UNIVERSIDAD DEL ISTMO

FACULTAD DE INGENIERÍA

INFORME ALGORITMO DES

SISTEMAS OPERATIVOS

KEVIN ALEXANDER CHAMPNEY LEÓN

ELISA MARGARITA MONZÓN GODOY

Guatemala, 24 de septiembre de 2019

ÍNDICE

INTRODUCCIÓN

La confidencialidad de la información es esencial para las personas y organizaciones, y garantizarla, en esta era digital, se ha convertido en un gran desafío, sin embargo, mediante el uso de la criptografía, es posible mantener la confidencialidad de la información.

La computación paralela es una de las principales herramientas para mejorar el rendimiento de aplicaciones que no son intrínsecamente secuenciales, por lo tanto, puede mejorar la eficiencia de los algoritmos criptográficos paralelizables.

Para este proyecto de encriptación y desencriptación, se estará utilizando las siguientes tecnologías:

- Lenguaje C+ +, fue diseñado en 1979, con la intención de extender al lenguaje de programación C mecanismos que permiten la manipulación de objetos.
- Hilos: son tareas que pueden ser ejecutadas al mismo tiempo que otra.
- Variables mutex y condicionales: utilizadas para bloquear el procesamiento de hilos, y poner en espera su ejecución
- Algoritmo DES: trabaja en bits o números binarios, similar a las computadores digitales.

CONTENIDO

Descripción de Algoritmo DES

DES, Data Encryption Estándar[1], es un esquema de encriptación simétrico, desarrollado en 1977 por el Departamento de Comercio y la Oficina Nacional de Estándares de Estados Unidos, en colaboración con la empresa IBM. Fue creado con el objeto de proporcionar al público en general un algoritmo de cifrado normalizado para las redes de ordenadores, que contara con los siguientes requisitos:

- Ofrecer un alto nivel de seguridad relacionado con una pequeña clave utilizada para cifrado y descifrado
- Comprensible
- No depender de la confidencialidad del algoritmo
- Adaptativo y económico
- Eficaz y exportable

El algoritmo se basa en un sistema mono alfabético, con un algoritmo de cifrado consistente en la aplicación sucesiva de varias permutaciones y sustituciones. Inicialmente el texto en claro a cifrar se somete a una permutación, con un bloque de entrada de 64 bits, para posteriormente ser sometido a la acción de dos funciones principales, una función de permutación con entrada de 8 bits y otra de sustitución con entrada de 5 bits, en un proceso que consta de 16 etapas de cifrado.

Especificaciones necesarias del algoritmo

Permutación inicial

La permutación inicial toma como parámetro un texto de 64 bits. La tabla contiene números del 1 al 64 en un orden aleatorio. Después el texto escrito es organizado en el orden que fue especificado en la matriz de la permutación inicial.

Luego el texto pasa por 16 rondas, cada una procesada con una llave de 48 bits. En cada ronda el texto de 64 bits es dividido en dos partes, cada una de 32 bits, y la parte derecha pasa por la permutación de expansión.

[1] En castellano: Algoritmo de Cifrado de Datos

Permutación de expansión

La parte derecha del texto dividido es dado a la permutación de expansión. El propósito de esta permutación es expandir el texto de 32 bits a un texto de 48 bits, agregándole 16 bits al texto.

Principio de funcionamiento

Se trata de un sistema de cifrado simétrico por bloques de 64 bits, de los que 8 bits se utilizan como control de paridad (para la verificación de la integridad de la clave). Cada uno de los bits de la clave de paridad se utiliza para controlar uno de los bytes de la clave por paridad impar, es decir, que cada uno de los bits de paridad se ajusta para que tenga un número impar dentro del byte al que pertenece. Por lo tanto, la clave tiene una longitud útil de 56 bites.

El algoritmo se encarga de realizar combinaciones, sustituciones y permutaciones entre el texto a cifrar y la clave, asegurándose al mismo tiempo de que las operaciones puedan realizarse en ambas direcciones (para el descifrado). La combinación entre sustituciones y permutaciones se llama cifrado del producto.

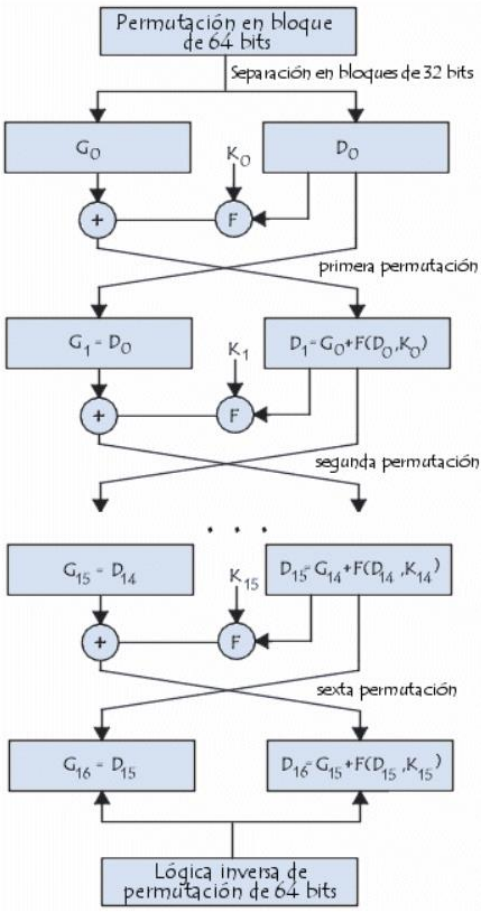
La clave es codificada en 64 bits y se compone de 16 bloques de 4 bits, generalmente anotadas de k_1 a k_{16}. Dado que "solamente" 56 bits sirven para el cifrado, puede haber hasta 2^{56} (o $7.2*10^{16}$) claves diferentes

Diagrama de flujo

Las partes principales del algoritmo constan de:

- Fraccionamiento del texto en bloques de 64 bits
- Permutación inicial de los bloques
- Partición de los bloques en dos partes:
    - Izquierda → I
    - Derecha → D
- Fases de permutación y de sustitución repetidas 16 veces (rondas)
- Reconexión de las partes izquierda y derecha, seguida de la permutación inicial inversa.

Imagen No.1: Diagrama algoritmo DES



Fuente: CCM, 2019.

Algoritmo descriptivo

1. Se incluyes las librerías.
2. Se declaran los hilos y las variables globales, mutex y condicionales.
3. En el main:
    a. El programa solicita la cantidad de caracteres que el usuario desea escribir. Si el texto es menor al especificado el programa termina.
    b. Se castea el texto escrito a un string para evitar problemas al hacer el llamado de la subrutina.
    c. Se crea un hilo para llamar a la subrutina.
    d. El programa espera a que el hilo termine de ejecutarse.
    e. Se inicializan las variables mutex y condicionales.

4. Dentro de la subrutina:
    a. Se hace un casting de puntero a string.
    b. Se declaran variables tipo Des.
    c. Se hace el llamado de la función Encrypt y se almacena el resultado en una variable tipo char*.
5. Se hace el llamado de la función Decrypt, al finalizar Encrypt, enviándole como parámetro el resultado devuelto por Encrypt.
6. Imprime los resultados devueltos por las funciones principales: Encrypt y Decrypt.
    a.
7. Dentro de Encrypt:
    a. Convierte cada carácter del texto escrito a binario.
    b. Se realiza la primera permutación.
    c. Divide el resultado de la permutación en dos partes.
    d. Inicia las 16 rondas para el proceso de encripción.
    e. Retorna el texto encriptado.
8. Dentro de Decrypt:
    a. Convierte cada carácter del texto encriptado a binario.
    b. Se realiza la primera permutación.
    c. Divide el resultado de la primera permutación en dos partes.
    d. Inicia las 16 rondas para realizar el proceso inverso hecho en Encrypt.
    e. Retorna el texto desencriptado.

Rondas de Feistel

El cifrado de Feistel es un método de cifrado en bloque, con una estructura particular, y es implementado por el algoritmo Data Ecnryption Estándar, ya que presenta la ventaja de ser reversibles por lo que las operaciones de cifrado y descifrado son idénticas, requiriendo únicamente invertir el orden de las subclaves utilizadas.

Algoritmo

Este algoritmo se denomina simétrico por rondas, es decir, realiza siempre las mismas operaciones un número determinado de veces (denominadas rondas). Los pasos de la red de Feistel son entre algunos más:

1. Se selecciona una cadena, N, normalmente de 64 o 128 bits, y se la divide en dos subcadenas, L y R, de igual longitud (N/2)
2. Se toma una función, F, y una clave Ki
3. Se realizan una serie de operaciones complejas con F y Ki y con L o R (solo una de ellas)
4. La cadena obtenida se cambia por la cadena con la que no se han realizado operaciones, y se siguen haciendo las rondas.

Variables mutex y condicionales

Mutex

Las variables mutex son un mecanismo de sincronización entre hilos, que ayudan a proteger una sección critica del código. Estas variables son de tipo pthread_mutex_t y los métodos para emplear los mutex son los siguientes:

- int pthread_mutex_init(pthread_mutex_t *mutex, pthread_mutexattr_t * attr);
    - Inicializa el mutex
- int pthread_mutex_destroy(pthread_mutex_t *mutex) ;
    - Destruye el mutex (lo elimina de la memoria).
- int pthread_mutex_lock(pthread_mutex_t *mutex);
    - Bloquea el mutex si no lo tiene nadie. Si alguien tiene bloqueado el mutex el proceso espera hasta que el proceso que lo tiene bloqueado lo libera.
- int pthread_mutex_unlock(pthread_mutex_t *mutex);
    - Libera el mutex.

Condicionales

Las variables condicionales son variables de sincronización asociadas a un mutex que ayudan a sincronizar nuestros bloques entre hilos. Por ejemplo, realizamos un bloqueo, pero tenemos que esperar a que otro recurso realice una acción, con estas variables podemos esperar dicha

acción. Estas variables pueden realizar principalmente dos acciones: *wait* y *signal* y conviene realizarlas entre *lock* y *unlock* de un mutex. Estas variables son de tipo pthread_cond_t y los métodos para emplear los mutex son los siguientes:

- int pthread_cond_init(pthread_cond_t*cond, pthread_condattr_t*attr);
    - Inicializa la variable condicional.
- int pthread cond destroy(pthread_cond_t *cond);
    - Destruye la variable condicional.
- int pthread_cond_wait(pthread_cond_t*cond, pthread_mutex_t*mutex);
    - Mediante este método suspendemos el hilo en el que se ejecuta y liberamos el mutex. El proceso queda suspendido hasta que recibe una señal de activación. Una vez que se activa se vuelve a luchar por el control del mutex.
- int pthread_cond_signal(pthread_cond_t *cond);
    - Se reanuda la ejecución de uno o más hilos que estén esperando por la variable condicional (cond).
    - En caso de no haber ningún proceso esperando no ocurre nada.
- int pthread_cond_broadcast(pthread_cond_t *cond);
    - Se reanudan todos los hilos que estaban suspendidos por la variable condicional (cond).

Catálogo de funciones principales de programa

Las principales funciones que utilizaremos para el programa de encripción y desencripción son las siguientes:

- Declaración de variables
    - pthread_mutex_t mutex
    - pthread_cond_t cond
    - pthread_t thread
- Inicialización de funciones
    - pthread_mutex_init(&mutex, NULL)
    - pthread_cond _init(&cond, NULL)
    - pthread_create(&thread, NULL, subrutina, NULL)

- Subrutina para encriptar y desencriptar
- Función de encriptar → incluyendo rondas y creación de llaves
- Función de desencriptar → incluyendo rondas y lectura de llaves

CONCLUSIONES

En conclusión, el proyecto constará de dos partes que resolverán encriptación y desencriptación de mensajes con una longitud 70 caracteres, implementando la idea del algoritmo DES que se basa en encriptar mensajes de 64 bits, que es similar a un numero 16 hexadecimal, asimismo se utilizan llaves que también son aparentemente hexadecimales.

Para finalizar, en la implementación del algoritmo DES se utilizarán variables mutex y condicionales, de manera que se puede llevar a cabo paralelismo de procesamiento y hacer uso de hilos.

BIBLIOGRAFÍA

- http://neo.lcc.uma.es/evirtual/cdd/tutorial/presentacion/des.html Recuperado 21 de septiembre de 2019
- https://es.ccm.net/contents/130-introduccion-al-cifrado-mediante-des Recuperado 21 de septiembre de 2019
- https://www.geeksforgeeks.org/data-encryption-standard-des-set-1/ Recuperado 21 de septiembre de 2019
- http://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm Recuperado 21 de septiembre de 2019