data Encryption Standard (DES). Extraído de: https://www.techtarget.com/searchsecurity/definition/Data-Encryption-Standard
- Gustavus, J. (2020). Data Encryption Standard. Extraido de: https://www.britannica.com/topic/Data-Encryption-Standard
- Owens, K. (2014). Data Encryption Standard (DES). Extraido de: https://www.slideserve.com/kareem-owens/data-encryption-standard-des