UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO VALLE DE CHALCO

PROPUESTA DE INCLUSIÓN DEL CRIPTOSISTEMA TRIPLE DES-96 EN EL SSL/TLS RECORD PROTOCOL

T E S I S

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A

Ing. en C. PEDRO DAVID FILIO AGUILAR

ASESOR:

Dr. SAMUEL OLMOS PEÑA

VALLE DE CHALCO SOLIDARIDAD, MÉXICO. OCTUBRE 2014

Valle de Chalco Solidaridad, Edo. de México a 22 de octubre de 2014.

Dr. Samuel Olmos Peña
Coordinador de la Maestría en Ciencias de la Computación
Centro Universitario UAEM Valle de Chalco
Presente

Por este medio le comunicamos a usted que la Comisión Revisora designada para analizar la tesis denominada "**Propuesta de Inclusión del Criptosistema Triple Des-96 en el SSL/TLS Record Protocol**", que como parte de los requisitos para obtener el grado académico de Maestría en Ciencias de la Computación presenta el **Ing. Pedro David Filio Aguilar** con número de cuenta **0722178** para sustentar el acto de Recepción Profesional, ha dictaminado que dicho trabajo reúne las características de contenido y calidad necesarios para proceder a la impresión del mismo.

Atentamente

Tutora Adjunta	**Tutor Académico**	**Tutor Adjunto**
Dra. Cristina Juárez Landín	**Dr. Samuel Olmos Peña**	**Dr. René Guadalupe Cruz Flores**

Oficio Coord MACSCO 033/2014
Valle de Chalco Solidaridad, Edo. de México a 27 de octubre del 2014.

Ing. Pedro David Filio Aguilar
Candidato al Grado de Maestría en Ciencias de la Computación
Centro Universitario UAEM Valle de Chalco
Presente

De acuerdo con el Reglamento de Estudios Avanzados de la Universidad Autónoma del Estado de México y habiendo cumplido con todas la indicaciones que la Comisión Revisora realizó con respecto a su trabajo de tesis titulado "***Propuesta de inclusión del criptosistema Triple DES-96 en el SSL/TLS Record Protocol***", la Coordinación de la Maestría en Ciencias de la Computación del Centro Universitario UAEM Valle de Chalco, concede la autorización para que proceda a la impresión de la misma.

Sin más por el momento, le reitero la seguridad de mi especial consideración y estima.

ATENTAMENTE
"PATRIA, CIENCIA Y TRABAJO"
"2014, 70 Aniversario de la Autonomía ICLA-UAEM"

Centro Universitario UAEM

Dr. Samuel Olmos Peña.
Coordinador de la Maestría en Ciencias de la Computación.
C.U UAEM Valle de Chalco.
Universidad Autónoma del Estado de México

c.c.p. Archivo.
SOP

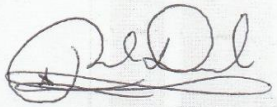
CARTA DE CESIÓN DE DERECHOS DE AUTOR

El que suscribe **Pedro David Filio Aguilar** Autor del trabajo escrito de evaluación profesional en la opción de **Tesis** con el título **Propuesta de Inclusión del Criptosistema Triple Des-96 en el SSL/TLS Record Protocol**, por medio de la presente con fundamento en lo dispuesto en los artículos 5, 18, 24, 25, 27, 30, 32 y 148 de la Ley Federal de Derechos de Autor, así como los artículos 35 y 36 fracción II de la Ley de la Universidad Autónoma del Estado de México; manifiesto mi autoría y originalidad de la obra mencionada que se presentó en **CU UAEM, Valle de Chalco Solidaridad, Edo. De México** para ser evaluada con el fin de obtener el Título Profesional de **Maestro en Ciencias de la Computación**.

Así mismo expreso mi conformidad de ceder los derechos de reproducción, difusión y circulación de esta obra, en forma NO EXCLUSIVA, a la Universidad Autónoma del Estado de México; se podrá realizar a nivel nacional e internacional, de manera parcial o total a través de cualquier medio de información que sea susceptible para ello, en una o varias ocasiones, así como en cualquier soporte documental, todo ello siempre y cuando sus fines sean académicos, humanísticos, tecnológicos, históricos, artísticos, sociales, científicos u otra manifestación de la cultura.

Entendiendo que dicha cesión no genera obligación alguna para la Universidad Autónoma del Estado de México y que podrá o no ejercer los derechos cedidos.
Por lo que el autor da su consentimiento para la publicación de su trabajo escrito de evaluación profesional.

Se firma la presente en la ciudad de **Estado de México**, a los **22** días del mes de **octubre** de 20**14**.



Ing. Pedro David Filio Aguilar
Nombre y firma de conformidad

Agradecimientos

A Dios:
Por permitirme despertar cada mañana y llenar mi corazón de esperanza para enfrentar los problemas que he enfrentado.

A mis Profesores:
Por los conocimientos que me trasmitieron, por su tiempo y su paciencia.

A mi Asesor el Dr. Samuel Olmos Peña:
Porque gracias a la libertad que me ofreció logré desarrollar y concluir un tema que me apasiona y gracias a su guía y consejos hoy puedo decir que tengo las bases para ser un buen investigador.

A mis Revisores:
La Dra. Cristina y Dr. René por las contribuciones y observaciones para mejorar este trabajo y por invertir su valioso tiempo en ello.

A mis Compañeros de Generación:
Con quienes compartí tiempo, espacio y viví muchas aventuras, en especial; a Lucy, Ary, Bucio, Héctor, Esteban y Ernesto, gracias por formar parte de mi historia.

A CU UAEM Valle de Chalco:
Mi alma mater lugar donde he vivido infinidad de experiencias y he aprendido muchas lecciones, gracias por hacer de mí un profesionista.

Al CONACYT
Por el apoyo económico que me otorgaron a través de la Beca Nacional y la Beca Mixta para Becarios CONACYT de Movilidad Nacional.

Al CIDETEC
Espacio donde me aceptaron como alumno de movilidad académica por un año y donde aprendí tanto, en especial al Dr. Víctor Manuel Silva García y al Dr. Rolando Flores Carapia, por su enseñanza y valioso tiempo.

A todos ellos mi más sincero agradecimiento.

Dedicatoria

A mis padres Alicia Aguilar y Venancio Filio Quienes con su ejemplo, sacrificio, educación y amor han hecho de mí una persona exitosa y a quienes jamás podré pagar como se merecen tanto cariño que he recibido.

A mis hermanos Manuel y Gaby, sé que siempre contare con ustedes, los quiero mucho.

A Don Félix y Doña Reyna por todo el apoyo y cariño para con mi familia.

A mi esposa Lorena por tu comprensión, apoyo incondicional, por acompañarme siempre y por estar con migo en las buenas y en las malas, te amo.

A Esme y Marlen quienes con su cariño e inocencia alegran mi vida, y a Sofía que hizo que saliera corriendo de mis clases para correr a presenciar su nacimiento y quien siempre estuvo redactando junto a mí. Las amo, ustedes son el motor que empuja mi vida, recuerden que detrás de cada logro, hay otro desafío.

RESUMEN

Debido a la seguridad requerida al momento de intercambiar información delicada a través de internet, es necesario el uso de algoritmos y protocolos criptográficos complejos como es el caso de Secure Sockets Layer y Transport Layer Security (SSL/TLS), el funcionamiento de éste está dividido en cuatro sub protocolos; Handshake Protocol, Change Cipher Spec Protocol, Alert Protocol y Record Protocol. El presente trabajo de investigación se centra en el SSL/TLS Record Protocol, donde se construye la inclusión del algoritmo Triple DES-96 en la suite de cifrado de SSL/TLS. Con este objetivo, se desarrollan en lenguaje de programación Java los algoritmos Triple DES y Triple DES-96, se realizan pruebas de cifrado y descifrado sobre 1000 archivos de diferentes tipos y tamaños, se mide el tiempo de cada algoritmo en cifrar y descifrar la información, se registran todos los resultados experimentales obtenidos y por último se comparan los resultados. Al descubrir que los resultados son favorables, en disminución de tiempo y mayor robustez en el cifrado, se plantea la inclusión del algoritmo Triple DES-96 en la suite de cifrado SSL/TLS Record Protocol.

ABSTRACT

Due to the required safety when exchanging sensitive information over the Internet, complex algorithms and cryptographic protocols such as Secure Sockets Layer and Transport Layer Security (SSL/TLS) are required. This is divided into four sub protocols; Handshake Protocol, Change Cipher Spec Protocol, Alert Protocol and Record Protocol. This research focuses on the SSL/TLS Record Protocol, where is constructed an inclusion of the algorithm Triple DES-96 in the encryption suite of SSL/TLS. For this purpose are developed in the Java programming language the algorithms Triple DES and Triple DES-96, tests encryption and decryption are performed over 1000 files of different types and sizes, the time it takes each algorithm to encrypt and decrypt the information is measured, all experimental results are recorded and finally the results are compared. Discovering that the results are favorable in time and reliability is suggested the inclusion of the Triple DES algorithm in the cipher suite of SSL/TLS Record Protocol.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICAS

ÍNDICE DE TABLAS

CAPÍTULO I: INTRODUCCIÓN

El presente trabajo de investigación se divide en cinco capítulos, el desarrollo de cada uno se apoya de figuras, cuadros, tablas y gráficas, con la finalidad de facilitar su lectura y comprensión, el contenido de estos se describe brevemente a continuación:

Capítulo I, Introducción. Es un breve preámbulo al trabajo y la metodología de investigación que sustenta el trabajo, además, enfatiza la importancia del uso de los algoritmos criptográficos para salvaguardar la integridad de la información.

Capítulo II, Marco Teórico. Se puntualizan cada una de las teorías con las cuales el problema de investigación adquiere un sentido, algunas de estas son: clasificación de los algoritmos criptográficos, los protocolos Secure Sockets Layer y Transport Layer Security, los algoritmos de cifrado simétrico Data Encryption Standard (DES) y Triple DES, y finalmente Numero trascendente e.

Capítulo III, Metodología Extreme Programming (XP). Se habla de la metodología para el desarrollo de software empleada y se describe detalladamente cada uno de los métodos y procedimientos que se realizan en este trabajo buscando comprobar de manera teórica la hipótesis.

Capítulo IV, Desarrollo de los algoritmos. Se presenta el desarrollo en lenguaje de programación Java del algoritmo de cifrado Data Encryption Standard (DES) para desarrollar Triple DES, y a este último se le realizan modificaciones para obtener Triple DES-96. Todos se desarrollan basados en la metodología de desarrollo ágil de software Extreme Programming (XP).

En el capítulo V, Resultados Experimentales. Se presentan los resultados, tanto para Triple DES como para Triple DES-96, estos resultados fueron obtenidos

con ayuda de un programa creado en lenguaje Java que se desarrolló en paralelo a esta tesis, inmediatamente se realiza un análisis de estos resultados.

Finalmente el trabajo de investigación termina con las conclusiones que determinan el cumplimiento de lo planteado inicialmente, un apartado de recomendaciones y sugerencias para trabajos futuros, se incluye un glosario de términos para una mejor comprensión, un listado con las referencias consultadas y por ultimo siete anexos.

Planteamiento del Problema

Inicialmente el uso de una red de computadoras era para el intercambio de correo electrónico o para compartir recursos como impresoras. En la actualidad cada vez es más el número de personas que hacen uso de estas conexiones para intercambiar información personal como fotografías y videos o para realizar compras en línea, declaración de impuestos y transacciones bancarias (Davila, 2000).

Junto a estas grandes bondades que ofrecen las redes existen riesgos considerables. Las transacciones bancarias pueden ser falsificadas, la identidad de un usuario consigue ser suplantada, los servidores podrían ser sustituidos, información privada se puede revelar, los cortafuegos pueden ser infringidos y las redes de computadoras saboteadas (Jaworski, 2001).

A medida que se intercambia cada vez más información, es más importante la necesidad de protegerla. Es por ello que han surgido una gran variedad de algoritmos criptográficos complejos, que son un conjunto de técnicas que protegen información en particular, mediante el uso de llaves y que son analizados constantemente por matemáticos y criptógrafos.

La investigación y constante estudio de estos algoritmos criptográficos da como resultado la aparición de protocolos como el Secure Sockets Layer (SSL), Freier et al. (2011), y su sucesor Transport Layer Security (TLS) (Dierks & Rescorla, 2008), que hacen uso de certificados digitales y conjuntos de métodos criptográficos modernos para establecer privacidad y comunicaciones seguras a través de Internet, permitiendo confiar información personal (Ramírez, 2011).

Estos protocolos permiten a las aplicaciones cliente-servidor comunicarse de una forma diseñada para prevenir la falsificación de la identidad del remitente y mantener la integridad del mensaje, ocultando los datos a través de métodos criptográficos mientras se navega por sitios seguros y sus usos se dan en el comercio electrónico, banca online, seguridad en redes, autenticación y cifrado de datos (Muñoz, 2011).

De acuerdo con Maiorano (2009) la suite de algoritmos criptográficos previstos en SSL/TLS son; RC2, RC4, IDEA, AES, Fortezza, Diffie-Hellman, DES, Triple DES, RSA, MD5 y SHA-1. Este trabajo analiza para después comparar el algoritmo criptográfico Triple DES y una variante de él llamada Triple DES-96 (Silva, et al., 2013), y finalmente desarrolla una algoritmo de inclusión dentro de la suite de cifrado del protocolo SSL/TLS Record Protocol.

Objetivos

A continuación se presentan el objetivo general y los objetivos particulares de este trabajo.

Objetivo General

Desarrollar e implementar el algoritmo de criptografía simétrica Triple DES-96, así como, proponer un modelo para su inclusión en la suite de cifrado del SSL/TLS Record Protocol.

Objetivos particulares

- Realizar un análisis de los algoritmos de criptografía simétrica Triple DES y Triple DES-96 y desarrollarlos en lenguaje de programación Java.
- Utilizar el número trascendente e para generar una permutación diferente en el Triple DES-96.
- Combinar permutaciones, buscando acelerar el proceso de cifrado y descifrado de Triple DES-96 sin perder complejidad computacional.
- Realizar pruebas de cifrado y descifrado sobre 1000 archivos en un mismo equipo de cómputo y con las mismas características para cada uno de los dos algoritmos.
- Demostrar mediante los resultados de las pruebas aplicadas que Triple DES-96 es más eficiente en tiempo y robustez de cifrado que Triple DES.
- Proponer la inclusión del algoritmo con Triple DES-96 en la suite de cifrado del SSL/TLS Record Protocol.

Delimitación de la investigación

El SSL/TLS Record Protocol especifica la forma de encapsular los datos transmitidos y recibidos. Para su funcionamiento utiliza los siguientes algoritmos criptográficos: Para intercambio de clave; RSA, Diffie-Hellman, DSA o Fortezza, para las funciones de resumen o hash; MD5 o alguno de la familia SHA, y para el cifrado simétrico AES, IDEA, DES, Triple DES y Fortezza (Stallings, 2010).

Este trabajo se limita al análisis del algoritmo de cifrado simétrico Triple DES así como su implementación utilizando lenguaje de programación Java. Los resultados que se obtengan de examinar mil archivos de diferentes tipos y tamaños serán comparados con los obtenidos por el algoritmo propuesto Triple DES-96.

Productos de la investigación

- Desarrollo de un software multiplataforma que cifre archivos basado en el algoritmo de cifrado Triple DES.
- Desarrollo de un software multiplataforma que cifre archivos basado en el algoritmo de cifrado Triple DES-96.
- Resultados experimentales del cifrado de 1000 archivos de diferentes tipos y tamaños, además de la respectiva comparación de funcionamiento.
- Propuesta de inclusión de Triple DES-96 en el SSL/TLS Record Protocol.

Hipótesis

El trabajo de investigación demuestra que el algoritmo Triple DES-96 es más eficiente que Triple DES, un estándar aceptado internacionalmente y que forma parte de la suite de cifrado de SSL/TLS. Por tanto la hipótesis es:

Si se conocen y entienden los elementos que conforman la suite de cifrado SSL/TLS, entonces, se puede sustituir alguno de estos elementos por otro de mayor confiabilidad, mediante algoritmos de programación que eficiente el cifrado de cualquier archivo.

Justificación

El aporte de este trabajo es significativo ya que adicional a lo publicado por el Dr. Silva, et al (2013) este trabajo propone:

1. Ocupar los primeros 96 valores después del número decimal del número trascendente e.
2. Utilizar el resultado del punto 1 para construir la permutación variable PV.
3. Mover la tabla de expansión E a la salida de las $S\,\mathrm{Boxes}$ para que sea posible combinarla con la permutación P y se realicen en una sola instrucción.

Los aportes de esta investigación es la divulgación de las modificaciones realizadas al algoritmo Triple DES-96 así como su comprobación y funcionamiento en busca de optimizar e innovar los procesos, esto tiene como consecuencia el incremento en la velocidad de cifrado y la obtención de un resultado completamente diferente al que se obtiene con Triple DES. Ambas, sin tener pérdida en poder computacional.

Los beneficiados son todos aquellos interesados en investigar, comprobar y conocer estas modificaciones, ya que durante su desarrollo se tocan temas de interés como son; programación orientada a objetos, optimización de procesos, innovación, investigación y análisis de seguridad.

Metodología de investigación

Si bien es cierto que existen diferentes tipos de investigación científica dependiendo del método y los fines que se persiguen, este trabajo de tesis, para su desarrollo, se sustenta en cuatro tipos de investigación científica los cuales se mencionan a continuación:

1. **Investigación Exploratoria.** Con la cual se busca obtener conocimiento sobre la tarea a realizar, comprender los conceptos principales y encontrar otros estudios e investigaciones referentes al tema de estudio y se basa en consultas de tipo bibliográfico tales como libros, trabajos de tesis, artículos científicos, memorias, revistas, bases de datos especializados, páginas web, videos y otros recursos especializados.

2. **Investigación Descriptiva.** Que para este trabajo y con base en la hipótesis consiste en la descripción de los conceptos fundamentales sobre criptografía, la descripción detallada de los algoritmos y protocolos de cifrado involucrados y las relaciones que existen entre ellos así como la descripción de los programas de cómputo que se generan a partir de esta información.

3. **Investigación Experimental.** En este trabajo se utiliza este tipo de investigación para realizar las pruebas y análisis de los resultados además sirve para determinar los alcances y las características del modelo que se propone. Se basa en la aplicación recurrente de dos programas de cómputo a 1000 archivos de diferentes tipos y la comparación del tiempo de ejecución de cada programa.

4. **Investigación Explicativa.** Derivado de la ejecución de cada uno de los programas desarrollados, sobre determinados tipos de archivos con características diferentes. Por medio relaciones causa-efecto se explica el comportamiento de los resultados obtenidos.

CAPÍTULO II: MARCO TEÓRICO

Según Stallings (2004) de la necesidad por proteger la información y los sistemas que la administran surgen los siguientes términos:

Seguridad de la Información: Se puede hablar de la Seguridad de la Información como el conjunto de reglas, planes y acciones que permiten asegurar la información manteniendo las propiedades de confidencialidad, integridad y disponibilidad de la misma.

- **La Confidencialidad:** Es que la información sea accesible sólo para quien que está autorizado.
- **La Integridad:** Es que la información sólo puede ser creada y modificada por quien esté autorizado a hacerlo.
- **La Disponibilidad:** Es que la información debe ser accesible para su consulta o modificación cuando se requiera.

Seguridad Informática: Conjunto de políticas y mecanismos que permiten garantizar la confidencialidad, la integridad y la disponibilidad de los recursos de un sistema.

Con la intención de garantizar la integridad y la confidencialidad, se implementan algoritmos criptográficos, de ahí la importancia de la criptografía en la seguridad informática en los sistemas actuales.

Criptografía

La criptografía también conocida como escritura secreta, se ha utilizado para proteger los secretos de la humanidad desde los tiempos en que el hombre empezó a escribir (Kahn, 1974). Él identifica el uso de la criptografía en Egipto en 1900 antes de cristo, en Mesopotamia en 1500 antes de cristo y en la escritura de la biblia en el año 500 antes de cristo.

La palabra Criptografía proviene del griego kryptos, que significa oculto, y graphein, que significa escribir, etimológicamente su significado es "escritura oculta". Es el proceso de transformar un texto llano o plano en texto cifrado, bajo la reflexión de Darthje (2012), es el arte y la ciencia de hacer las comunicaciones ininteligibles para todos excepto para el receptor, quien poseerá la llave para descifrar el mensaje original.

Un mecanismo de encriptación consiste en un algoritmo de cifrado que utiliza una clave para transformar un mensaje de su forma original (texto plano) a un nuevo mensaje cifrado (texto cifrado) de la forma $f(m,c) = x$, donde f es el algoritmo de cifrado, m el mensaje original y x el mensaje cifrado. Para conocer el contenido de este mensaje el receptor debe calcular la función inversa $f^{-1}(x,c) = m$ (Ramió, 2006).

Clasificación de la criptografía

De acuerdo con Cortés (2007) La Criptografía, puede clasificarse de acuerdo a:

a) **Su relación con la Historia en:** Criptografía Clásica y Criptografía Moderna.
b) **El procesamiento de la información:** Cifrado en Bloque y Cifrado en Flujo.
c) **El tipo de llave utilizada:** Cifrado Simétrico o llave privada y Cifrado Asimétrico o de Llave Pública.

Criptografía Clásica

El adjetivo de clásica, se debe básicamente a las primeras operaciones de sustitución y transposición o permutación de caracteres, con o sin clave pero siempre unido al concepto de clave secreta, en este capítulo sólo se analizarán los siguientes:

La Cifra de Cesar

Uno de los practicantes más famosos de la criptografía fue Julio Cesar, quien en el siglo I A.C. desarrollo una clave conocida como la cifra de cesar, que está basada en la rotación fija de las letras del alfabeto, asigna a cada letra otra que está separada un número fijo de letras. La Figura 2.1 ilustra el funcionamiento (Jaworski & Perrone, 2001).

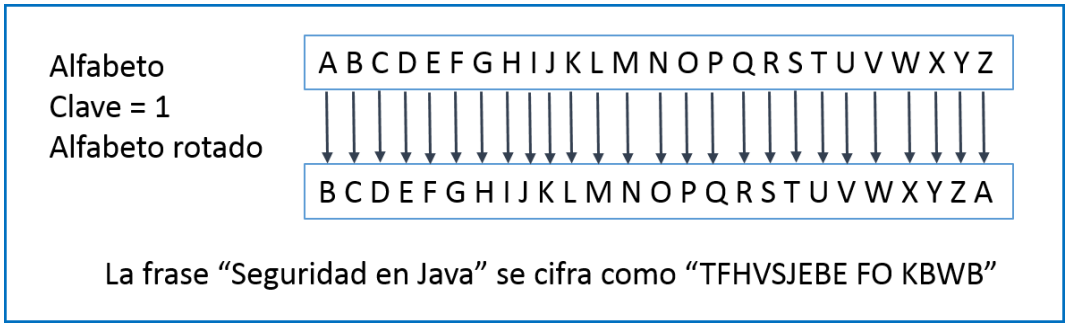


Figura 2.1 La Cifra de César. (Jaworski, 2001).

El Cifrador de Polybios

A mediados del siglo II A.C., se encuentra el cifrador por sustitución de caracteres más antiguo que se conoce (Ramió, 2006). Atribuido al historiador griego Polybios, que consistía en hacer corresponder a cada letra del alfabeto, dos números que indicaban la fila y la columna de una tabla de 5 x 5, en la cual se posicionaba (Tabla 2.1).

Tabla 2.1 Cifrador de Polybios. (Ramió, 2006).

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

De tal forma que la codificación de la frase “buenos días” corresponde a “124515333443” “14241143”.

La Escítala

En siglo V a.c. un antiguo pueblo griego llamado los Lacedemonios utilizaba este método que consiste en una cinta y dos varas idénticas que tenían tanto el emisor como el receptor. La cinta se enrollaba en todo el largo de la vara y en ella se escribía un mensaje en forma longitudinal como se muestra en la Figura 2.2.

Figura 2.2 La Escítala. (Granados, 2006).

Una vez escrito el mensaje, la cinta se desenrollaba y era entregada al mensajero, si este era interceptado por un enemigo, lo único que conseguía ver, era un conjunto de caracteres escritos de forma aleatoria, el resultado era el mismo incluso si el enemigo intentaba enrollar la cinta en un bastón con diámetro diferente (Granados, 2006).

Criptografía Moderna

Gracias a la aparición de las computadoras es posible incrementar la velocidad de grandes cálculos, la criptografía moderna hace uso de algunas propiedades matemáticas como; la dificultad del cálculo del logaritmo discreto o el problema de la factorización de grandes números y la operación se realiza sobre una cadena de bits y no sobre caracteres, gracias a este crecimiento actualmente es posible cifrar datos y comunicaciones. La Criptografía Moderna se puede clasificar en dos grandes grupos: la Criptografía de Llave Secreta o Asimétrica y la Criptografía de Llave Pública o Asimétrica (Granados, 2006).

Cifrado en Bloque

El proceso de cifrado se aplica sobre un grupo de bits de texto claro y producen un bloque de tamaño fijo de texto cifrado, generalmente del mismo tamaño que la entrada. La asignación se realiza mediante sustituciones y permutaciones sobre el texto claro hasta obtener el texto cifrado y deben asignarse uno a uno, para lograr que el proceso sea reversible. Existen dos clases importantes de este cifrado que son: cifrado por sustitución y cifrado por transposición (Stinson, 2006).

- **Cifrado de por Sustitución:** Se realiza remplazando la información de entrada (texto plano) ya sea de forma individual o en bloques manejables.
- **Cifrado por Transposición:** Es el cambio de posición del texto plano siguiendo un esquema bien definido.

Granados (2006) menciona que este tipo de criptografía está basado en el diseño propuesto por Horst Feistel en los años 70 y consiste en un bloque de tamaño N bits que se divide en dos bloques A y B, a partir de ahí se comienza el proceso de cifrado al bloque B en función de una subllave K_1 generada a partir de la llave secreta, después se mezcla el bloque A con el resultado de la función mediante un XOR. Luego se permutan los bloques y se repite el proceso n veces. Finalmente se unen los dos bloques resultantes. Su funcionamiento se ilustra en la Figura 2.3.

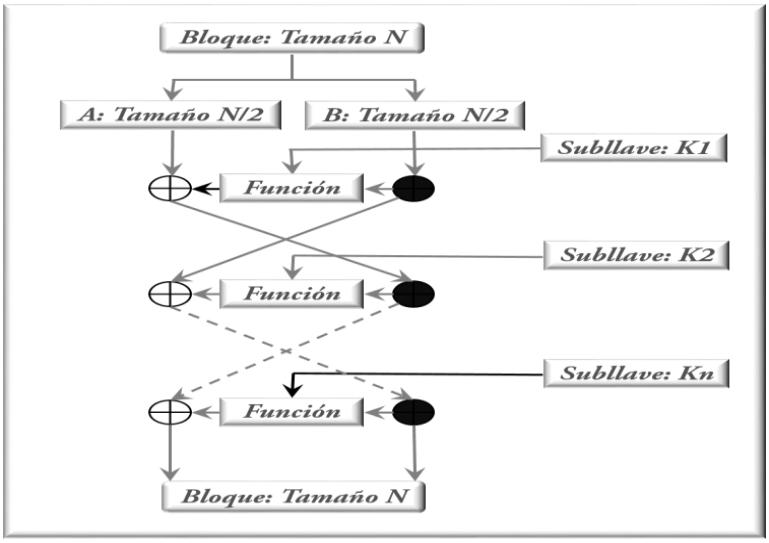


Figura 2.3 Cifrado por bloques de Feistel. (Elaboración propia).

Cifrado en Flujo

Fundamenta su funcionamiento en un algoritmo que genera una secuencia pseudoaletoria de bits y es aplicado bit a bit o byte a byte sobre el texto plano de forma incremental. Además tanto emisor como receptor utilizan un generador de claves que produce una secuencia de bits igual a la longitud del mensaje que es empleada como la clave durante el proceso (Stinson, 2006).

Cifrado Simétrico

Para este tipo de cifrado la clave secreta es la misma que se utiliza para el cifrado y para el descifrado, Astilla (2009) menciona que este sistema tiene dos características muy importantes; La primera es la clave secreta que se utiliza para realizar operaciones de transposición y sustitución con la finalidad de generar confusión y difusión de la información que se pretende cifrar. La segunda es que este algoritmo utiliza varias sustituciones y transformaciones para procesar el texto claro, el diseño de este cifrado se muestra en la Figura 2.4.

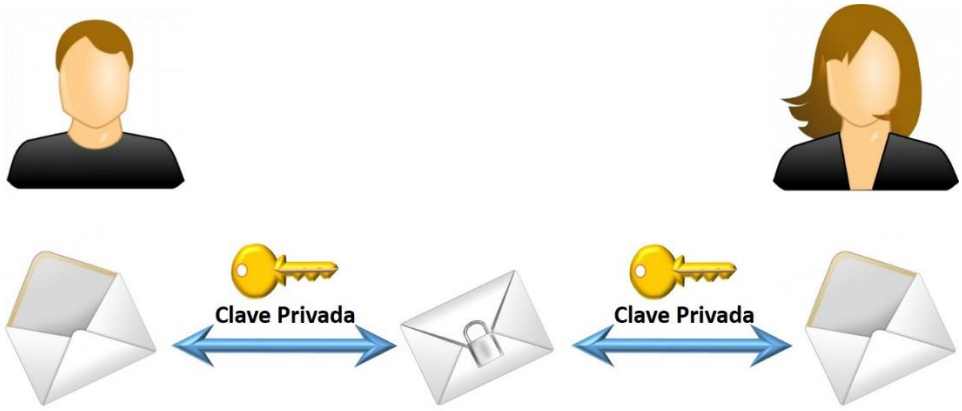


Figura 2.4 Sistema de Cifrado Simétrico. (Elaboración propia).

Cifrado Asimétrico

En este esquema de cifrado se utilizan dos claves; una pública y una privada el emisor cifra con su clave pública y el receptor descifra con su clave privada y viceversa, estas claves tienen métodos matemáticos especiales y complejos que

hacen que las dos llaves tengan relación entre sí logrando que sin la otra parte el criptosistema, no funcione, de esta manera se descarta la necesidad de un canal de comunicación seguro, un diseño de este funcionamiento se muestra en la Figura 2.5 (López, 2010).

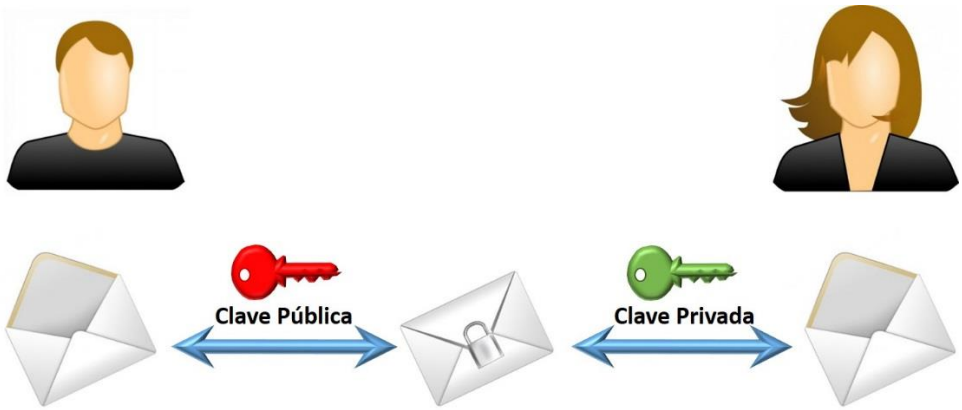


Figura 2.5 Sistema de Cifrado Asimétrico. (Elaboración propia).

Secure Sockets Layer/ Transport Layer Security

De acuerdo con Dierks y Rescorla (2008) este protocolo permite la confidencialidad mediante el cifrado de los datos y generando un canal de comunicación seguro mantiene la integridad de los mensajes además proporciona autenticación entre cliente (navegador web) y servidor mediante certificados y firma digital, todo esto de forma transparente al usuario. En la actualidad ha sido sustituido por TLS el cual está basado en SSL y son totalmente compatibles, Freier et al. (2011). La Figura 2.6 muestra la ubicación del protocolo SSL/TLS dentro de las capas del modelo OSI (Open Systems Interconnection, 1994) se puede apreciar que se ejecuta en una capa por debajo los protocolos de aplicación como HTTP, FTP, Telnet y otros. Y por encima del protocolo de transporte TCP.

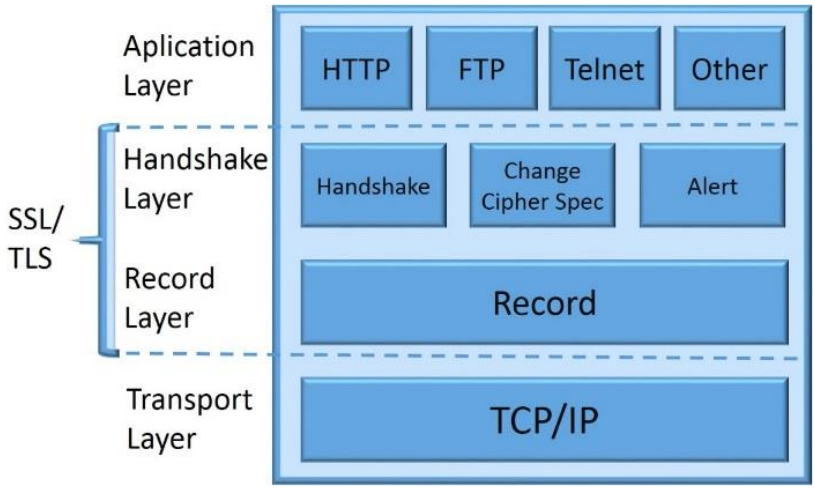


Figura 2.6 Ubicación del protocolo SSL/TLS dentro del modelo OSI. (Microsoft, 2003).

Un servidor HTTP ordinario no entiende la petición SSL/TLS e interpreta estos datos como basura por lo que el cliente debe tener una función adicional para que el proceso funcione correctamente, es por ello que se utiliza un localizador uniforme de recursos (URL) el cual comienza con https (Figura 2.7) en lugar de http para indicar que un cliente quiere usar SSL/TLS, es por lo anterior que HTTP y HTTPS utilizan puertos separados 80 y 443 respectivamente (Hayoz & Ultes-Nitsche, 2003). Ya que SSL/TLS depende de un mecanismo de entrega de paquetes fiable no es capaz de correr a través de UDP o directamente a través de IP.

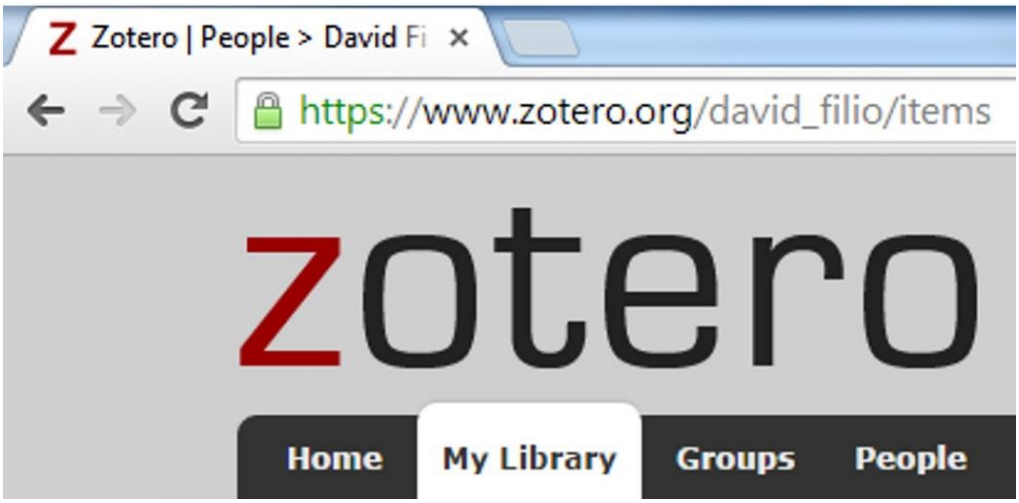


Figura 2.7 Uso de protocolo HTTPS. (Zotero, 2014).

SSL/TLS negocia entre servidor y cliente los algoritmos que se usará durante la comunicación una estructura general de su funcionamiento se muestra en la Figura 2.8, básicamente realiza el Intercambio de claves públicas y de autenticación basadas en certificados digitales. Cifra el tráfico para lo cual regularmente utiliza un cifrado simétrico pues es mucho más rápido que un asimétrico (Ramírez L & Espinosa M, 2011).

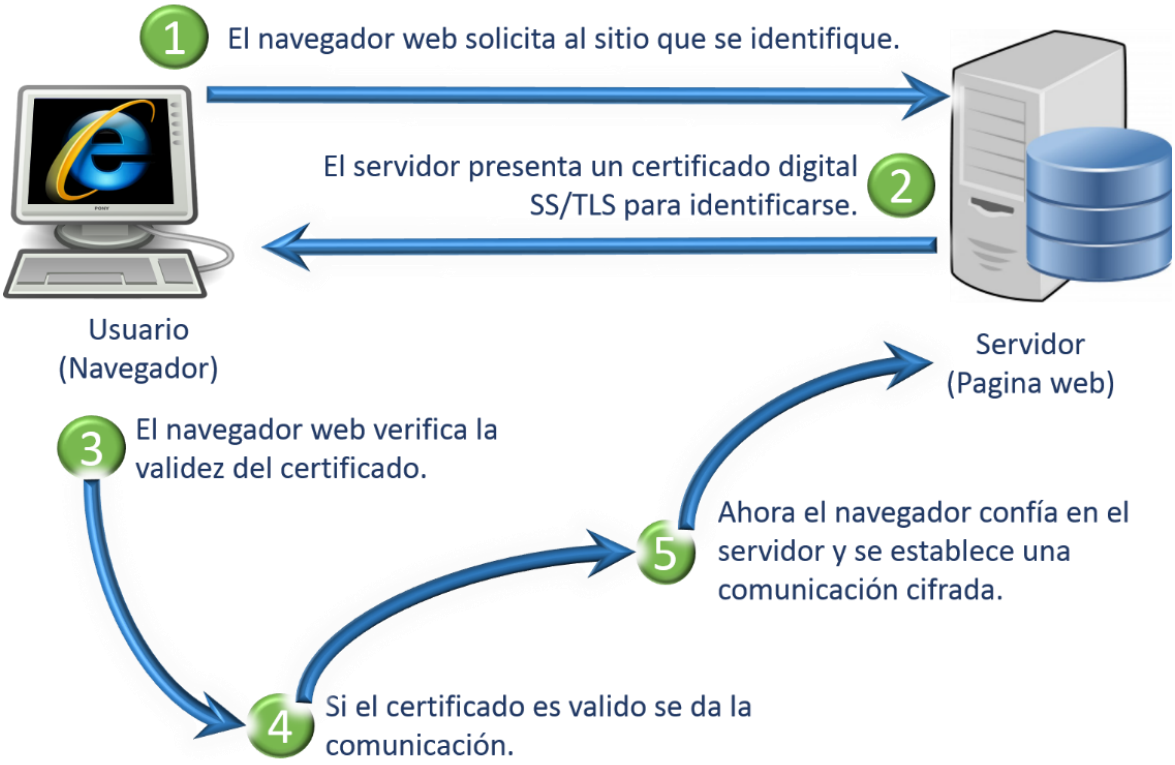


Figura 2.8 Funcionamiento general del protocolo SSL/TLS. (Elaboración propia).

Stallings en (2010) menciona que SSL/TLS para su funcionamiento consta de cuatro sub protocolos; SSL Handshake Protocol, SSL Change Cipher Spec Protocol y SSL Alert Protocol que operan sobre la capa de aplicación, y uno más; SSL Record Protocol que trabaja en la capa de red.

Handshake Protocol

Su funcionamiento es el siguiente. El navegador solicita al servidor que se identifique, el servidor muestra un certificado digital para presentarse, el navegador verifica la validez del certificado, si es válido se da la comunicación segura, de lo contrario, esta falla. Este protocolo se ejecuta antes de que se realice cualquier intercambio de datos y también permite acordar los algoritmos y claves que se utilizaran de forma segura, incluyendo las suites de cifrado y longitud de clave que ambos admiten (Figura 2.9), pareciera que el cifrado más potente disponible es el que se debería utilizar siempre, pero en realidad no es así esto depende de muchos factores tales como el consumo de recursos, la velocidad, el valor en tiempo de los datos, restricciones gubernamentales, patentes y problemas de licencias. Tracy et al. (2007).

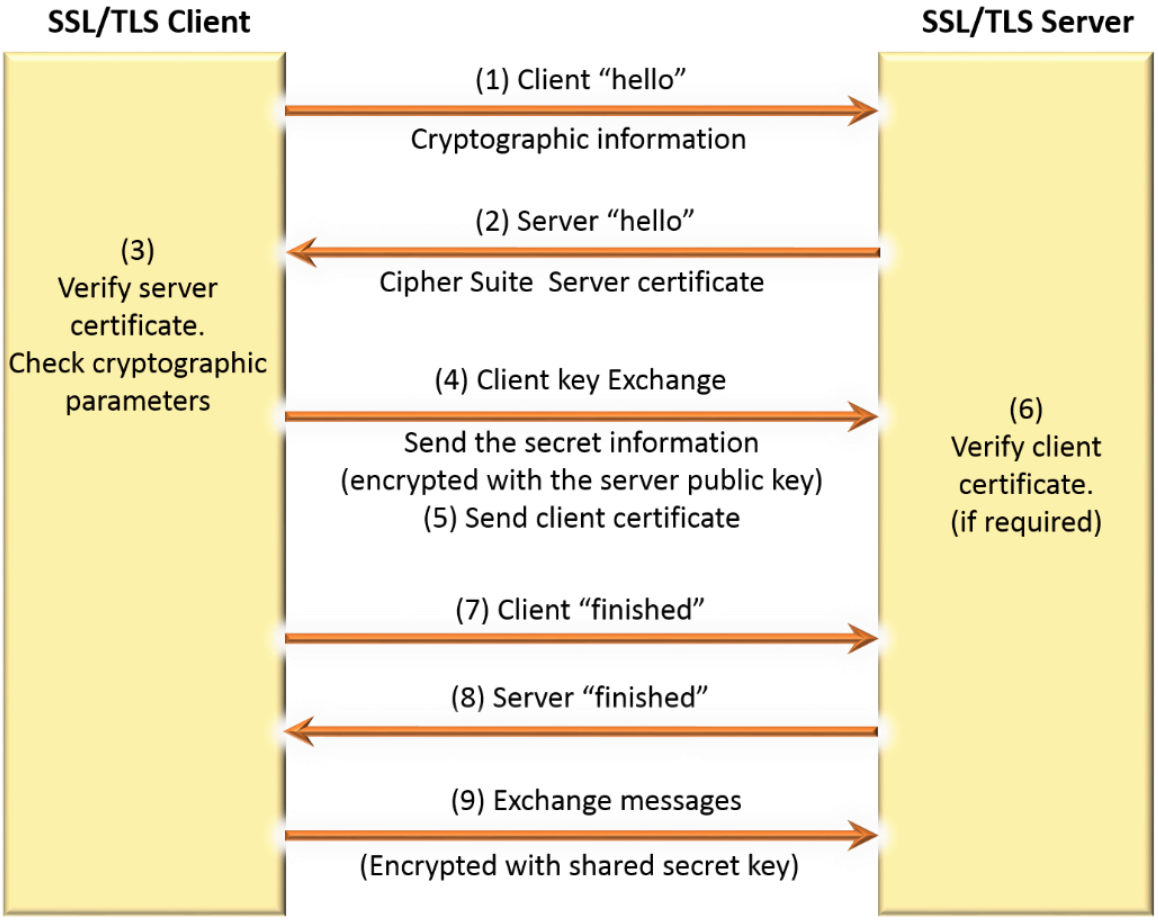


Figura 2.9 Funcionamiento del Handshake Protocol. (Adaptada de IBM, 2010).

Change Cipher Spec Protocol

Consiste en un solo mensaje de un solo byte con el valor 1, funciona como estado de transición entre cliente y servidor, para notificar que los siguientes mensajes van a ser protegidos bajo los nuevos parámetros de seguridad negociados (Stallings, 2010).

Alert Protocol

Permite notificar que la conexión va a ser cerrada o que ha ocurrido un error durante la conexión, cada mensaje de alerta siempre es cifrado y consta de dos bytes el primero indica la gravedad de una alerta cuyo valor puede ser de 1 o 2 donde 1 equivale a una advertencia y 2 a un error fatal, si es fatal termina inmediatamente la conexión. El segundo byte contiene un código que indica específicamente el tipo alerta, se puede consultar detalladamente el listado de alertas en (Rescorla, 2001).

Record Protocol

Este protocolo especifica la forma de encapsular los datos transmitidos y recibidos incluidos los de negociación, fragmenta los datos en bloques manejables, opcionalmente los comprime, calcula y aplica un código de autenticación de mensaje (MAC), esté se añade al fragmento de datos generando un paquete el cual es cifrado utilizando criptografía simétrica y luego se añade un Record Header que contiene; el tipo de contenido, la longitud y la versión SSL/TLS, a continuación se transmiten los bloques resultantes. Este funcionamiento se ilustra en la Figura 2.10.

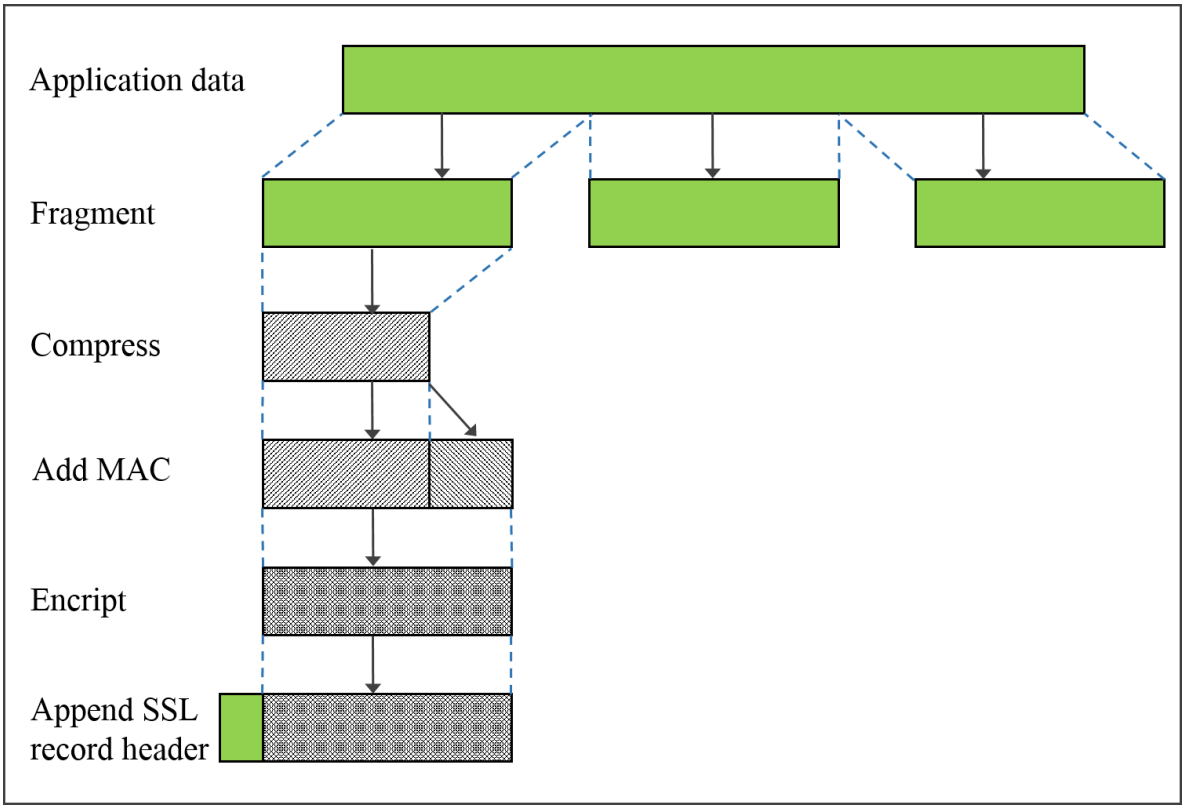


Figura 2.10 Funcionamiento del Record Protocol. (Adaptada de Stallings, 2010).

Los datos recibidos son desencriptados, verificados, descomprimidos y reensamblados, es importante aclarar que mientras no se hayan acordado los algoritmos en la fase de negociación (Hand Shake) los registros no se cifran, ni autentican en ese momento tendrán valores nulos (Hayoz & Ultes-Nitsche, 2003).

Data Encryption Standard (DES)

Es un algoritmo de criptografía simétrica inventado inicialmente por IBM en 1970 bajo el nombre "Lucifer" después fue aprobado por el Instituto Nacional Americano de Estándares (ANSI) en el año 1973 y como estándar federal del mismo país por la FIPS (Federal Information Processing Standard) en el año 1977 (Stinson, 2006). En este trabajo se explica el funcionamiento de este algoritmo en tres partes; la primera es una visión general del algoritmo y la segunda trata sobre cómo se generan las llaves y la última cómo se realiza una ronda o round.

Visión general del algoritmo

Según la norma (FIPS PUB 46-3, 1999), este algoritmo utiliza el cifrado por bloque, se trata de un algoritmo que descompone el mensaje original en bloques de 64 bits de longitud que equivalen a ocho símbolos en codificación ASCII, Durán et al. (2000), en este proceso el bloque de entrada también conocido como input o texto plano es seguido por una permutación inicial IP, y su división en dos bloques de 32 bits L_0 y R_0, en función de la llave K cuya longitud es de 56 bits $f(R,K)$, los cuales sufren un proceso de permutaciones, a esta operación se conoce como ronda o round y se aplica este proceso 16 veces sobre el texto plano, las mismas operaciones en las cuales la información es combinada con las 16 sub llaves generadas a partir de K. En la última ronda las dos mitades resultantes L_{15} y R_{15}, vuelven a juntarse y pasan por la permutación inversa $IP^{(-1)}$ que producirá una salida u output de 64 bits, la que se conocerá como texto cifrado, finalmente para descifrar; basta con realizar el proceso inverso. Un esquema general de su funcionamiento se aprecia en la Figura 2.11.

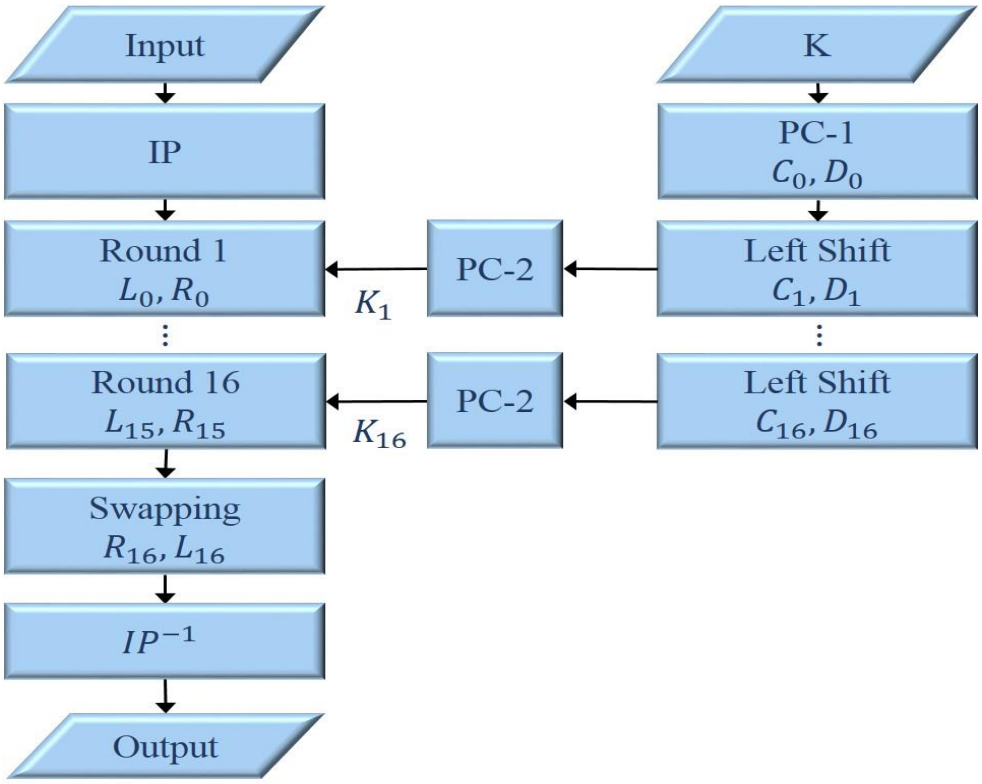


Figura 2.11 Visión general DES. (Elaboración propia).

Cómo se generan las llaves

El algoritmo para generar las llaves se compone de los siguientes pasos:

- Ingresan 64 bits es decir la $K = 64\ Bits$.
- A esos 64 bits se le aplica la Permuted Choice 1 $(PC - 1)$ para obtener un resultado de 56 Bits.
- El resultado anterior se divide en dos bloques $C_0\ y\ D_0$ cada uno de 28 Bits.
- Ahora se realizan corrimientos a la izquierda (pueden ser 1 o 2) según indique la tabla $Left\ Shifts$ (FIPS PUB 46-3, 1999) y obtendremos $C_1\ y\ D_1$ cada uno de 28 Bits.
- Se juntan $C_1\ y\ D_1$ y se aplica la Permuted Choice 2 $(PC - 2)$, el resultado será una K_1 de 48 Bits.
- Para obtener K_2 se sigue el procedimiento anterior, a partir de $C_1\ y\ D_1$ y así sucesivamente hasta obtener K_{16}.

Una descripción grafica de este algoritmo se aprecia claramente en la Figura 2.12.

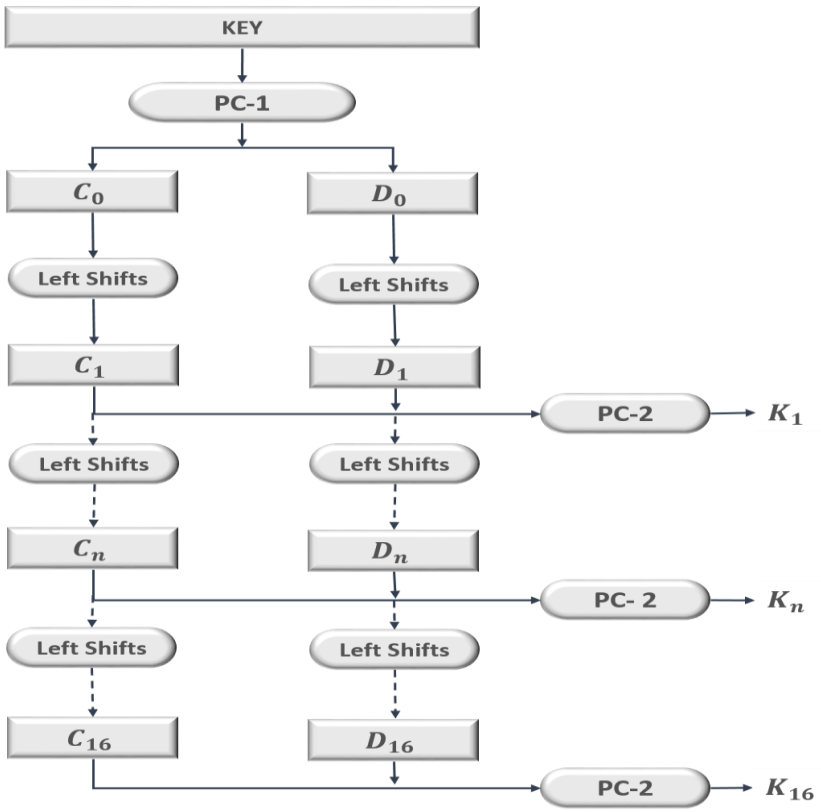


Figura 2.12 Cálculo de las llaves (Keys). (Elaboración propia).

Cómo se realiza una ronda DES

Una ronda (Round) DES se compone de los siguientes pasos y su funcionamiento se muestra en la Figura 2.13.

- La entrada (input) es de 64 bits.
- Estos se dividen en dos bloques L_0 y R_0 de 32 Bits cada uno.
- El lado bloque derecho se iguala al izquierdo $R_0 = L_0$
- Se realiza la función $f(R, K)$ la que inicia pasando los 32 bits de R_0 por una tabla de permutación llamada E (E bit-Selection Table) el resultado será de 48 Bits.
- Al resultado anterior se le aplica la función XOR junto con la K_1 que también es de 48 Bits.
- Los bits resultantes pasan por las $S\ Boxes$ y se obtendrá una cadena de 32 bits para mayor información de este paso se recomienda consultar la (FIPS PUB 46-3, 1999).
- Estos 32 Bits se pasan por P (the primitive function).
- Después se le aplica la función XOR junto con L_0 y el resultado será llamado R_1 y este proceso se repite 16 veces.

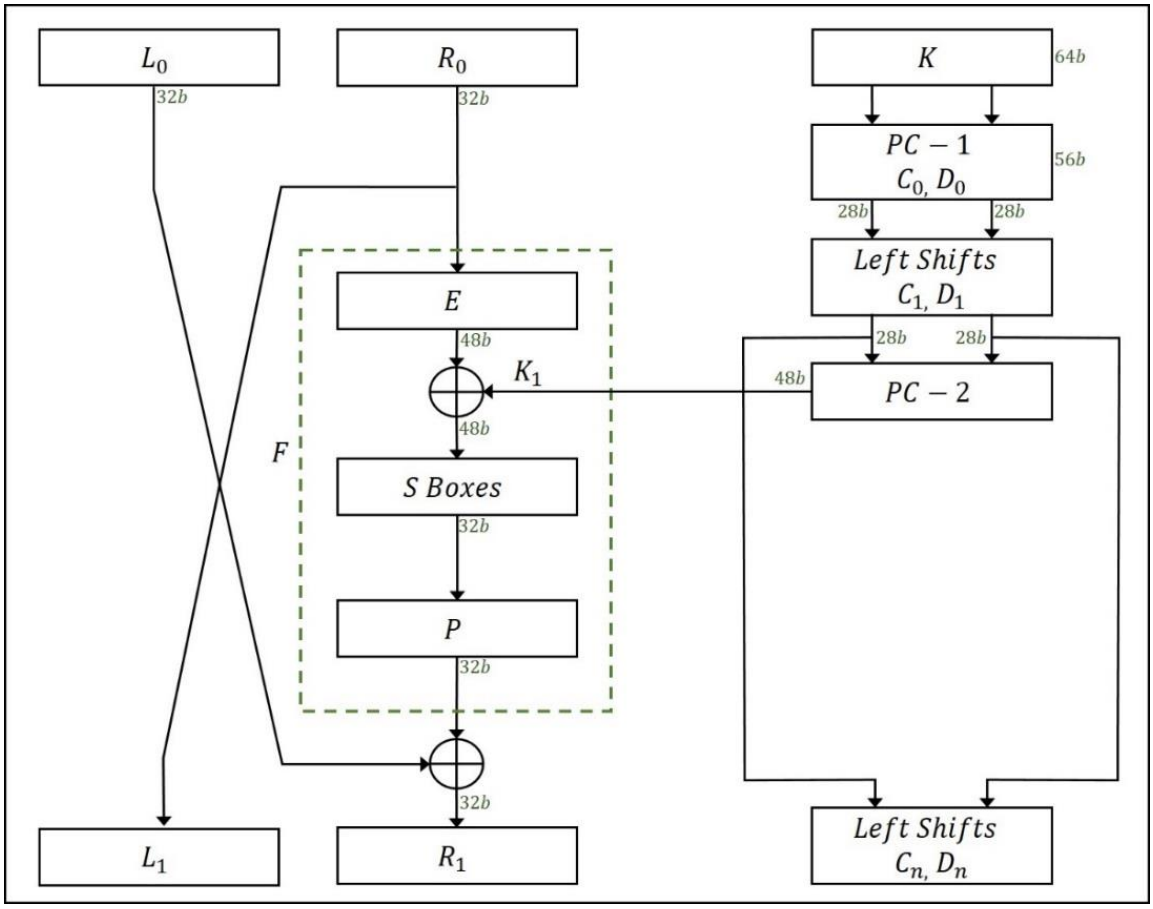


Figura 2.13 Una ronda DES. (Elaboración propia).

Triple DES

En el año 1998 utilizando una maquina llamada “DES Cracker” la EFF (Foundation Electronic Frontier, 1998) rompió el cifrado DES mediante un ataque por fuerza bruta el cual consiste en probar todas las claves posibles entonces una clave de 56 bits ya no era suficiente para evitar un ataque de este tipo, lo que dio origen a Triple DES.

Tuchman (2012) menciona que Triple DES realiza el mismo procedimiento que DES, pero este se repite tres veces de la forma Encrypt-Decrypt-Encrypt (E-D-E) y Consta de tres llaves K_1, K_2 y K_3 de 64 bits cada una, y de las cuales K_1 y K_2 deben ser diferentes entre sí. Los datos se cifran con K_1, se descifran con K_2, y

finalmente se cifran de nuevo con K_1, en este último paso se puede utilizar una llave diferente que se llama K_3.

Si se utiliza $K_1 = K_3$ la complejidad de Triple DES es de 2^{112} y si $K_1 \neq K_2 \neq K_3$ la complejidad es de 2^{168}, por lo que actualmente los ataques de fuerza bruta son efectivamente imposibles, sin embargo debido a que se utiliza una longitud de clave más grande, Triple DES tarda más en realizar las operaciones de cifrado esto triplica su tiempo de ejecución, por lo que se le considera un algoritmo lento (Schneier, 1995) la figura 2.14 representa este algoritmo.

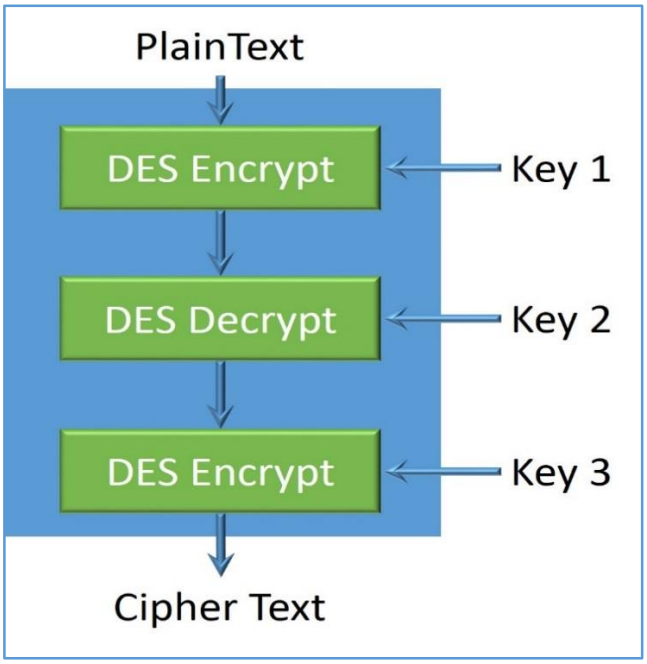


Figura 2.14 Esquema de funcionamiento Triple DES. (Elaboración propia).

Numero trascendente *e*

Rosales (2010) menciona que el concepto de número trascendente o de función trascendente se ha formado poco a poco a medida que ha ido progresando el álgebra. La palabra “Trascendente” fue utilizada por primera vez por Leibniz en 1704.

Baker (1990) define un número trascendente como aquel que no puede ser obtenido mediante la resolución de una ecuación algebraica con coeficientes racionales. Según Balanzario (2003) un número que no es algebraico se llama trascendente. En 1873 Ch. Hermite pudo demostrar que el número $\boldsymbol{e}$ es trascendente, este número fue utilizado por primera vez por el matemático John Napier pero fue Leonard Euler quien descubrió muchas de sus propiedades. El valor de $\boldsymbol{e}$ esta dado por la ecuación (1).

$$\mathrm{e} = \lim_{n\to\infty}\left(1+\frac{1}{n}\right)^{n} = 2{,}718281828459045235360287471352 7 \qquad (1)$$

El resultado anterior es una aproximación a sus primeras cifras, las cuales se ocupan en este trabajo para elaborar una permutación variable PV y generar su inversa PV^{-1}.

CAPÍTULO III: METODOLOGÍA EXTREME PROGRAMMING (XP)

En este capítulo se habla de la metodología utilizada para llevar el desarrollo de los algoritmos en lenguaje de programación Java. Conocida popularmente como XP (Extreme Programming), es una de las metodologías ágiles de desarrollo rápido más utilizadas basada en los principios de simplicidad, comunicación y retroalimentación fue diseñada para ser utilizada por equipos pequeños que necesitan realizar desarrollo de software rápido y se programa en parejas, se basa en el trabajo orientado directamente al objetivo (Pressman, 2010). Es por eso que XP representa la mejor alternativa para el desarrollo de los programas de computadora que trata esta tesis en especial porque el proyecto es pequeño y se cuenta con poco tiempo y numero de programadores.

Beck (2000) hace gran énfasis en el equipo de desarrollo y en el trabajo en equipo, propone que un representante del cliente trabaje en forma conjunta con el equipo de desarrollo además del trabajo de los programadores en parejas y las reuniones frecuentes para evaluar el estado, avance de las actividades y retroalimentación. A continuación se describen las características más significativas de la Programación Extrema.

Elementos que componen la Programación Extrema XP

Los elementos utilizados por XP son; Historias de Usuario, Tarjetas de Ingeniería, Pruebas de Aceptación y Tarjetas CRC las cuales se describen enseguida.

Historias de usuario (User Stories)

Benk (2000) menciona que las Historias de Usuario son tarjetas de papel que describen brevemente las características, requisitos y funcionalidad del sistema requerido. Cohn (2004) refiere que las Tarjetas de Usuario deben ser valiosas tanto para cliente como para el usuario. La Tabla 3.1 muestra un formato general de estas tarjetas. Es importante aclarar que:

- Cada historia debe llevar el número y nombre de la historia así como nombre del usuario y programador responsable.
- La prioridad en negocio y riesgo en desarrollo se establece como;

$$\{\text{muy alta} = 4,\ \text{alta} = 3,\ \text{media} = 2\ \text{y}\ \text{baja} = 1\}.$$

- Los Puntos Estimados se calculan de acuerdo a la fórmula 2 y el resultado se mide en semanas.

$$(0.25 * prioridad\ en\ negocio) + (0.75 * riesgo\ en\ desarrollo) \qquad (2)$$

- En el apartado Descripción; el usuario explica al programador que quiere y como quiere que funcione.
- Cada iteración toma de una a cuatro semanas en ejecución.

Tabla 3.1 Tarjeta Historia de Usuario. (Adaptada de Cohn, 2004).

Historia de Usuario	
Número:	**Usuario:**
Nombre de la Historia:	
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Puntos Estimados:	**Iteración Asignada:**
Programador Responsable:	
Descripción:	
Observaciones:	

Tareas de ingeniería

Estas tarjetas describen las actividades a realizar y se derivan de los procesos descritos las tarjetas de usuario y deben llevar nombre de la tarea a realizar (Tabla 3.2).

Tabla 3.2 Tarjeta de Ingeniería. (Adaptada de Cohn, 2004).

Tarea de Ingeniería	
Número de Tarea:	**Historia de Usuario (No. y Nombre):**
Nombre de la Tarea:	
Prioridad en Negocio:	**Riesgo en Desarrollo:**
Tipo de Tarea: **Desarrollo / Corrección / Mejora / Otra**	**Puntos Estimados:**
Fecha Inicio:	**Fecha Fin:**
Programador Responsable:	
Descripción:	

Tarjetas CRC

Cada una de estas tarjetas (Clase-Responsabilidad-Colaborador) representa una clase en la programación orientada a objetos y especifica sus responsabilidades (que es lo que hace) y la colaboración que tiene con otras clases para su funcionamiento. Son utilizadas a nivel de diseño se colocan en una mesa y así cada integrante observa cómo interactúan las clases de cada tarjeta (Sommerville, 2011). Un ejemplo de estas tarjetas se muestra en la Tabla 3.3.

Tabla 3.3 Tarjeta CRC. Elaboración propia tomada de (Cohn, 2004).

Nombre de la Clase	
Responsabilidades	**Colaboración**

Pruebas de aceptación

Un formato de estas tarjetas se observa en la Tabla 3.4. Estas las escribe el cliente en base a; la acción a probar, los datos de prueba y el resultado esperado conforme a los siguientes pasos;

1. Identificar todas las acciones en la historia.
2. Por cada acción escribir dos pruebas.
3. Debe proporcionar los datos de entrada que deberían tener éxito, y llenar el resultado satisfactorio.
4. Suministrar los datos de entrada que tengan una acción fallida, y llenar la respuesta que debería tener.

Tabla 3.4 Tarjeta Prueba de Aceptación. (Adaptada de Cohn, 2004).

Prueba de Aceptación	
Código:	**Historia de Usuario (Nro. y Nombre):**
Nombre:	
Descripción:	
Condiciones de Ejecución:	
Entrada / Pasos de Ejecución:	
Resultado Esperado:	
Evaluación de la Prueba:	

Proceso de la Programación Extrema XP

La Figura 3.1 muestra el proceso de la metodología XP. Es importante mencionar que durante todo el proceso tanto cliente como programador aprenden, no se recomienda cualquier forma de presión sobre el programador, ya que se pierde calidad en el software o pueden no cumplirse los plazos (Pressman, 2010).

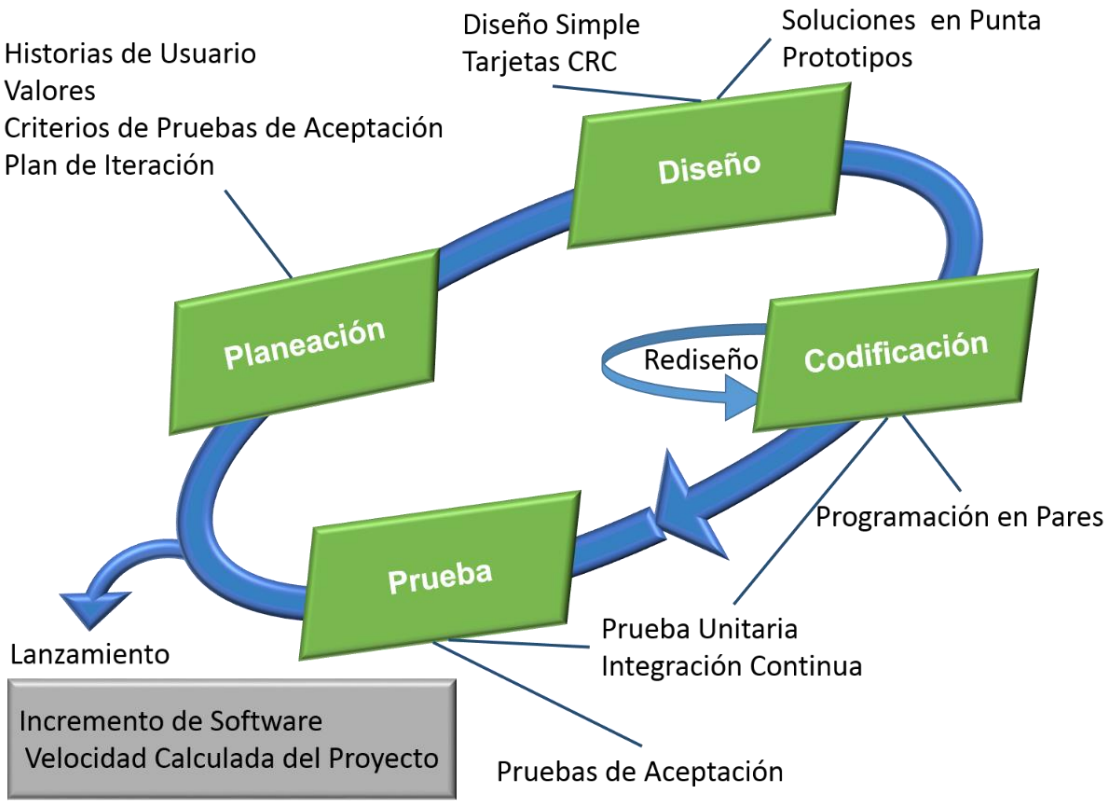


Figura 3.1 Proceso de la Programación Extrema. (Adaptada de Stallings, 2010).

Planeación. También llamada (juego de la planeación) es la actividad realizada por cliente y desarrolladores para recabar todos los requerimientos, sirve para entender y conocer las características principales y funcionalidad del sistema requerido, mediante el uso de las tarjetas de Historia de Usuario.

Diseño. Es aquí donde se ocupan las tarjetas CRC que sirven para identificar y organizar las clases orientadas a objetos y son el único producto para el trabajo de diseño en XP.

Codificación. Después de realizar las tarjetas de usuario y el diseño preliminar, se realizan pruebas a cada una de las historias con la intención de que el programador este mejor capacitado y se concentre en lo que debe implementar para pasar la prueba, después es el momento de codificar en parejas.

Pruebas. En XP existen tres tipos de prueba**.** Las Pruebas Unitarias se realizan continuamente como se ha dicho incluso antes de la codificación, Wells (1999) dice: “Corregir pequeños problemas cada cierto número de horas toma menos tiempo que resolver problemas enormes justo antes del plazo final.” Las Pruebas de Cliente son especificadas por el cliente en base a las características y funcionalidad del sistema y las Pruebas de Aceptación se derivan de las historias de usuario.

Como parte de la aplicación de las etapas o fases de planeación y diseño de la metodología XP se documentó lo siguiente:

Planeación y diseño del algoritmo de cifrado DES

Planeación

En esta etapa se ordenan y asigna prioridad a las ideas, características y requisitos, y se descomponen en Tarjetas de Usuario y de Ingeniería y se delimita la fecha de entrega.

Requerimientos. Los requisitos del programa por orden y prioridad son; recibir una cadena de texto plano, cifrar ese texto usando el algoritmo DES, posteriormente descifrarlo e imprimirlo en pantalla. Se requiere que el programa se ejecute sobre cualquier sistema operativo por lo que se elige programar en lenguaje Java que es multiplataforma (Deitel & Deitel, 2008), se trabaja en el entorno de desarrollo Netbeans. Las acciones que seguirá el sistema se descomponen en las siguientes dos Historias de Usuario (Tablas 3.5 y 3.6):

Tabla 3.5 Historia de usuario Cifrado de la Información. (Elaboración propia).

Historia de Usuario	
Número: 1	**Usuario:** Usuario A
Nombre de la Historia: Cifrado de la Información	
Prioridad en Negocio: alta	**Riesgo en Desarrollo:** alto
Puntos Estimados: 3	**Iteración Asignada:** 1
Programador Responsable: David Filio y Rolando Flores	
Descripción: Se requiere un programa de computadora que realice los pasos necesarios para cifrar la información que se le ingrese.	
Observaciones: Ninguna.	

Tabla 3.6 Historia de usuario Descifrado de la Información. (Elaboración propia).

Historia de Usuario	
Número: 2	**Usuario:** Usuario B
Nombre de la Historia: Descifrado de la Información	
Prioridad en Negocio: alta	**Riesgo en Desarrollo:** alto
Puntos Estimados: 3	**Iteración Asignada:** 2
Programador Responsable: David Filio y Rolando Flores	
Descripción: Se requiere que el mismo programa de computadora realice los pasos necesarios para descifrar la información que anteriormente se cifró.	
Observaciones: Ninguna.	

Para lograr el objetivo de las Historias de Usuario es apropiado descomponerlas en las siguientes ocho Tarjetas de Ingeniería (Tablas 3.7 a la 3.13):

Tabla 3.7 Tarjeta de Ingeniería Generar Tablas de Permutación. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 1	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Generar Tablas de Permutación	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 02/05/2014	**Fecha Fin:** 02/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar una clase que contenga las tablas de permutación que recorrerá la información durante el proceso de su cifrado, que de acuerdo con la norma son; E, IP, IP_1, LEFTSHIFTS, P, PC1, PC2 y SBoxes.	

Tabla 3.8 Tarjeta de Ingeniería Conversión a Hexadecimal. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 2	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Conversión a Hexadecimal	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 05/05/2014	**Fecha Fin:** 05/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método que convierta la información a hexadecimal.	

Tabla 3.9 Tarjeta de Ingeniería Conversión a Binario. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 3	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Conversión a Binario	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 05/05/2014	**Fecha Fin:** 05/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Para que la información pueda ser modificada se requiere que este a su nivel más mínimo de información el cual se conoce como bit y eso es lo que hará esta tarea.	

Tabla 3.10 Tarjeta de Ingeniería Generar las Keys. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 4	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Generar las Keys	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 06/05/2014	**Fecha Fin:** 07/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método que a partir de la llave de ingreso genere las 15 sub-llaves que se ocupan para cifrar la información.	

Tabla 3.11 Tarjeta de Ingeniería Método Función. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 5	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Método Función	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 08/05/2014	**Fecha Fin:** 09/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método que realice las funciones que requiere el algoritmo DES tanto para cifrar como para descifrar y realizar la interacción con las tablas de permutación y keys.	

Tabla 3.12 Tarjeta de Ingeniería Cifrado de la Información. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 6	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Cifrado de la Información	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 12/05/2014	**Fecha Fin:** 14/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método que realice las funciones que requiere el algoritmo DES para cifrar la información.	

Tabla 3.13 Tarjeta de Ingeniería Descifrado de la Información. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 7	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Descifrado de la Información	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 15/05/2014	**Fecha Fin:** 16/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método que realice las funciones que requiere el algoritmo DES para descifrar la información.	

Tabla 3.14 Tarjeta de Ingeniería Método Principal. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 8	**Historia de Usuario (No. y Nombre):** 1 y 2 Cifrado y Descifrado de la información.
Nombre de la Tarea: Método Principal	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 19/05/2014	**Fecha Fin:** 21/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Elaborar un método principal el cual ejecutara todo el sistema y en el que se ingresan los datos; texto plano y key, también debe imprimir el resultado.	

Compromiso de entrega. Se considera que los programadores puedan ejecutar los requisitos de las Historias de Usuario en tres semanas y tres iteraciones.

Diseño

Desde el punto de vista de la simplicidad se busca diseñar un sistema funcional, la Figura 3.2 muestra el diseño general del sistema.

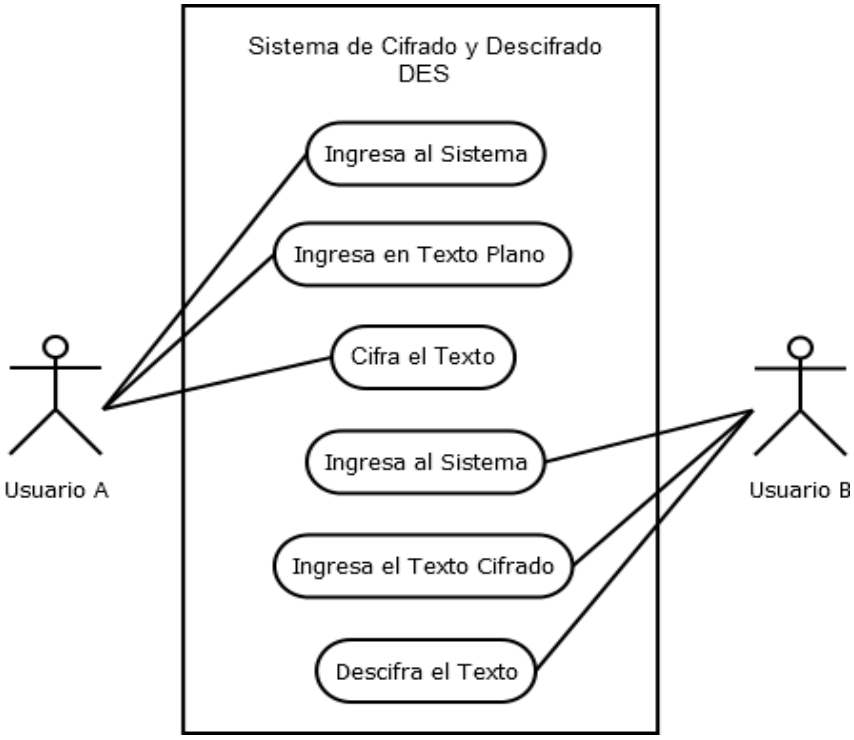


Figura 3.2 Diseño general del programa DES. (Elaboración propia).

Del diseño general se desprenden las siguientes ocho Tarjetas CRC (Tablas 3.15 a la 3.22) que se realizan lo más sencillo posible, es importante mencionar que para este trabajo cada tarjeta CRC corresponde a un método que posteriormente será implementado en código:

Tabla 3.15 Tarjeta CRC Tablas de Permutación. (Elaboración propia).

Clase	
DES	
Responsabilidades	**Colaboración**
private static long K[] public class DES static final int[] IP static final int[] IP_1 static final int[] E static final int[][] S static final int[] P static final int[] PC1 static final int[] LeftShifts static final int[] PC2	public static void setKey(long key) public static char[] cypher(char txt[]) public static char[] decypher(char txt[]) private static int f(int data,int ki) private static String binary(long x, int bits) private static String hex(char x) public static void main(String[] args)

Tabla 3.16 Tarjeta CRC para la Conversión a Hexadecimal. (Elaboración propia).

Método	
Conversión a Hexadecimal	
Responsabilidades	**Colaboración**
private static String hex(int x)	public static void main(String[] args)

Tabla 3.17 Tarjeta CRC para la Conversión a Binario. (Elaboración propia).

Método	
Conversión a Binario	
Responsabilidades	**Colaboración**
private static String binary(int x, int bits)	public static int[] cypher(int txt[]) public static int[] decypher(int txt[]) private static int f(int data, int ki)

Tabla 3.18 Tarjeta CRC para Generar las Keys. (Elaboración propia).

Método	
Generar las Keys	
Responsabilidades	**Colaboración**
public static void setKey(int key[])	public static void main(String[] args)

Tabla 3.19 Tarjeta CRC para el Método Función. (Elaboración propia).

Método	
Método Función	
Responsabilidades	**Colaboración**
private static int f(int data, int ki)	public static int[] cypher(int txt[]) public static int[] decypher(int txt[])

Tabla 3.20 Tarjeta CRC para el Cifrado de la Información. (Elaboración propia).

Método	
Cifrado de la Información	
Responsabilidades	**Colaboración**
public static int[] cypher(int txt[])	public static void main(String[] args)

Tabla 3.21 Tarjeta CRC para el Descifrado de la Información. (Elaboración propia).

Método	
Descifrado de la Información	
Responsabilidades	**Colaboración**
public static int[] decypher(int txt[])	public static void main(String[] args)

Tabla 3.22 Tarjeta CRC para el Método Principal. (Elaboración propia).

Método	
Método Principal	
Responsabilidades	**Colaboración**
public static void main(String[] args)	public static void setKey(int key[]) private static String hex(int x) public static int[] cypher(int txt[]) public static int[] decypher(int txt[])

Planeación y diseño del algoritmo de cifrado Triple DES

Planeación

A continuación se presentan los elementos necesarios para el desarrollo del sistema conforme a las fases de la metodología XP.

Requerimiento. Desarrollar un programa cuya finalidad es seleccionar un archivo de cualquier tipo, cifrarlo para hacer su contenido ilegible, posteriormente descifrarlo ambos procedimientos usando el algoritmo de cifrado Triple DES y durante los procesos de cifrado y descifrado medir el tiempo que tarda en ejecutar la acción. La Figura 3.3 muestra el diseño general del programa.

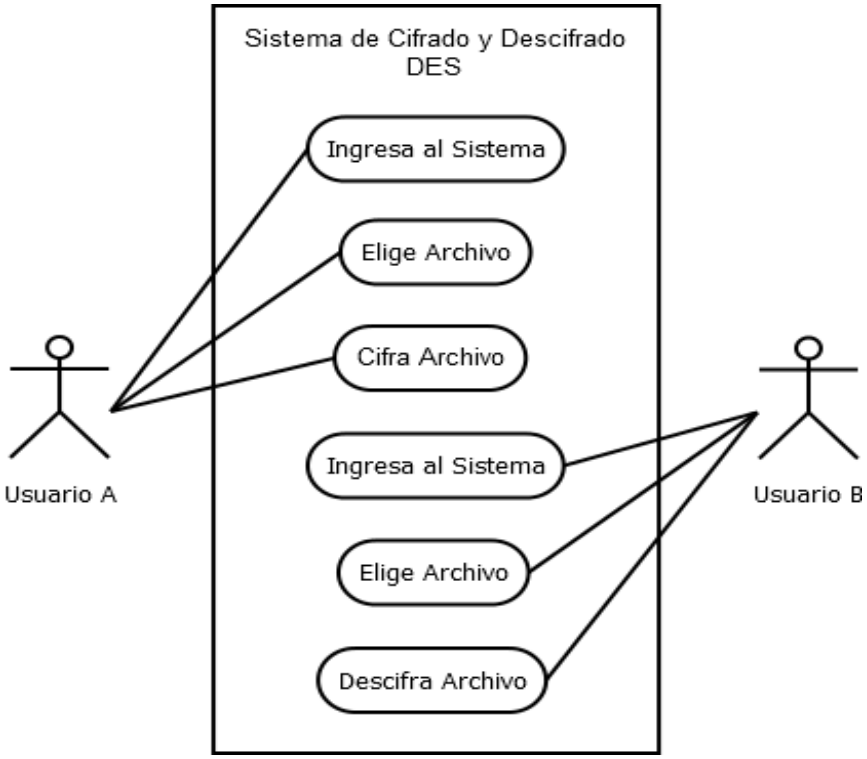


Figura 3.3 Diseño general del programa Triple DES. (Elaboración propia).

Las acciones que seguirá el sistema se descomponen en la siguiente Historia de Usuario (Tabla 3.23):

Tabla 3.23 Historia de usuario cifrado y descifrado de un archivo. (Elaboración propia).

Historia de Usuario	
Número: 1	**Usuario:** Usuario A y B
Nombre de la Historia: Cifrado y Descifrado de un archivo	
Prioridad en Negocio: alta	**Riesgo en Desarrollo:** alto
Puntos Estimados: 3	**Iteración Asignada:** 1
Programador Responsable: David Filio y Rolando Flores	
Descripción: Se requiere un programa que realice los pasos necesarios para seleccionar de una computadora cualquier archivo sin importar el tipo después cifrarlo usando el algoritmo Triple DES y que imprima en consola el tiempo que tarda en realizar el proceso de cifrado. Se requiere que el mismo programa realice los pasos necesarios para seleccionar de una computadora un archivo cifrado y usando el algoritmo Triple DES lo descifre y que imprima en consola el tiempo que tarda en realizar el proceso de descifrado.	
Observaciones: Utilizar el siguiente grupo de Keys: key1=133457799bbcdff1, key2=fedcba9876543210, key3=9876542310fedcba	

Para cumplir con lo solicitado en las Historias de Usuario estas se convierten en las siguientes dos Tarjetas de Ingeniería (Tablas 3.24 y 3.25):

Tabla 3.24 Tarjeta de Ingeniería Cifrado y Descifrado de archivos. (Elaboración propia).

		Tarea de Ingeniería
Número de Tarea: 1	**Historia de Usuario (No. y Nombre):** 1 Cifrado y Descifrado de un archivo.	
Nombre de la Tarea: Cifrado y Descifrado de Archivos		
Prioridad en Negocio: Alta		**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo		**Puntos Estimados:** 3
Fecha Inicio: 22/05/2014		**Fecha Fin:** 23/05/2014
Programador Responsable: David Filio y Rolando Flores		
Descripción: Elaborar una clase que repita tres veces el proceso de cifrado y descifrado del algoritmo DES utilizando para ello tres keys proporcionadas por los usuarios.		

Tabla 3.25 Tarjeta de Ingeniería Interfaz de Usuario. (Elaboración propia).

		Tarea de Ingeniería
Número de Tarea: 2	**Historia de Usuario (No. y Nombre):** 1 Cifrado y Descifrado de un archivo.	
Nombre de la Tarea: Interfaz de Usuario		
Prioridad en Negocio: Alta		**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo		**Puntos Estimados:** 3
Fecha Inicio: 26/05/2014		**Fecha Fin:** 26/05/2014
Programador Responsable: David Filio y Rolando Flores		
Descripción: Elaborar una clase que funcione como interfaz de usuario la cual mediante presionar un botón abra un explorador de archivos y permita a los usuarios seleccionar un archivo para cifrarlo y mediante la presión de otro botón permita a los usuarios seleccionar un archivo para descifrarlo.		

Compromiso de entrega. Se considera que los programadores puedan ejecutar los requisitos de las Historias de Usuario en una iteración y tres días.

Diseño

La Figura 3.4 muestra el prototipo del diseño del programa Triple DES, en base a los requerimientos de las Historias de Usuario.

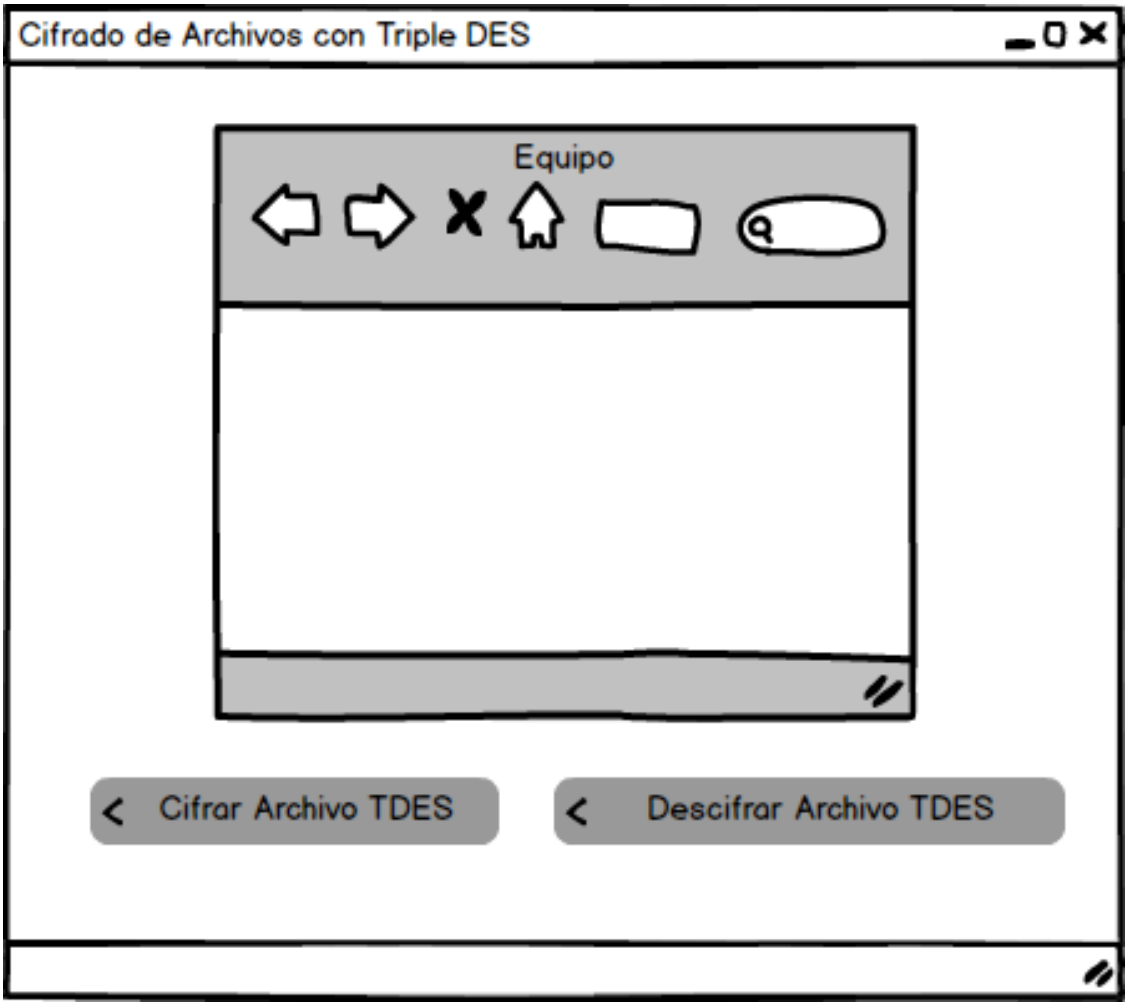


Figura 3.4 Prototipo del Programa de Cifrado Triple DES. (Elaboración propia).

Del Prototipo del programa de cifrado Triple DES, se desprenden las siguientes dos Tarjetas CRC (Tablas 3.26 y 3.27) las cuales posteriormente serán implementadas en código:

Tabla 3.26 Tarjeta CRC Clase Triple DES. (Elaboración propia).

Clase	
Triple DES	
Responsabilidades	**Colaboración**
private long K1[] private long K2[] private long K3[] static final int[] IP static final int[] IP_1 static final int[] E static final int[][] S static final int[] P static final int[] PC1 static final int[] LeftShifts static final int[] PC2 public void setKey(long key1,long key2,long key3) public byte[] cypher(byte txt[]) public byte[] decypher(byte txt[]) private int f(int data,int key, int ki) private String binary(long x, int bits) private String hex(byte x)	public class principal extends java.applet.Applet

Tabla 3.27 Tarjeta CRC Interfaz de Usuario. (Elaboración propia).

Clase	
Interfaz	
Responsabilidades	**Colaboración**
private JFileChooser fc public void init() private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)	public class TripleDES

Planeación y diseño del algoritmo de cifrado Triple DES-96

Planeación

Las acciones que seguirá el sistema se dividen en la siguiente Historia de Usuario (Tabla 3.28):

Tabla 3.28 Historia de usuario para el programa Triple DES-96. (Elaboración propia).

Historia de Usuario	
Número: 1	**Usuario:** Usuario A y B
Nombre de la Historia: Programa Triple DES-96	
Prioridad en Negocio: alta	**Riesgo en Desarrollo:** alto
Puntos Estimados: 3	**Iteración Asignada:** 1
Programador Responsable: David Filio y Rolando Flores	
Descripción: Se requiere que a partir del programa Triple DES se realice un programa que reúna las siguientes características de optimización: El bloque de entrada sea de 96 bits. En lugar de utilizar una permutación fija utilice una variable que para este caso se construya a partir de $PV = l\ x\ e$. Utilizar en la decimoquinta ronda del tercer ciclo PV^{-1}. Combinar las tablas de permutación E y P.	
Observaciones:	

Para cumplir con los requisitos de la Historia de Usuario esta se convierte en la siguiente Tarjeta de Ingeniería (Tabla 3.29):

Tabla 3.29 Tarjeta de Ingeniería para Triple DES-96. (Elaboración propia).

Tarea de Ingeniería	
Número de Tarea: 1	**Historia de Usuario (No. y Nombre):** 1 Programa Triple DES-96.
Nombre de la Tarea: Programa Triple DES-96	
Prioridad en Negocio: Alta	**Riesgo en Desarrollo:** Alta
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 27/05/2014	**Fecha Fin:** 28/05/2014
Programador Responsable: David Filio y Rolando Flores	
Descripción: Modificar el programa Triple DES para que permita la entrada de información en bloques de 96 bits en lugar de 64. Concatenar las Keys de entrada y multiplicarlas por el cálculo de los primeros valores del numero trascendente e después de su punto decimal, para obtener la PV. Realizar un ciclo for que cuando la información en proceso llegue a la decimoquinta ronda del tercer ciclo de cifrado la pase por permutación PV^{-1}. Realizar las operaciones necesarias para combinar los valores de las permutaciones E y P. Y generar una nueva permutación.	

Compromiso de entrega. Se considera que los programadores puedan ejecutar los requisitos de las Historias de Usuario en una iteración y dos días.

Diseño

La Figura 3.5 muestra el prototipo del diseño del programa Triple DES-96, en base a los requerimientos de las Historias de Usuario.

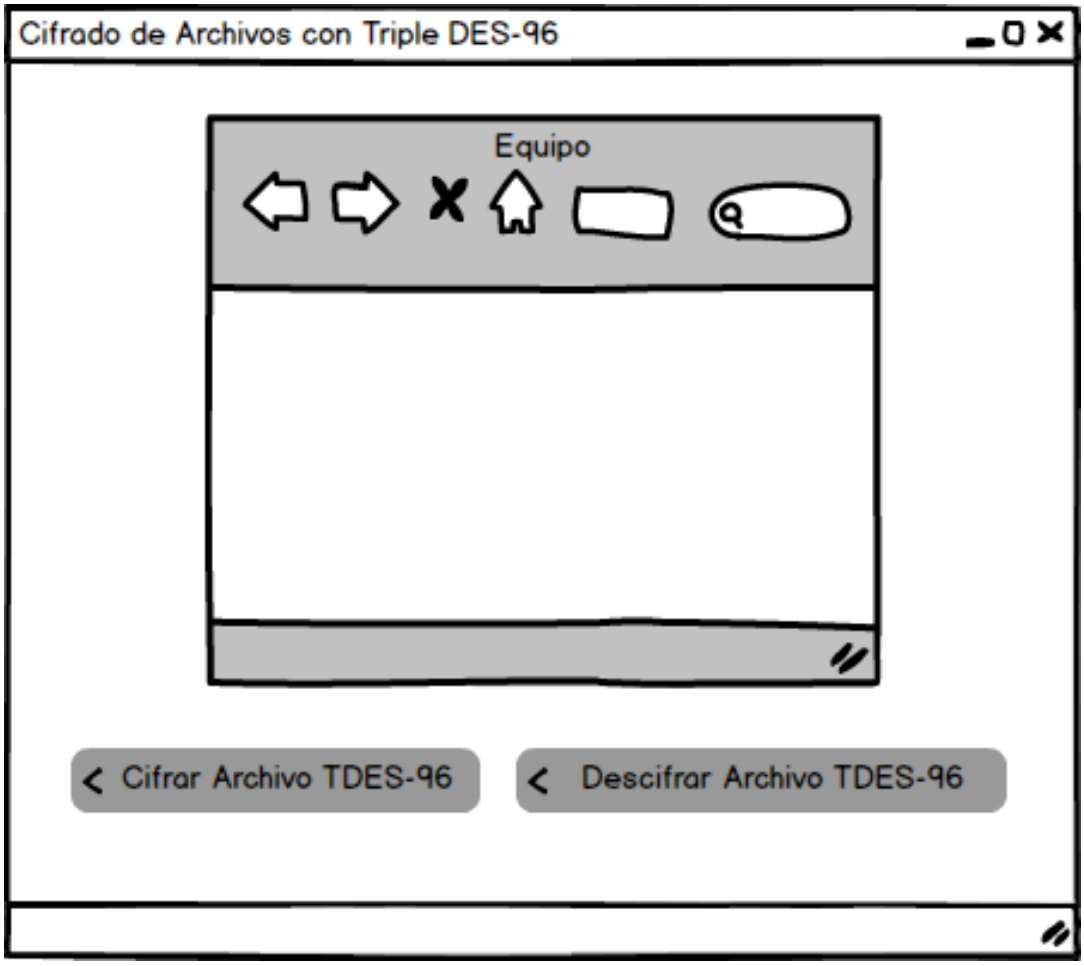


Figura 3.5 Prototipo del programa de Cifrado Triple DES-96. (Elaboración propia).

Del Prototipo del programa de cifrado Triple DES-96, se desprenden las siguientes dos Tarjetas CRC (Tablas 3.30 y 3.31) las cuales posteriormente serán implementadas en código:

Tabla 3.30 Tarjeta CRC Interfaz de Usuario. (Elaboración propia).

Clase	
Interfaz	
Responsabilidades	**Colaboración**
private JFileChooser fc public void init() private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) private void jButton4ActionPerformed(java.awt.event.ActionEvent evt)	public class TripleDES

Tabla 3.31 Tarjeta CRC Clase Triple DES-96. (Elaboración propia).

Clase	
Triple DES-96	
Responsabilidades	**Colaboración**
private long K1[] private long K2[] private long K3[] private int IP[] private int IP_1[] final int[] euler final int[][] S final int[] E final int[] PC1 final int[] LeftShifts final int[] PC2 private void getPermutation() public void setKey(long key1,long key2,long key3) public byte[] cypher(byte txt[]) public byte[] decypher(byte txt[]) public String hex(long x)	public class principal extends java.applet.Applet

CAPÍTULO IV: DESARROLLO DE LOS ALGORITMOS

Después de hacer la revisión teórica, la planeación y el diseño, este capítulo muestra la construcción de los algoritmos en programas de computadora resultantes de la aplicación de las metodologías de investigación anteriormente mencionadas. Todo cimentado en los valores, principios y elementos de Programación Extrema. Las etapas de construcción son:

1. Desarrollo del algoritmo de cifrado DES.
2. Desarrollo del algoritmo de cifrado Triple DES.
3. Desarrollo del algoritmo de cifrado Triple DES-96.
4. Optimización de las propuestas.
5. Codificación en lenguaje de programación.

Desarrollo del algoritmo de cifrado DES

Como ya se menciona en el capítulo II, el Data Encryption Standard (DES) es un es un algoritmo de criptografía simétrica aprobado como estándar internacional. En esta tesis se desarrolla ese algoritmo en lenguaje de programación Java, estrictamente en base a la norma publicada en (FIPS PUB 46-3, 1999) a continuación se describe su construcción.

Codificación

Para este apartado se utiliza la Programación Orientada a Objetos (POO). En este tipo de programación se trata de resolver un problema mediante componentes de código que son abstracciones a imagen y semejanza de la organización de los objetos animados e inanimados del mundo real. Una abstracción es una representación parcial de los atributos y comportamiento de un objeto real. Los atributos son las características, comportamiento y funcionalidad del objeto y el método es el comportamiento del objeto (Lopez, 2006) y parte de la información contenida en cada una de las tarjetas que puede ser codificada y después

implementada, se busca que la codificación sea lo más claro y conciso posible. Para este trabajo se utilizaron los estándares de programación que se muestran en la Tabla 4.1.

Tabla 4.1 Estándares de programación utilizados. (Elaboración propia).

Identación
Comentarios de cabecera
Comentarios de implementación
Comentarios de documentación
Cometarios de bloque
Programación en parejas
Propiedad colectiva de código

El código fuente completo de este programa se puede observar en el Anexo D: Código para el Cifrado-Descifrado de texto con DES.

Pruebas

De acuerdo a la metodología XP no se puede saber si algo funciona si no se prueba.

Prueba 1 Resultados Generales.

Criterio de prueba de aceptación. Para comprobar que el programa está bien desarrollado se comparan los resultados obtenidos con los publicados por (Stinson, 2006) en su libro (Cryptography Theory and Practice) capítulo 3.2.1 “An example of DES Encryption”, páginas 79 a la 81. Conforme a los datos de entrada (expresados en hexadecimal) que se muestran en la Figura 4.1.

3.2.1 An example of DES Encryption

Here is an example of encryption using the **DES**. Suppose we encrypt the (hexadecimal) plaintext

0123456789ABCDEF

Using the (hexadecimal) key

133457799BBCDFF1

The key, in binary, without parity-check bits, is

0001001001101001010110111100100110110111011011111111000

Applying **IP**, we obtain L_0 and R_0 (in binary):

$$L_0 = \mathbf{11001100000000001100110011111111}$$
$$L_1 = R_0 = \mathbf{11110000101010101111000010101010}$$

Figura 4.1 Datos de entrada para el programa DES. (Adaptado de Stinson, 2006).

Resultado esperado. La Figura 4.2 muestra los resultados obtenidos después completar el proceso de cifrado DES según (Stinson, 2006), con el texto plano "123456789ABCDEF" como entrada y "133457799BBCDFF1" usada como llave.

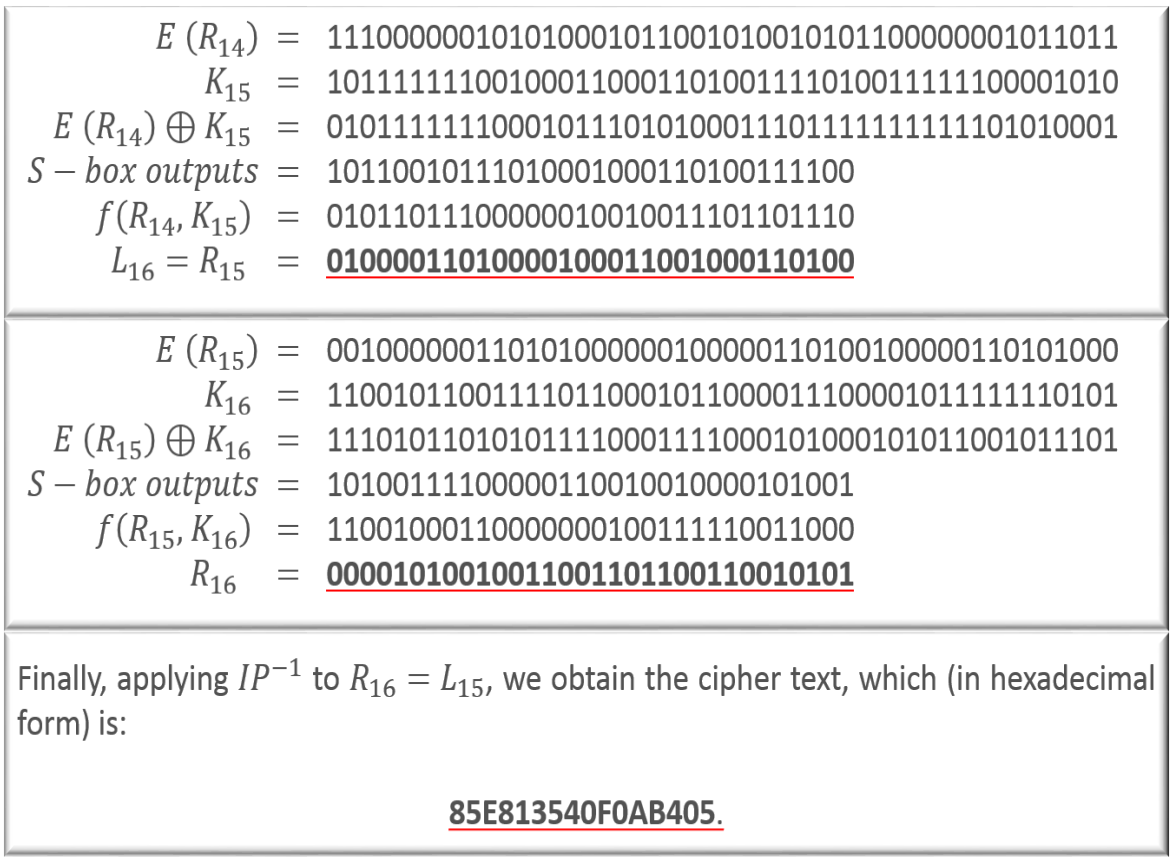


Figura 4.2 Resultado de cifrado según Stinson. (Adaptado de Stinson, 2006).

Resultado obtenido. La Figura 4.3 muestra los resultados obtenidos después completar el proceso de cifrado DES y se aprovecha para mostrar los resultados del descifrado del mismo algoritmo del programa de computadora desarrollado, ambos relacionados con las Historia de Usuario 1 y 2.

Evaluación de la prueba. Prueba satisfactoria.

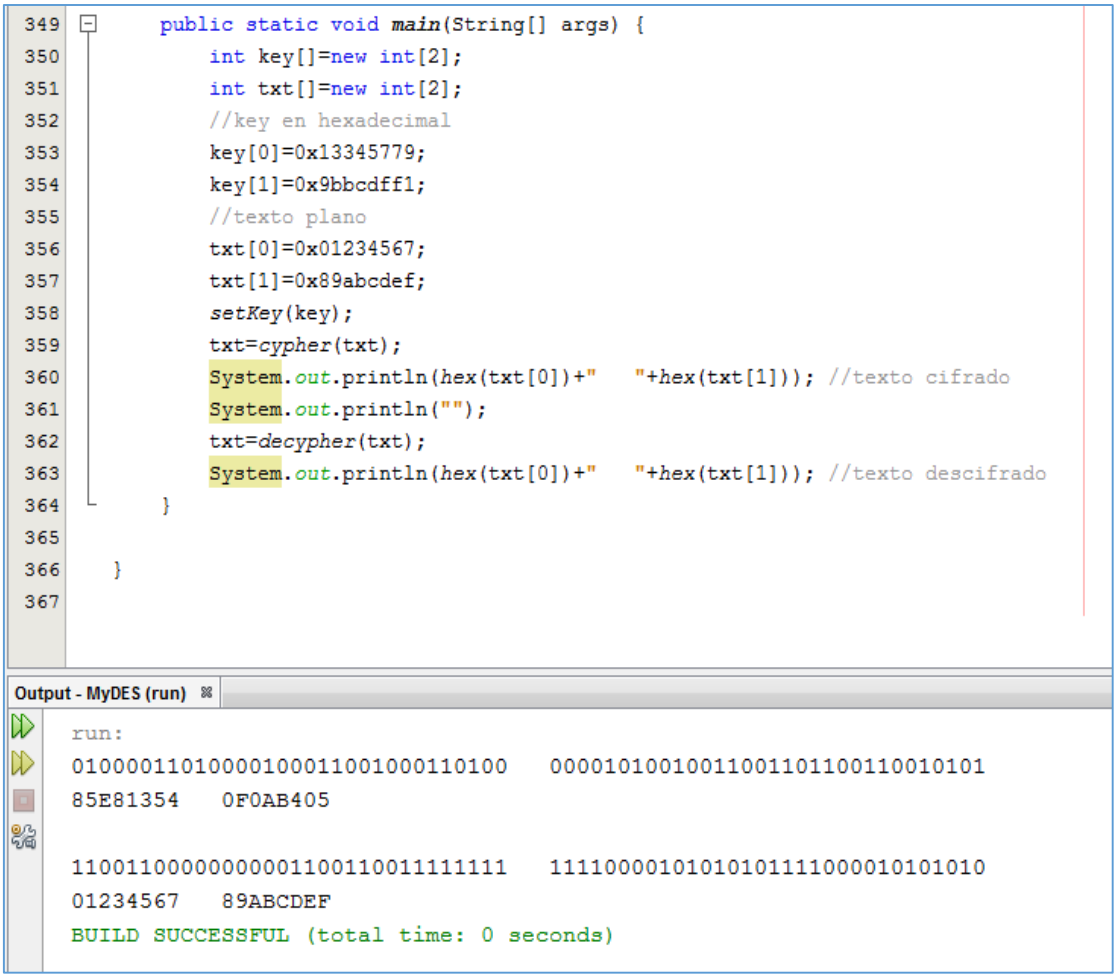


Figura 4.3 Resultados de la programación del algoritmo de cifrado DES. (Elaboración propia).

Prueba 2 Cifrado.

Descripción. De acuerdo con la figura 4.3 en la primer línea impresa, la primer cadena de bits corresponde a: $L_{16} = R_{15}$ de la Figura 4.2, y la segunda es idéntica al resultado de: R_{16} también de la Figura 4.2 (en hexadecimal el resultado es; 85E813540F0AB405).

Resultado de cifrado esperado. 85E813540F0AB405

Evaluación de la prueba. Prueba exitosa.

Prueba 3 Descifrado.

Prueba de Descifrado. De acuerdo con la figura 4.3 en la segunda línea impresa, la primera cadena de bits corresponde: a L_0 de la Figura 4.1. La segunda cadena de la figura 4.3 equivale a: $L_1 = R_0$ de la Figura 4.1 (en hexadecimal el resultado es; 0123456789ABCDEF).

Resultado de Descifrado esperado. 0123456789ABCDEF (texto plano).

Evaluación de la prueba. Prueba exitosa.

Desarrollo del algoritmo de cifrado Triple DES

Como ya se explicó en el capítulo II el algoritmo DES se repite tres veces de la forma Encrypt-Decrypt-Encrypt (E-D-E) y consta de tres llaves K_1, K_2 y K_3 y el resultado es un algoritmo llamado Triple DES.

Codificación

El código fuente completo, se presenta en el Anexo E: Código para el cifrado-descifrado de archivos con Triple DES. Y el código de la clase principal que hace ejecutable el programa se encuentra en el Anexo G: Código de la clase que ejecuta a Triple DES y a Triple DES-96.

Pruebas

Los siguientes puntos muestran las dos pruebas realizadas al programa:

Prueba 1 Selección de un Archivo.

Criterio de prueba de aceptación. Se requiere que la interfaz de usuario del programa después de presionar un botón muestre el explorador de archivos para permitir la selección de un documento que posteriormente será cifrado y/o descifrado.

Resultado esperado. Selección de un archivo mediante el explorador de archivos.

Resultado obtenido. La Figura 4.4 muestra el despliegue del explorador de archivos después de presionar el botón “cifrar” y la selección de un archivo.

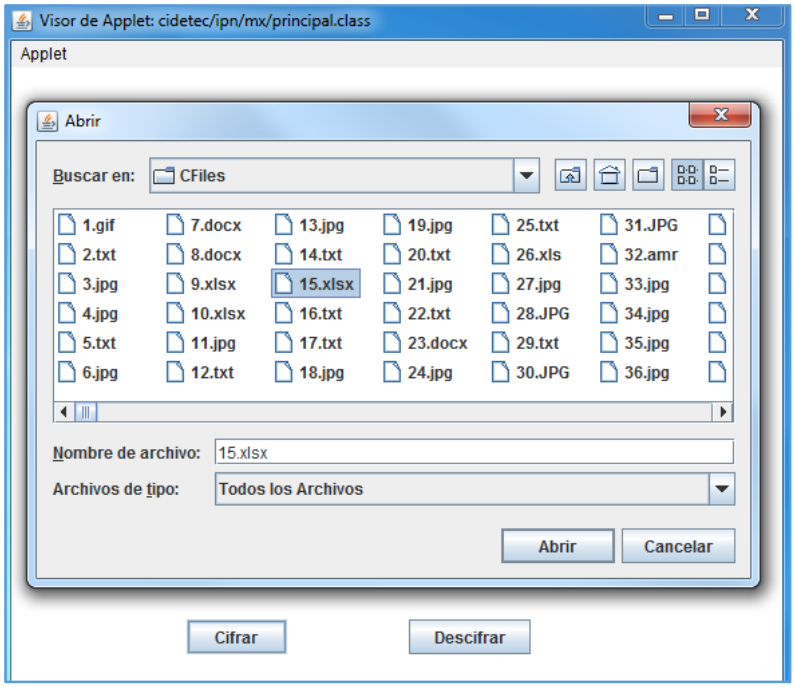


Figura 4.4 Interfaz de usuario. (Elaboración propia).

Evaluación de la prueba. Prueba satisfactoria.

Prueba 2 Cifrado y Descifrado de un archivo.

Criterio de prueba de aceptación. Las pruebas de este programa parten de las pruebas realizadas para el programa de cifrado DES y para este apartado se requiere el cifrado y posteriormente el descifrado de un mismo archivo y la impresión en consola del tiempo de ejecución.

Resultado esperado. Mostrar el archivo original, el archivo cifrado y después el archivo descifrado.

Resultado obtenido. La Figura 4.5 muestra el archivo original de nombre “50”, el archivo 50.jpg.enc que es el archivo cifrado al cual se le añadió intencionalmente mediante código la extensión .enc para diferenciarlo, cabe aclarar que aunque manualmente se le cambie la extensión por la original el archivo es ilegible ya que internamente se han modificado la posición de los bits que lo componen. Por último se muestra el archivo 50d que es el archivo descifrado y al cual manualmente se le añadió la letra d para que pudieran mostrarse los tres archivos en la misma carpeta.

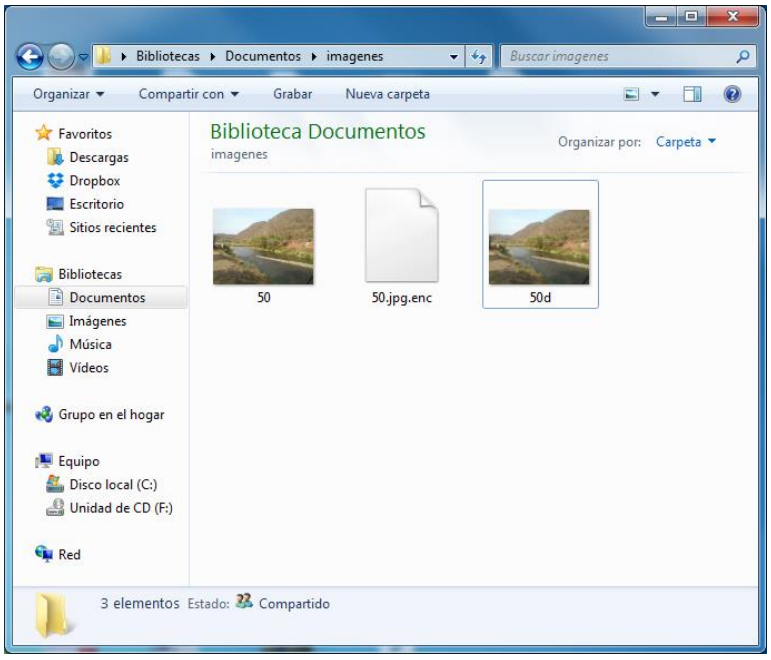


Figura 4.5 Archivos; original, cifrado y descifrado con Triple DES. (Elaboración propia).

La figura 4.6 es una impresión en consola del tiempo que tardo en cifrar el archivo de nombre 50 medido en milisegundos y también muestra la ruta donde se guardó el archivo resultante.

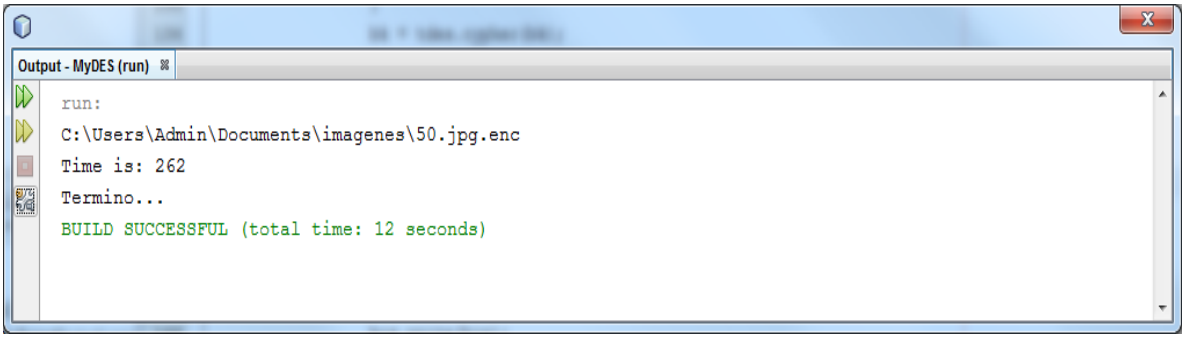


Figura 4.6 Tiempo de ejecución Triple DES. (Elaboración propia).

Evaluación de la prueba. Prueba exitosa.

Desarrollo del algoritmo de cifrado Triple DES-96

En este apartado a partir del programa Triple DES desarrollado en el punto anterior se realiza un Rediseño, que es el proceso mediante el cual se cambia un sistema de tal forma que no altere el comportamiento exterior del código fuente pero sí se mejore la estructura interna.

Triple-DES-96

A diferencia de Triple DES que inicia con la permutación fija de un bloque de 64 bits, Triple-DES-96 inicia con un bloque de 96 bits seguido de aplicar una permutación variable PV al comienzo del primer ciclo en la tercer ronda y su inversa PV^{-1} es aplicada en el comienzo del ciclo de la decimoquinta ronda, a continuación se explica el funcionamiento de Triple DES-96:

1. La cadena de entrada $Input$ es de 96 bits, la cual se divide en dos bloques (Izquierdo y Derecho) cada uno de 48 bits L_i, R_i, a R_i se le aplica la operación XOR $Input = R_i \oplus k_i$ donde el valor de k_i es de 48 bits.
2. $Input$ con valor de 48 bits pasa a ser la entrada de las $S\ Boxes$ y la salida de 32 bits recibirá el nombre de C.
3. El paso siguiente es aplicar la permutación P a C, y el resultado es una cadena de 32 bits, sin embargo es necesario una cadena de 48 bits para

realizar el siguiente paso de acuerdo con el estándar (FIPS PUB 46-3, 1999) que es la operación XOR con L_i, es por ello que en este paso se utiliza la permutación E y se combina con la permutación P con la finalidad de obtener los 48 bits necesarios y obtenemos $f_{96}(R_{i,}k_i) = E(C)$, este resultado se puede observar en la Tabla 4.2.

Tabla 4.2 Tabla de permutación resultante de combinar E y P. (Elaboración propia).

25	16	7	20	21	29
21	29	12	28	17	1
17	1	15	23	26	5
26	5	18	31	10	2
10	2	8	24	14	32
14	32	27	3	9	19
9	19	13	30	6	22
6	22	11	4	25	16

4. Para su funcionamiento Triple DES-96 utiliza tres ciclos, el primero es un DES-96, Flores, et al. (2013), con la inclusión de la permutación variable PV, en específico para este trabajo la PV se genera a partir del cálculo de las primeras cifras después del punto decimal del numero trascendente e (ver Tabla 4.3) esta se ocupa en la entrada de la tercer ronda.
5. El segundo ciclo es un algoritmo es de descifrado DES-96.
6. El tercer ciclo es de cifrado DES-96, pero en la entrada de la decimoquinta ronda se incluye la PV^{-1} es decir se aplica en la cadena L_{14} y R_{14}.
7. Triple DES-96 también utilizan tres llaves.

Tabla 4.3 Permutación resultante PV. (Elaboración propia).

77	69	35	8	2	41	75	63	4	70	7	59
82	92	22	16	33	31	47	72	80	6	5	12
40	79	24	36	87	51	17	93	29	81	21	55
57	66	58	74	10	49	1	83	88	44	32	94
90	43	64	85	62	86	78	34	27	15	61	28
84	60	54	20	42	95	68	9	96	19	14	37
18	76	73	25	11	56	65	3	13	52	67	38
71	39	46	45	91	30	50	48	23	26	53	89

Codificación

El código fuente completo, se presenta en el Anexo F: Código para el Cifrado-Descifrado de archivos con Triple DES-96. Y el código de la clase principal que hace ejecutable este programa se incluye en el mismo que cifra con Triple DES y se puede apreciar en el Anexo G: Código de la clase que ejecuta a Triple DES y a Triple DES-96.

Pruebas

Prueba 1 Cifrado y Descifrado de un Archivo.

Criterio de prueba de aceptación. Para este apartado se requiere el cifrado de un archivo y posteriormente el descifrado del mismo y la impresión en consola del tiempo de ejecución además considerando las modificaciones en el rediseño de código debe trabajar más rápido comparado con Triple DES.

Resultado esperado. Mostrar el archivo original, el archivo cifrado y después el archivo descifrado y la impresión en consola del tiempo de ejecución el cual debe ser menor al de Triple DES.

Resultado obtenido. La Figura 4.7 muestra el mismo archivo original utilizado para cifrar con Triple DES de nombre “50”, el archivo “50.jpg.enc” que es el archivo cifrado. Y por último se muestra el archivo “50e” que es el archivo descifrado.

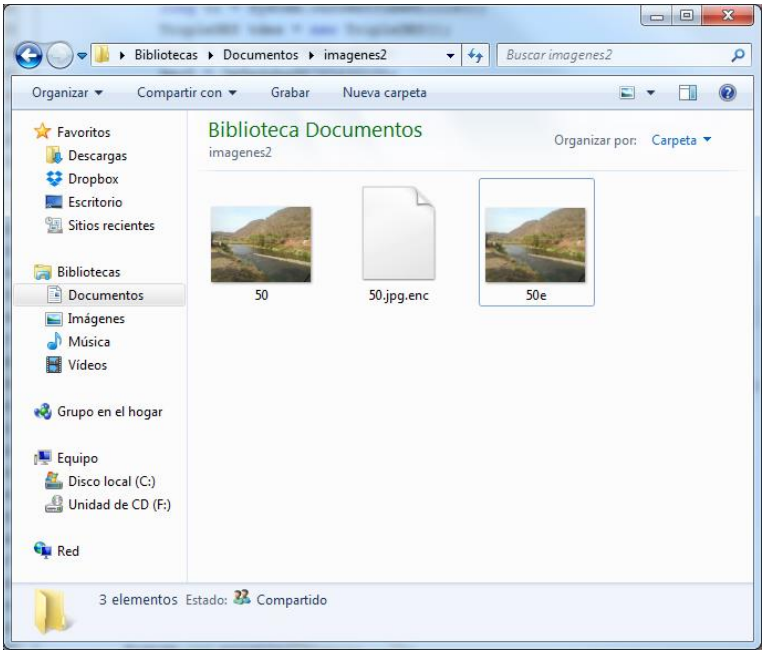


Figura 4.7 Archivos; original, cifrado y descifrado con Triple DES-96. (Elaboración propia).

La figura 4.8 es una impresión en consola del tiempo que tardo en cifrar el archivo de nombre “50” medido en milisegundos y también muestra la ruta donde se guardó el archivo resultante.

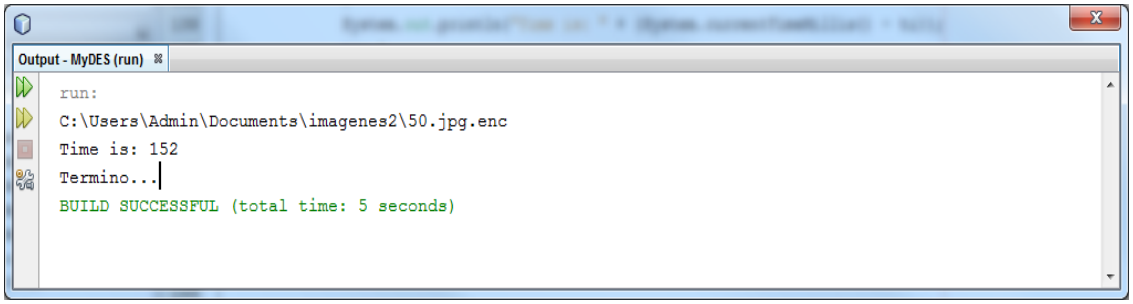


Figura 4.8 Tiempo de ejecución Triple DES-96. (Elaboración propia).

Evaluación de la prueba. Prueba exitosa.

CAPÍTULO V: RESULTADOS

En este capítulo se presentan los resultados obtenidos, según los procedimientos mencionados en el capítulo IV, haciendo uso de los siguientes recursos:

- Una computadora con Sistema Operativo Windows SP1, con procesador intel Core 2 Duo a 2.99 GHz y 4 Gb de memoria RAM.
- El desarrollo se realiza con apego al estándar (FIPS PUB 46-3, 1999) publicado por el National Institute of Standards and Technology (NIST).
- El entorno de desarrollo integrado (IDE) utilizado fue Netbeans.
- Lenguaje de programación Java versión 7, update 60.
- 1000 archivos de diferentes tipos y tamaños que van desde un kb y hasta 254 Mg para realizar las pruebas sobre ellos.
- Microsoft Excel para almacenar los resultados.
- En ambos casos se utiliza el siguiente grupo de llaves: K_1 = 133457799bbcdff1, K_2 = fedcba9876543210 y K_3 = 9876542310fedcba.

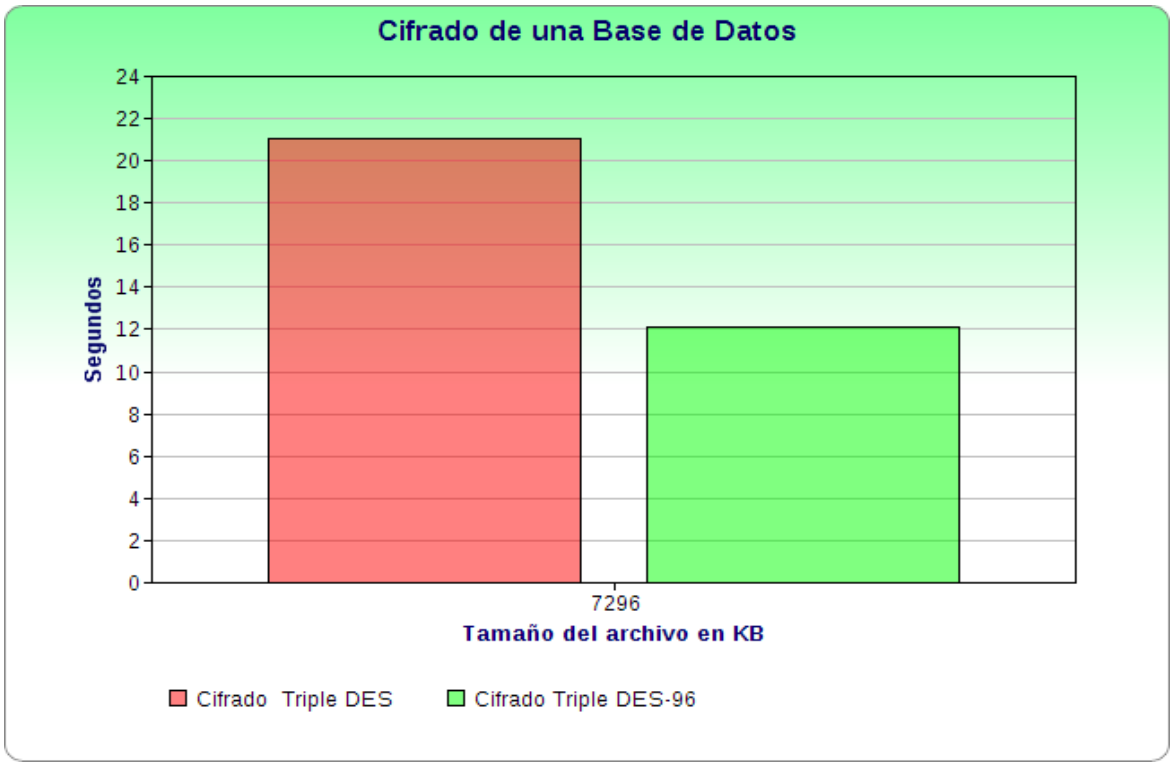
Las pruebas se ejecutaron aplicando de forma individual los algoritmos Triple DES y triple DES-96 sobre cada uno de los 1000 archivos seleccionados de una computadora, se presenta un reporte de casos en los cuales el tiempo de ejecución es expresado en segundos, cabe señalar que ese es el tiempo que tarda en ejecutarse cada algoritmo en un equipo con las características antes señaladas. A continuación se muestran algunos casos específicos para observar el comportamiento de los sistemas:

Caso uno

Como primer caso se presenta el tiempo de ejecución que tardan ambos programas en cifrar una base de datos con las características descritas en la Tabla 5.1 y se muestra su comportamiento en la Gráfica 5.1.

Tabla 5.1 Características de la Base de datos cifrada. (Elaboración propia).

No.	Tipo de Archivo	Tamaño en KB	Cifrado Triple DES (segundos)	Cifrado Triple DES-96 (segundos)
1	Base de Datos Access (.accdb)	7296	21.013	12.09



Gráfica 5.1 Comportamiento de la Base de datos cifrada. (Elaboración propia).

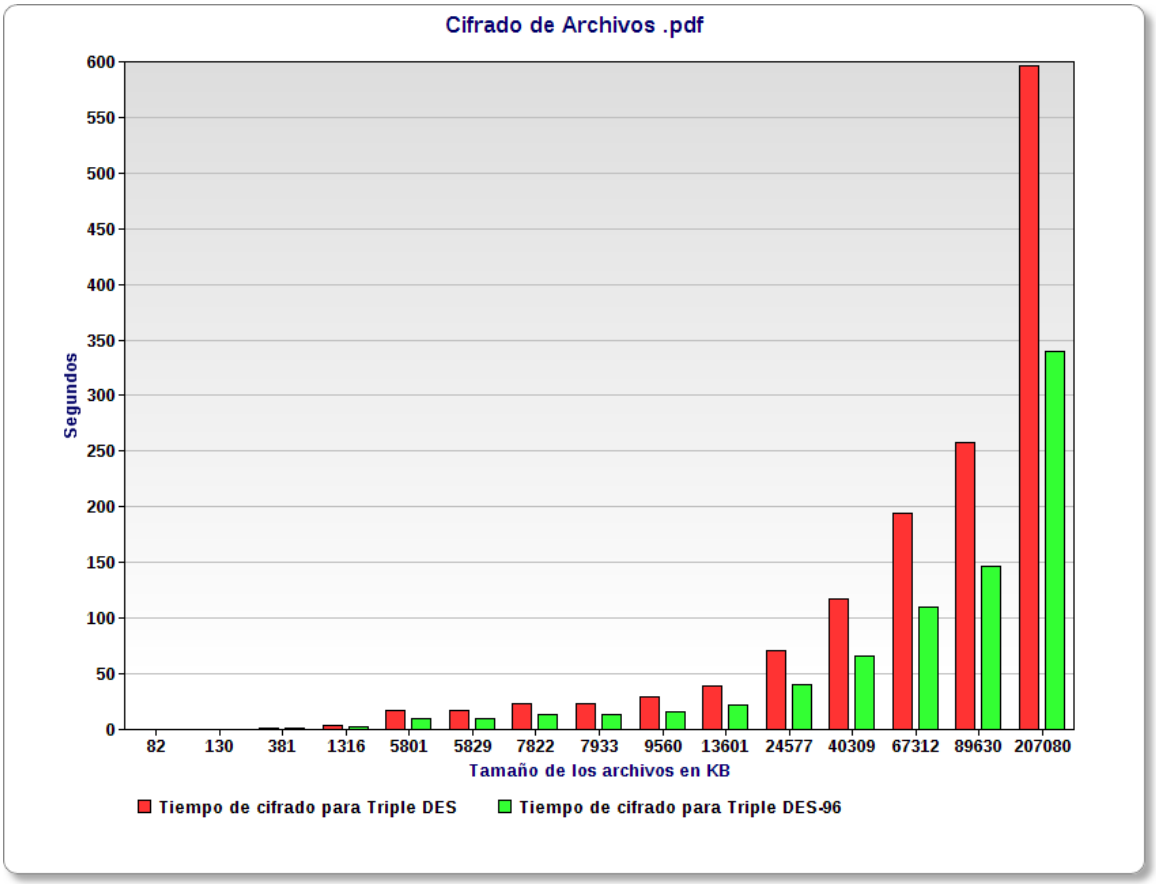
Al analizar la gráfica de la figura 5.1 Se puede ver que Triple DES-96 se tardó hasta 8.923 milisegundos menos que Triple DES.

Caso dos

En seguida se presentan 15 archivos en formato .pdf entre ellos (manuales, tutoriales, revistas y libros) con las características mostradas en la Tabla 5.2. Se mide el tiempo que tarda en cifrar cada algoritmo programado y su comportamiento se aprecia en la Gráfica 5.2.

Tabla 5.2 Características de los archivos en formato .pdf. (Elaboración propia).

No.	Tipo de Archivo	Tamaño en KB	Cifrado Triple DES (segundos)	Cifrado Triple DES-96 (segundos)
1	Adobe Acrobat Document (.pdf)	82	0.234	0.125
2	Adobe Acrobat Document (.pdf)	130	0.375	0.218
3	Adobe Acrobat Document (.pdf)	381	1.107	0.624
4	Adobe Acrobat Document (.pdf)	1316	3.791	2.153
5	Adobe Acrobat Document (.pdf)	5801	16.77	9.563
6	Adobe Acrobat Document (.pdf)	5829	17.363	9.89
7	Adobe Acrobat Document (.pdf)	7822	23.478	13.01
8	Adobe Acrobat Document (.pdf)	7933	23.353	13.026
9	Adobe Acrobat Document (.pdf)	9560	29.063	15.912
10	Adobe Acrobat Document (.pdf)	13601	39.14	22.604
11	Adobe Acrobat Document (.pdf)	24577	71.105	40.794
12	Adobe Acrobat Document (.pdf)	40309	116.876	66.113
13	Adobe Acrobat Document (.pdf)	67312	194.06	110.277
14	Adobe Acrobat Document (.pdf)	89630	257.884	146.843
15	Adobe Acrobat Document (.pdf)	207080	596.623	339.644



Gráfica 5.2 Tiempo de cifrado de los archivos pdf. (Elaboración propia).

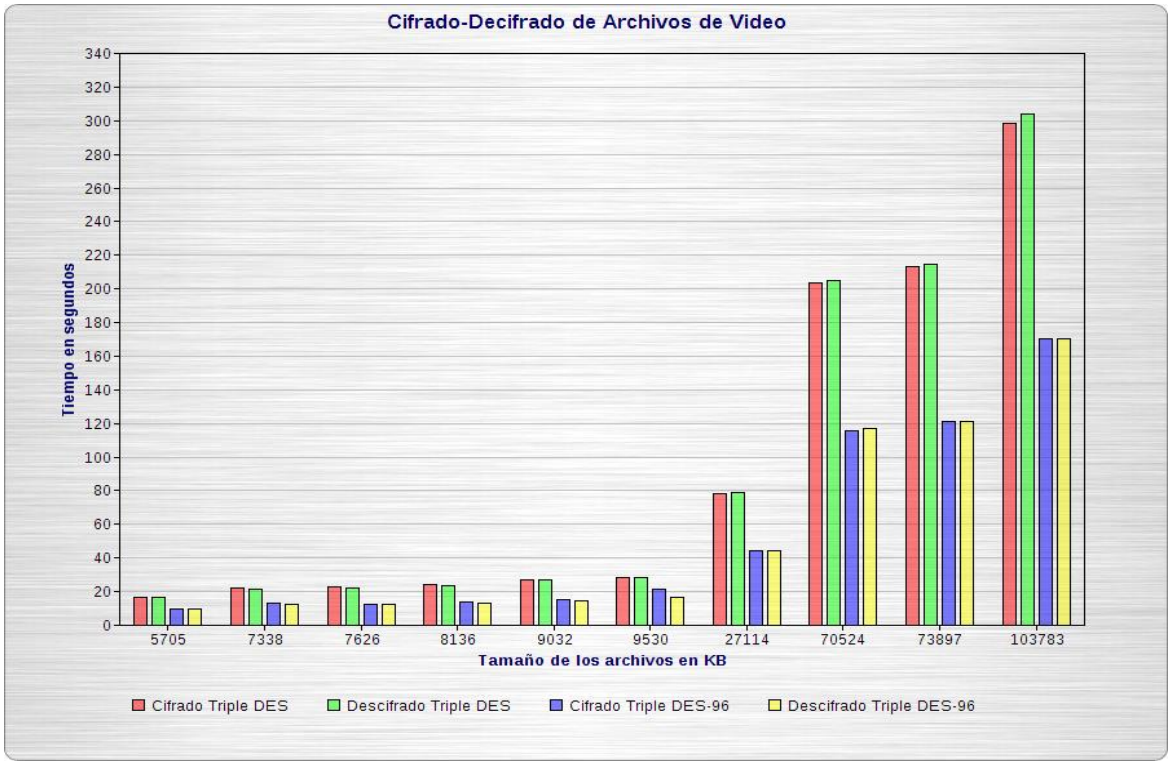
En la gráfica 5.2 se puede observar claramente que el tiempo que le tomó a Triple DES-96 para cifrar los archivos es menor que el que le tomó a Triple DES.

Caso tres

Los siguientes resultados se aplicaron sobre diez archivos de video con las características mostradas en la Tabla 5.3 también se muestran los tiempos que tardo cada algoritmo para cifrar y para descifrar:

Tabla 5.3 Características de los archivos de video. (Elaboración propia).

No.	Tipo de Archivo	Tamaño en KB	Cifrado Triple DES (segundos)	Descifrado Triple DES (segundos)	Cifrado Triple DES-96 (segundos)	Descifrado Triple DES-96 (segundos)
1	Vídeo MP4	5705	16.474	16.552	9.657	9.376
2	Vídeo MP4	7338	21.856	21.622	13.167	12.293
3	Vídeo MP4	7626	23.151	22.183	12.729	12.698
4	Vídeo MP4	8136	24.461	23.712	13.635	13.416
5	Vídeo MP4	9032	26.817	26.785	15.038	14.851
6	Vídeo MP4	9530	28.642	28.174	21.809	16.302
7	Windows Media (.avi)	27114	78.483	78.765	44.647	44.647
8	Windows Media (.avi)	70524	203.378	205.063	115.565	116.844
9	Windows Media (.avi)	73897	212.956	214.719	121.088	121.103
10	Windows Media (.avi)	103783	298.507	303.826	170.54	170.306



Gráfica 5.3 Tiempo de cifrado de los archivos de video. (Elaboración propia).

Tal como se observa en la Gráfica 5.3 el comportamiento es muy semejante al de los casos anteriores Triple DES-96 realiza el cifrado y descifrado en menos tiempo.

La tabla con los resultados de cifrado y descifrado que se obtuvieron de los 1000 archivos se incluye en el la Tabla 5.4 en el Anexo A: Resultado de cifrado y descifrado sobre 1000 archivos.

Propuesta de modificación

Con base en los resultados anteriores es posible saber que modificando la suite de cifrado SSL/TLS Record Protocol para que trabaje con Triple DES-96 el cifrado y descifrado de la información puede ser hasta 2.08 veces más rápido que cuando se trabaja con Triple DES, lo que resulta de dicha modificación se puede observar en la figura 5.1.

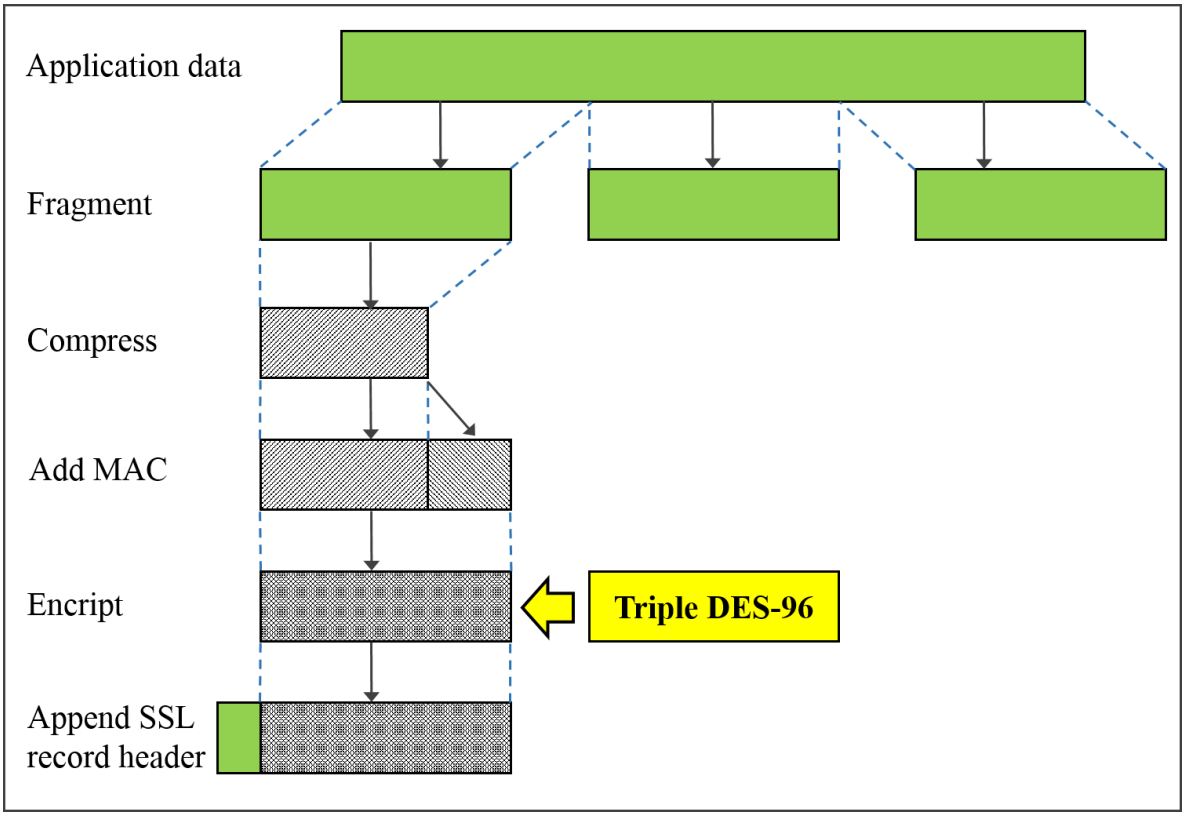


Figura 5.1 Propuesta de inclusión. (Elaboración propia).

Publicaciones

Los aspectos más relevantes de la presente investigación se redactaron en manuscrito científico y fueron enviados a la revista *Contemporary Engineering Sciences*, de la editorial HIKARI Ltd, con sede en Bulgaria, el mencionado artículo logró su aceptación y publicación en el volumen 7, número 20, paginas 947-956. Actualmente se encuentra disponible en su formato electrónico en la siguiente dirección: http://dx.doi.org/10.12988/ces.2014.48115

Ver Anexo B: Artículo publicado.

CONCLUSIONES Y TRABAJO FUTURO

Con la realización de este trabajo de investigación se comprueba que:

- ✓ Al ingresar bloques de información de 96 bits en lugar de 64 se logra procesar más información en menos tiempo sin perder confiabilidad.
- ✓ Al mover la tabla de permutación E a la salida de las Sboxes es posible conbinarla con la tabla P considerando que en Triple DES este proceso se realiza en cada una de las 16 rondas que se repiten en tres ciclos entonces Triple DES-96 se ahorran 48 procesos de permutación esto se refleja en un incremento de la velocidad para cifrar o descifrar información.
- ✓ Triple DES-96 utiliza una tabla de permutación variable PV lo que significa que los valores para esa permutación se pueden cambiar, en consecuencia el resultado obtenido es completamente diferente al que genera Triple DES.

Los objetivos planteados inicialmente en esta investigación se cumplieron, por tanto, los resultados fueron:

- ✓ Desarrollo de todos los sistemas planeados.
- ✓ Realización de modificaciones propuestas.
- ✓ Aplicación de pruebas de cifrado y descifrado de archivos con éxito.
- ✓ Desarrollo del algoritmo Triple DES-96 con base en la norma (FIPS PUB 46-3, 1999) conservando la fortaleza, complejidad y seguridad del estándar.

El principal aporte generado en la investigación es un algoritmo de cifrado seguro y confiable que genera resultados diferentes con lo que actualmente se maneja, conjuntamente, trabaja a mayor velocidad sin perder robustez, muy al contrario haciéndolo más potente.

Los resultados de las pruebas proyectan que mediante la modificación de la suite de cifrado SSL/TLS Record Protocol para que trabaje con Triple DES-96, el cifrado y descifrado de la información puede ser hasta 2,08 veces más rápido que cuando se utiliza Triple DES.

Concluido este trabajo de investigación se propone el siguiente listado de trabajo a futuro:

- Obtener un algoritmo de cifrado más robusto cambiando la longitud del grupo de Keys (K) que se utilizó en este trabajo y realizar las mismas pruebas.

- Utilizar el código que se generó y optimizarlo desarrollándolo en otro lenguaje de programación por ejemplo como Python que simplifica mucho la programación con esto se puede obtener mayor velocidad de cifrado.

- Determinar que el algoritmo Triple DES-96 no tenga debilidades mediante la realización de pruebas de criptoanálisis, esto pude servir también para incrementar su seguridad y confirmar su mayor robustez.

- Implementar el programa Triple DES-96 en dispositivos externos que no dependan de la memoria de la computadora, como en una tarjeta de desarrollo FPGA que sería de utilidad por ejemplo para la autenticación de usuarios por medio de hardware.

REFERENCIAS

Astilla E, M. B. (19 de junio de 2009). DISEÑO DE UN ESQUEMA DE COMUNICACIÓN SEGURA UTILIZANDO COMO CRIPTOSISTEMA SIMÉTRICO A TRIPLE DES Y ASIMÉTRICO A RSA. México D.F.: Instituto Politécnico Nacional CIDETEC. Obtenido de http://tesis.ipn.mx:8080/xmlui/handle/123456789/5828

Baker, A. (1990). Transcendental Number Theory. 3-4. London: Cambridge University Press.

Balanzario, E. P. (2003). NUMEROS ALGEBRAICOS Y TRASCENDENTES. 6-9. Morelia, Michoacán, México: Instituto de Matemáticas, UNAM-Morelia.

Beck, K. (2000). Extreme Programming Explained: Embrace Change. Reading, Mass: Addison-Wesley Professional.

Cohn, M. (2004). User Stories Applied: For Agile Software Development (Vol. 1). Boston: Addison-Wesley Professional.

Cortés L, C. J. (2007). CRIPTOANALISIS PARA TRIPE DES UTILIZANDO EL TEOREMA LR. Tesis de Maestría. México, D.F.: Instituto Politécnico Nacional CIDETEC.

Darthje. (2012). Curso Básico de Criptografía clásica. Obtenido de KRIPTÓPOLIS Criptografía y Seguridad: http://www.kriptopolis.com/criptografia-clasica-i

Davila M, J., & Luna R, E. (2000). Seguridad y Comercio por Internet. Conciencia Tecnológica.

Deitel, P. J., & Deitel, H. M. (2008). Java Como programar (Séptima ed.). México D.F.: Pearson Education.

Dierks, T., & Rescorla, E. (Agosto de 2008). The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246. Standard Tracks, Network Working Group.

Durán D, R., Hernández E, L., & Muñoz M, J. (2000). Ataques a DES y módulos factorizados de RSA. Madrid, España: DIGITAL.CSIC Ciencia en Abierto.

FIPS PUB 46-3. (1999). Data Encryption Standard (DES). U.S.: DEPARTMENT OF COMMERCE/National Institute of Standards and Technology.

Flores, R., Silva, V., & Rentería, C. (2013). DES BLOCK OF 96 BITS: AN APPLICATION TO IMAGE ENCRYPTION. International Journal of Research and Reviews in Applied Sciences, 14, Issue 1.

Foundation Electronic Frontier. (1998). Cracking DES: Secrets of Encryption Research, Wiretap Politics and Chip Design. Sebastopol, CA, USA: O'Reilly & Associates, Inc.

Freier, A., Karlton, P., & Kocher, P. (Agosto de 2011). The Secure Sockets Layer (SSL) Protocol Version 3.0. Internet Engineering Task Force, Netscape Communications.

Granados P, G. (2006). Introduccion a la Criptografía. Revista Digital Universitaria, 7(7). Obtenido de Revista Digital Universitaria: http://www.revista.unam.mx/vol.7/num7/art55/art55-1.htm#a

Hayoz, M., & Ultes-Nitsche, U. (16 de junio de 2003). Introducing SSL The Secure Sockets Layer Protocol. 13. Ue, Switzerland: MSc Seminar in Telecommunications.

IBM.(2014) Knowledge Center. Visión general del reconocimiento SSL Obtenido de: http://www01.ibm.com/support/knowledgecenter/SSFKSJ_7.0.1/com.ibm.mq.csqzas.doc/sy10660_.htm

Jaworski, J., & Perrone, P. (2001). Edición Especial Seguridad en Java. Madrid: Prentice Hall.

Kahn, D. (1974). The Codebreakers. New York: Macmillan USA; Edición: Reissue.

López G, E. (2010). ESQUEMA DE ENCRIPTACIÓN DE IMÁGENES USANDO 3 - DES DE PERMUTACIÓN VARIABLE. Tesis de Maestría. D.F., México: CIDETEC IPN.

Lopez R, L. (2006). Metodologia de la Programacion Orientada a Objetos. México, D.F.: Alfaomega.

Maiorano, A. (2009). Criptografía técnicas de desarrollo para profesionales. México: Alfaomega.

Microsoft.(31 de julio de 2003). SSL/TLS in Windows Server 2003. Obtenido de Overview of SSL/TLS Encryption: http://technet.microsoft.com/en-us/library/cc781476(v=ws.10).aspx

Muñoz M, A. (2011). INTRODUCCIÓN AL PROTOCOLO SSL Lección 9. Obtenido de Intypedia Information Security Encyclopedia: http://intypedia.com/

Open Systems Interconnection. (1994). ISO/IEC 7498-1: Basic Reference Model: The Basic Model. International Standard.

Pressman, R. S. (2010). Ingeniería de Software Un enfoque práctico. México: McGraw-Hill Interamericana Editores S.A. de C.V.

Ramió A, J. (Marzo de 2006). Libro Electrónico de Seguridad Informática y Criptografía Versión 4.1. Madrid, España. Obtenido de http://www.criptored.upm.es/guiateoria/gt_m001a.htm

Ramírez L, D. O., & Espinosa M, C. C. (2011). El Cifrado Web (SSL/TLS). .Seguridad Cultura de Prevención Para TI(10).

Rescorla, E. (2001). SSL and TLS: designing and building secure systems. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Rosales G, A. (Marzo de 2010). Números Trascendentes: Desarrollo Histórico. 10(2). Bahía de Almería, España: Revista Virtual Matemática, Educación e Internet.

Schneier, B. (1995). Applied Cryptography (2Nd Ed.): Protocols, Algorithms, and Source Code in C. New York, NY, USA: John Wiley & Sons, Inc.

Silva G, V. M., Flores C, R., Rentería M, C., & Luna B, B. (2013). The Triple-Des-96 Cryptographic System. 8. doi:10.12988/ijcms.2013.39104

Sommerville, I. (2011). Ingeniería de Software (9 ed.). México: Pearson Educación.

Stallings, W. (2004). Fundamentos de seguridad en redes: aplicaciones y estándares (2 ed.). Madrid: Pearson Educación.

Stallings, W. (2010). Cryptography and Network Security Principles and Practice (5 ed.). Boston: Prentice Hall.

Stinson, D. R. (2006). Cryptography Theory and Practice (3 ed.). Ontario, Canada: Chapman & Hall/CRC.

Tracy, M., Jansen, W., Scarfone, K., & Winograd, T. (september de 2007). Guidelines on Securing Public Web Servers, Special Publication 800-44 Version 2. Recommendations of the National Institute of Standards and Technology, 7-3 7-6.

Tuchman, W. (10 de diciembre de 2012). Hellman presents no shortcut solutions to DES. 16, 7, 40 - 41. (I. Spectrum, Ed.) doi:10.1109/MSPEC.1979.6368160

Wells, D. (1999). Unit Tests. Recuperado el 26 de 08 de 2014, de Extreme Programming: http://www.extremeprogramming.org/rules/unittests.html

Zotero.(2014). Obtenido de Zotero: https://www.zotero.org/david_filio/items

GLOSARIO

AES	Advanced Encryption Standard.
Algoritmo	Conjunto finito de instrucciones o pasos que sirven para ejecutar una tarea o resolver un problema.
ANSI	American National Standards Institute.
Ataque de fuerza bruta	Ataque que requiere intentar todos los valores posibles hasta que se encuentre el valor correcto.
Autenticación	Proceso para verificar la identidad de un usuario, dispositivo u otra entidad en un sistema informático.
Bit	Digito binario cuyos posibles valores son 1 y 0.
Bloque	Secuencia de bits de una longitud determinada.
Box	(Caja) Es una tabla de sustitución utilizada para representar una función matemática con propiedades deseadas, como la no linealidad.
Byte	Conjunto de 8 bits.
Certificado Digital	Es el documento digital que emite una autoridad de certificación para garantizar la identidad de las partes que intervienen en una comunicación.
Cifrado	Procedimiento reversible por el cual se transforma información original en información ilegible.
Cifrado de flujo	Es un sistema criptográfico que cifra de bit en bit.
Cifrado por Bloques	Es un sistema criptográfico que cifra de bloque en bloque.
Delito Informático	Toda conducta ilícita susceptible de ser sancionada por el Derecho penal.
DES	Data Encryption Standard.
Descifrado	Proceso inverso del cifrado.
Diffie-Hellman	Protocolo criptográfico desarrollado por Whitfield Diffie y Martin Hellman.
FIPS	Federal information Processing Standards.
Firma Digital	Proceso que garantiza la identidad de las partes involucradas en un proceso o transacción electrónica.
Fortezza	Algoritmo criptográfico desarrollado por el gobierno de Estados Unidos.
IDEA	Internacional Data Encryption Algorithm.
IEEE	Institute of Electrical and Electrinics Engineers.
IP	Internet Protocol.
ISO	International Standards Organization.
K	Llave o clave.

MD5	Message Digest v5.
Método	Procedimiento que se sigue para decir o hacer algo con orden.
Netscape	Empresa de informática con sede en Mountain View (California, Estados Unidos).
NIST	National Institute of Standards and Technology.
OSI	Es un modelo de referencia constituido por siete capas, que permite la interconexión de redes.
Padding	En criptografía es añadir caracteres de relleno hasta completar un formato o tamaño exigido.
Permutación	Sustitución en la que los bits de un bloque de entrada son reordenados para producir el bloque cifrado.
Plaint text	(Texto plano) Texto en su forma original.
RC2	Rivest Cipher v2.
RC4	Rivest Cipher v4.
Round	Un solo recorrido a través de un procedimiento, durante la ejecución de un sistema criptográfico.
RSA	(Rivest, Shamir y Adleman) Algoritmo de cifrado de clave pública que se basa en exponenciación de aritmética modular y su seguridad recae en el problema de factorización de números primos grandes.
SHA	Security Hash Algorithm.
SSL	Secure Socket Layer, protocolo criptográfico que proporciona comunicaciones seguras por una red.
Sustitución	Es el reemplazo de un valor de entrada por otro de los posibles valores de salida.
TCP/IP	Transmission Control Protocol, es un protocolo de conexión de redes.
Texto Cifrado	Forma protegida de un texto el cual es ilegible y es el resultado de aplicar un sistema criptográfico sobre él.
TLS	Transport Layer Security, sucesor de SSL y totalmente compatibles.
Triple DES	Resultado de aplicar tres veces el algoritmo de cifrado DES.
UDP	User Datagram Protocol, Es un protocolo del nivel de transporte basado en el intercambio de datagramas y se ubica en la capa 4 del Modelo OSI.
XOR	Operador lógico, que resultara en 1 si los valores son diferentes y en 0 si son iguales.

ANEXOS

Anexo A. Resultado de cifrado y descifrado sobre 1000 archivos.

Tabla 5.4 Resultado de Cifrado-Descifrado de 1000 archivos. (Elaboración propia).

No.	Tipo de Archivo	Tamaño en KB	Cifrado Triple DES	Descifrado Triple DES	Cifrado Triple DES-96	Descifrado Triple DES-96
1	Imagen GIF (.gif)	1	0	0	0	0
2	Documento de texto (.txt)	3	0.016	0.015	0.015	0.015
3	Imagen JPEG (.jpg)	5	0.016	0.016	0.016	0.016
4	Imagen JPEG (.jpg)	8	0.016	0.016	0.016	0.015
5	Documento de texto (.txt)	9	0.031	0.032	0.016	0.015
6	Imagen JPEG (.jpg)	11	0.031	0.031	0.015	0.015
7	Microsoft Word (.docx)	12	0.047	0.031	0.032	0.031
8	Microsoft Word (.docx)	13	0.031	0.047	0.031	0.032
9	Microsoft Excel (.xlsx)	14	0.047	0.031	0.016	0.031
10	Microsoft Excel (.xlsx)	15	0.031	0.047	0.015	0.031
11	Imagen JPEG (.jpg)	17	0.047	0.046	0.031	0.031
12	Documento de texto (.txt)	18	0.047	0.047	0.031	0.031
13	Imagen JPEG (.jpg)	20	0.062	0.062	0.031	0.031
14	Documento de texto (.txt)	21	0.062	0.063	0.031	0.032
15	Microsoft Excel (.xlsx)	24	0.062	0.062	0.047	0.031
16	Documento de texto (.txt)	25	0.078	0.062	0.047	0.047
17	Documento de texto (.txt)	27	0.078	0.078	0.031	0.031
18	Imagen JPEG (.jpg)	28	0.078	0.078	0.031	0.031
19	Imagen JPEG (.jpg)	29	0.094	0.093	0.047	0.047
20	Documento de texto (.txt)	30	0.094	0.094	0.047	0.047
21	Imagen JPEG (.jpg)	31	0.093	0.094	0.047	0.047
22	Documento de texto (.txt)	32	0.094	0.094	0.047	0.047
23	Microsoft Word (.docx)	33	0.094	0.093	0.048	0.047
24	Imagen JPEG (.jpg)	34	0.094	0.094	0.047	0.048
25	Documento de texto (.txt)	35	0.093	0.093	0.047	0.047
26	Microsoft Excel 97-2003 (.xlsx)	38	0.125	0.109	0.062	0.063
27	Imagen JPEG (.jpg)	39	0.109	0.125	0.062	0.062
28	Imagen JPEG (.jpg)	40	0.109	0.125	0.062	0.062
29	Documento de texto (.txt)	41	0.124	0.11	0.062	0.062
30	Imagen JPEG (.jpg)	43	0.124	0.125	0.078	0.078
31	Imagen JPEG (.jpg)	44	0.125	0.125	0.062	0.062
32	VLC media file (.amr)	46	0.14	0.14	0.078	0.078
33	Imagen JPEG (.jpg)	48	0.14	0.141	0.078	0.078
34	Imagen JPEG (.jpg)	49	0.14	0.14	0.078	0.078
35	Imagen JPEG (.jpg)	52	0.141	0.156	0.093	0.093
36	Imagen JPEG (.jpg)	54	0.156	0.156	0.094	0.094
37	Microsoft Excel 97-2003 (.xlsx)	55	0.156	0.156	0.093	0.094
38	Imagen JPEG (.jpg)	58	0.172	0.172	0.094	0.094
39	Microsoft Word (.docx)	60	0.172	0.187	0.109	0.109
40	Imagen JPEG (.jpg)	64	0.171	0.188	0.109	0.093
41	Imagen JPEG (.jpg)	66	0.187	0.187	0.109	0.11
42	Imagen JPEG (.jpg)	67	0.187	0.203	0.109	0.11
43	Microsoft Word (.docx)	70	0.203	0.202	0.125	0.109
44	Imagen JPEG (.jpg)	71	0.203	0.218	0.125	0.125
45	Imagen JPEG (.jpg)	73	0.203	0.218	0.124	0.125
46	Imagen JPEG (.jpg)	76	0.219	0.219	0.125	0.125
47	Imagen JPEG (.jpg)	77	0.218	0.219	0.125	0.125
48	Microsoft Word (.docx)	81	0.234	0.234	0.125	0.14
49	Adobe Acrobat Document (.pdf)	82	0.234	0.249	0.125	0.14
50	Imagen JPEG (.jpg)	90	0.25	0.265	0.156	0.14
51	Adobe Acrobat Document (.pdf)	92	0.265	0.265	0.156	0.14
52	Imagen JPEG (.jpg)	93	0.281	0.281	0.156	0.156
53	Imagen JPEG (.jpg)	94	0.281	0.281	0.156	0.156
54	Imagen JPEG (.jpg)	99	0.297	0.296	0.156	0.172
55	Imagen JPEG (.jpg)	102	0.297	0.296	0.156	0.172
56	Imagen JPEG (.jpg)	104	0.312	0.312	0.172	0.172
57	Microsoft Excel (.xlsx)	105	0.297	0.312	0.171	0.172
58	Imagen JPEG (.jpg)	106	0.328	0.312	0.171	0.172
59	Imagen JPEG (.jpg)	107	0.312	0.312	0.171	0.172
60	Microsoft Excel (.xlsx)	110	0.312	0.312	0.171	0.187
61	Imagen JPEG (.jpg)	111	0.328	0.343	0.187	0.202
62	Imagen JPEG (.jpg)	115	0.343	0.328	0.187	0.187
63	Adobe Acrobat Document (.pdf)	124	0.375	0.374	0.203	0.202
64	Imagen JPEG (.jpg)	129	0.39	0.374	0.203	0.219
65	Adobe Acrobat Document (.pdf)	130	0.375	0.39	0.218	0.218
66	Microsoft Excel (.xlsx)	136	0.405	0.39	0.218	0.234
67	Imagen JPEG (.jpg)	138	0.406	0.406	0.234	0.218
68	Imagen JPEG (.jpg)	147	0.437	0.436	0.234	0.234
69	Microsoft Excel (.xlsx)	148	0.452	0.452	0.249	0.249
70	Imagen JPEG (.jpg)	152	0.453	0.452	0.25	0.25
71	Microsoft Excel (.xlsx)	156	0.452	0.437	0.265	0.25

72	Imagen JPEG (.jpg)	168	0.484	0.5	0.28	0.265
73	Imagen JPEG (.jpg)	171	0.499	0.499	0.28	0.281
74	Microsoft Word (.docx)	182	0.53	0.531	0.297	0.296
75	Imagen JPEG (.jpg)	186	0.561	0.546	0.312	0.312
76	Imagen JPEG (.jpg)	188	0.562	0.546	0.328	0.312
77	Imagen JPEG (.jpg)	192	0.561	0.561	0.312	0.328
78	Imagen JPEG (.jpg)	197	0.562	0.578	0.328	0.328
79	Imagen JPEG (.jpg)	201	0.577	0.592	0.328	0.328
80	Imagen JPEG (.jpg)	222	0.671	0.639	0.375	0.374
81	Imagen JPEG (.jpg)	226	0.64	0.655	0.375	0.374
82	Microsoft Word (.docx)	228	0.655	0.671	0.374	0.374
83	Imagen JPEG (.jpg)	234	0.686	0.686	0.39	0.406
84	Microsoft Word (.docx)	237	0.717	0.702	0.39	0.39
85	Imagen JPEG (.jpg)	240	0.718	0.702	0.39	0.406
86	Imagen JPEG (.jpg)	251	0.734	0.717	0.422	0.421
87	Imagen JPEG (.jpg)	258	0.764	0.748	0.436	0.437
88	Imagen JPEG (.jpg)	263	0.764	0.749	0.437	0.437
89	Imagen JPEG (.jpg)	268	0.78	0.764	0.453	0.452
90	Microsoft Excel 97-2003 (.xlsx)	275	0.796	0.796	0.453	0.452
91	Microsoft Excel 97-2003 (.xlsx)	276	0.796	0.811	0.452	0.453
92	Microsoft Excel 97-2003 (.xlsx)	277	0.812	0.811	0.468	0.609
93	Imagen JPEG (.jpg)	279	0.811	0.811	0.468	0.468
94	Imagen JPEG (.jpg)	286	0.827	0.827	0.483	0.468
95	Imagen JPEG (.jpg)	289	0.843	0.843	0.484	0.484
96	Imagen JPEG (.jpg)	291	0.843	0.842	0.484	0.483
97	Archivo PNG (.png)	297	0.858	0.874	0.514	0.499
98	Imagen JPEG (.jpg)	301	0.873	0.873	0.515	0.514
99	Microsoft Excel 97-2003 (.xlsx)	309	0.89	0.905	0.515	0.515
100	Imagen JPEG (.jpg)	315	0.921	0.905	0.515	0.53
101	Imagen JPEG (.jpg)	322	0.936	0.92	0.546	0.531
102	Microsoft Excel 97-2003 (.xlsx)	336	0.967	0.967	0.562	0.546
103	Imagen JPEG (.jpg)	337	0.967	0.983	0.561	0.562
104	Imagen JPEG (.jpg)	344	0.999	1.014	0.578	0.562
105	Imagen JPEG (.jpg)	350	1.014	1.03	0.577	0.577
106	Imagen JPEG (.jpg)	360	1.045	1.045	0.592	0.592
107	Imagen JPEG (.jpg)	362	1.045	1.06	0.593	0.593
108	Imagen JPEG (.jpg)	365	1.046	1.061	0.608	0.608
109	Imagen JPEG (.jpg)	371	1.061	1.092	0.609	0.624
110	Imagen JPEG (.jpg)	377	1.092	1.092	0.624	0.624
111	Imagen JPEG (.jpg)	379	1.107	1.108	0.639	0.64
112	Adobe Acrobat Document (.pdf)	381	1.107	1.092	0.624	0.624
113	Imagen JPEG (.jpg)	386	1.107	1.123	0.64	0.639
114	Imagen JPEG (.jpg)	390	1.124	1.139	0.639	0.64
115	Imagen JPEG (.jpg)	392	1.139	1.139	0.655	0.64
116	Imagen JPEG (.jpg)	395	1.139	1.139	0.655	0.655
117	Imagen JPEG (.jpg)	400	1.154	1.154	0.655	0.671
118	Imagen JPEG (.jpg)	403	1.155	1.185	0.671	0.67
119	Microsoft Excel (.xlsx)	410	1.17	1.186	0.717	0.686
120	Imagen JPEG (.jpg)	417	1.217	1.467	0.702	0.686
121	Imagen JPEG (.jpg)	423	1.217	1.232	0.702	0.702
122	Archivo PNG (.png)	428	1.279	1.248	0.702	0.718
123	Archivo PNG (.png)	430	1.264	1.248	0.702	0.718
124	Imagen JPEG (.jpg)	435	1.248	1.248	0.733	0.717
125	Base de Datos Access (.accdb)	440	1.279	1.279	0.733	0.717
126	Imagen JPEG (.jpg)	443	1.279	1.295	0.749	0.749
127	Imagen JPEG (.jpg)	447	1.295	1.31	0.749	0.733
128	Archivo PNG (.png)	452	1.295	1.31	0.749	0.749
129	Archivo PNG (.png)	456	1.326	1.326	0.749	0.764
130	Imagen JPEG (.jpg)	460	1.326	1.326	0.764	0.795
131	Imagen JPEG (.jpg)	463	1.341	1.357	0.78	0.78
132	Imagen JPEG (.jpg)	466	1.358	1.357	0.765	0.78
133	Imagen JPEG (.jpg)	468	1.357	1.357	0.78	0.78
134	Imagen JPEG (.jpg)	470	1.357	1.373	0.78	0.78
135	Imagen JPEG (.jpg)	473	1.357	1.357	0.78	0.796
136	Imagen JPEG (.jpg)	476	1.373	1.373	0.78	0.796
137	Imagen JPEG (.jpg)	479	1.388	1.7	0.78	0.811
138	Imagen JPEG (.jpg)	482	1.388	1.701	0.796	0.796
139	Imagen JPEG (.jpg)	485	1.388	1.42	0.796	0.811
140	Imagen JPEG (.jpg)	488	1.42	1.419	0.811	0.811
141	Imagen JPEG (.jpg)	489	1.545	1.42	0.811	0.811
142	Imagen JPEG (.jpg)	491	1.513	1.436	0.811	0.827
143	Imagen JPEG (.jpg)	495	1.435	1.435	0.812	0.811
144	Imagen JPEG (.jpg)	497	1.435	1.513	0.827	0.827
145	Imagen JPEG (.jpg)	499	1.451	1.467	0.827	0.826
146	Imagen JPEG (.jpg)	501	1.435	1.451	0.827	0.874
147	Imagen JPEG (.jpg)	503	1.466	1.451	0.842	0.842
148	Imagen JPEG (.jpg)	507	1.467	1.482	0.827	0.843
149	Microsoft Excel 97-2003 (.xlsx)	510	1.468	1.482	0.858	0.842
150	Imagen JPEG (.jpg)	513	1.498	1.482	0.843	0.842
151	Imagen JPEG (.jpg)	515	1.482	1.528	0.858	0.873
152	Imagen JPEG (.jpg)	518	1.482	1.498	0.858	0.858
153	Imagen JPEG (.jpg)	520	1.513	1.498	0.874	0.858
154	Imagen JPEG (.jpg)	523	1.498	1.528	0.858	0.858
155	Imagen JPEG (.jpg)	524	1.514	1.513	0.874	0.858
156	Imagen JPEG (.jpg)	527	1.413	1.544	0.874	0.858
157	Imagen JPEG (.jpg)	530	1.438	1.545	0.874	0.874

158	Imagen JPEG (.jpg)	533	1.444	1.545	0.873	0.89
159	Imagen JPEG (.jpg)	535	1.444	1.544	0.874	0.889
160	Imagen JPEG (.jpg)	538	1.56	1.56	0.873	0.889
161	Imagen JPEG (.jpg)	541	1.591	1.576	0.89	0.89
162	Imagen JPEG (.jpg)	543	1.575	1.56	0.905	0.89
163	Imagen JPEG (.jpg)	546	1.575	1.575	0.905	0.905
164	Imagen JPEG (.jpg)	551	1.591	1.607	0.905	0.92
165	Microsoft Excel 97-2003 (.xlsx)	554	1.606	1.591	0.936	0.904
166	Imagen JPEG (.jpg)	557	1.607	1.607	0.92	0.921
167	Imagen JPEG (.jpg)	559	1.622	1.607	0.92	0.968
168	Imagen JPEG (.jpg)	561	1.638	1.638	0.92	0.967
169	Imagen JPEG (.jpg)	563	1.622	1.622	0.936	0.983
170	Imagen JPEG (.jpg)	567	1.638	1.638	0.936	1.014
171	Imagen JPEG (.jpg)	569	1.654	1.653	0.936	0.951
172	Imagen JPEG (.jpg)	572	1.638	1.669	0.936	0.936
173	Imagen JPEG (.jpg)	575	1.654	1.654	0.936	0.952
174	Microsoft Excel 97-2003 (.xlsx)	577	1.654	1.794	0.952	0.967
175	Imagen JPEG (.jpg)	580	1.669	1.669	0.951	0.952
176	Imagen JPEG (.jpg)	583	1.668	1.685	0.967	0.967
177	Imagen JPEG (.jpg)	585	1.669	1.685	0.967	0.967
178	Imagen JPEG (.jpg)	588	1.701	1.7	1.014	1.029
179	Imagen JPEG (.jpg)	591	1.716	1.716	0.983	0.967
180	Imagen JPEG (.jpg)	593	1.716	1.716	0.982	0.983
181	Imagen JPEG (.jpg)	596	1.716	1.732	0.982	0.983
182	Imagen JPEG (.jpg)	599	1.747	1.732	0.983	0.999
183	Imagen JPEG (.jpg)	602	1.732	1.732	0.983	0.998
184	Imagen JPEG (.jpg)	605	1.747	1.763	0.983	0.999
185	Imagen JPEG (.jpg)	608	1.747	1.762	1.014	0.998
186	Sonido en formato MP3	611	1.779	1.762	1.029	1.014
187	Microsoft Excel 97-2003 (.xlsx)	614	1.763	1.794	0.998	1.03
188	Sonido en formato MP3	625	1.81	1.81	1.029	1.029
189	Imagen JPEG (.jpg)	627	1.81	1.825	1.029	1.03
190	Imagen JPEG (.jpg)	630	1.825	1.841	1.029	1.03
191	Imagen JPEG (.jpg)	633	1.826	1.841	1.045	1.045
192	Imagen JPEG (.jpg)	636	1.825	1.872	1.045	1.046
193	Imagen JPEG (.jpg)	639	1.841	1.887	1.046	1.061
194	Imagen JPEG (.jpg)	642	1.857	1.856	1.061	1.061
195	Imagen JPEG (.jpg)	645	1.857	1.95	1.06	1.061
196	Imagen JPEG (.jpg)	648	1.856	1.887	1.077	1.061
197	Imagen JPEG (.jpg)	651	1.872	1.888	1.076	1.077
198	Imagen JPEG (.jpg)	654	1.872	1.887	1.076	1.077
199	Imagen JPEG (.jpg)	657	1.997	1.903	1.092	1.092
200	Imagen JPEG (.jpg)	659	1.919	1.903	1.092	1.092
201	Sonido en formato MP3	662	1.903	1.918	1.092	1.108
202	Imagen JPEG (.jpg)	665	1.919	1.918	1.092	1.108
203	Imagen JPEG (.jpg)	669	1.919	1.934	1.107	1.108
204	Imagen JPEG (.jpg)	672	1.934	1.95	1.107	1.108
205	Imagen JPEG (.jpg)	675	1.935	1.966	1.123	1.108
206	Imagen JPEG (.jpg)	679	1.966	1.965	1.123	1.123
207	Imagen JPEG (.jpg)	684	1.981	1.981	1.139	1.155
208	Imagen JPEG (.jpg)	689	1.997	1.981	1.139	1.139
209	Imagen JPEG (.jpg)	691	1.996	1.997	1.139	1.139
210	Imagen JPEG (.jpg)	695	2.013	2.012	1.139	1.139
211	Imagen JPEG (.jpg)	697	2.013	2.028	1.155	1.154
212	Imagen JPEG (.jpg)	700	2.012	2.012	1.155	1.154
213	Imagen JPEG (.jpg)	703	2.028	2.043	1.154	1.154
214	Imagen JPEG (.jpg)	708	2.044	2.044	1.17	1.185
215	Imagen JPEG (.jpg)	715	2.075	2.075	1.185	1.185
216	Imagen JPEG (.jpg)	721	2.091	2.09	1.186	1.186
217	Imagen JPEG (.jpg)	724	2.075	2.09	1.185	1.201
218	Imagen JPEG (.jpg)	730	2.106	2.121	1.202	1.201
219	Imagen JPEG (.jpg)	734	2.106	2.137	1.202	1.201
220	Imagen JPEG (.jpg)	739	2.964	2.138	1.232	1.217
221	Imagen JPEG (.jpg)	743	2.153	2.153	1.233	1.233
222	Imagen JPEG (.jpg)	748	2.168	2.168	1.232	1.233
223	Imagen JPEG (.jpg)	750	2.153	2.184	1.232	1.232
224	Imagen JPEG (.jpg)	755	2.215	2.184	1.248	1.248
225	Imagen JPEG (.jpg)	759	2.184	2.199	1.264	1.248
226	Imagen JPEG (.jpg)	763	2.199	2.215	1.264	1.264
227	Imagen JPEG (.jpg)	769	2.215	2.231	1.264	1.279
228	Imagen JPEG (.jpg)	773	2.215	2.246	1.263	1.279
229	Imagen JPEG (.jpg)	777	2.792	2.278	1.28	1.279
230	Imagen JPEG (.jpg)	781	2.246	2.262	1.295	1.295
231	Imagen JPEG (.jpg)	783	2.262	2.278	1.295	1.279
232	Imagen JPEG (.jpg)	785	2.262	2.308	1.279	1.31
233	Imagen JPEG (.jpg)	795	2.293	2.294	1.31	1.31
234	Imagen JPEG (.jpg)	801	2.309	2.309	1.326	1.31
235	Imagen JPEG (.jpg)	804	2.325	2.34	1.326	1.342
236	Sonido en formato MP3	807	2.34	2.34	1.451	1.342
237	Sonido en formato MP3	810	2.34	2.34	1.326	1.326
238	Sonido en formato MP3	820	2.372	2.372	1.357	1.357
239	Sonido en formato MP3	825	2.371	2.387	1.372	1.357
240	Imagen JPEG (.jpg)	829	2.387	2.403	1.373	1.373
241	Imagen JPEG (.jpg)	833	2.403	2.418	1.373	1.388
242	Imagen JPEG (.jpg)	838	2.418	2.434	1.373	1.388
243	Imagen JPEG (.jpg)	841	2.418	2.434	1.388	1.388

244	Imagen JPEG (.jpg)	844	2.433	2.433	1.388	1.389
245	Imagen JPEG (.jpg)	847	2.434	2.45	1.404	1.388
246	Imagen JPEG (.jpg)	851	2.449	2.449	1.404	1.404
247	Imagen JPEG (.jpg)	854	2.449	2.481	1.42	1.404
248	Imagen JPEG (.jpg)	860	2.496	2.527	1.419	1.42
249	Imagen JPEG (.jpg)	863	2.481	2.412	1.42	1.42
250	Imagen JPEG (.jpg)	868	2.527	2.496	1.435	1.42
251	Imagen JPEG (.jpg)	871	2.511	2.527	1.436	1.435
252	Imagen JPEG (.jpg)	874	2.527	2.527	1.451	1.435
253	Imagen JPEG (.jpg)	879	2.543	2.543	1.451	1.451
254	Imagen JPEG (.jpg)	882	2.543	2.574	1.45	1.466
255	Microsoft Word 97-2003	885	2.543	2.574	1.451	1.451
256	Imagen JPEG (.jpg)	888	2.574	2.574	1.467	1.466
257	Imagen JPEG (.jpg)	891	2.637	2.574	1.466	1.482
258	Imagen JPEG (.jpg)	896	2.589	2.574	1.467	1.482
259	Imagen JPEG (.jpg)	898	2.574	2.589	1.482	1.529
260	Microsoft Word 97-2003	901	2.605	2.605	1.514	1.513
261	Imagen JPEG (.jpg)	904	2.605	2.621	1.482	1.482
262	Imagen JPEG (.jpg)	909	2.621	2.62	1.498	1.497
263	Imagen JPEG (.jpg)	913	2.636	2.652	1.528	1.498
264	Imagen JPEG (.jpg)	919	2.668	2.652	1.513	1.514
265	Imagen JPEG (.jpg)	923	2.668	2.668	1.529	1.514
266	Sonido en formato MP3	925	2.652	2.699	1.513	1.513
267	Microsoft Excel (.xlsx)	927	2.668	2.699	1.529	1.529
268	Imagen JPEG (.jpg)	929	2.683	2.699	1.529	1.529
269	Imagen JPEG (.jpg)	931	2.699	2.699	1.544	1.529
270	Imagen JPEG (.jpg)	937	2.698	2.698	1.545	1.56
271	Sonido en formato MP3	945	2.73	2.73	1.544	1.56
272	Imagen JPEG (.jpg)	948	2.73	2.745	1.56	1.56
273	Imagen JPEG (.jpg)	955	2.73	2.777	1.576	1.576
274	Sonido en formato MP3	959	2.869	2.792	1.575	1.591
275	Sonido en formato MP3	961	2.761	2.777	1.576	1.591
276	Imagen JPEG (.jpg)	963	2.762	2.808	1.591	1.592
277	Imagen JPEG (.jpg)	966	2.777	2.808	1.575	1.592
278	Sonido en formato MP3	977	2.808	2.823	1.607	1.607
279	Imagen JPEG (.jpg)	980	2.839	2.824	1.622	1.622
280	Imagen JPEG (.jpg)	983	2.824	2.855	1.607	1.607
281	Imagen JPEG (.jpg)	987	2.839	2.855	1.622	1.638
282	Imagen JPEG (.jpg)	991	2.854	2.855	1.623	1.622
283	Imagen JPEG (.jpg)	993	2.871	2.886	1.685	1.7
284	Sonido en formato MP3	995	2.855	2.886	1.638	1.638
285	Sonido en formato MP3	999	2.87	2.886	1.653	1.638
286	Imagen JPEG (.jpg)	1001	2.901	2.902	1.638	1.653
287	Sonido en formato MP3	1005	2.902	2.901	1.638	1.654
288	Sonido en formato MP3	1008	2.886	2.901	1.654	1.653
289	Sonido en formato MP3	1010	2.902	2.917	1.653	1.654
290	Sonido en formato MP3	1023	2.948	2.949	1.669	1.685
291	Sonido en formato MP3	1034	2.98	2.996	1.7	1.7
292	Sonido en formato MP3	1055	3.042	3.042	1.747	1.731
293	Sonido en formato MP3	1068	3.089	3.073	1.763	1.762
294	Sonido en formato MP3	1072	3.074	3.089	1.762	1.763
295	Imagen JPEG (.jpg)	1086	3.136	3.151	1.794	1.778
296	Sonido en formato MP3	1096	3.167	3.167	1.81	1.809
297	Sonido en formato MP3	1120	3.26	3.276	1.856	1.84
298	Sonido en formato MP3	1124	3.23	3.261	1.841	1.856
299	Sonido en formato MP3	1131	3.245	3.26	1.857	1.856
300	Sonido en formato MP3	1132	3.246	3.26	1.857	1.872
301	Sonido en formato MP3	1149	3.307	3.322	1.888	1.919
302	Microsoft Word 97-2003	1151	3.322	3.338	1.919	1.934
303	Sonido en formato MP3	1155	3.323	3.369	1.919	1.903
304	Sonido en formato MP3	1166	3.37	3.369	1.935	1.919
305	Sonido en formato MP3	1170	3.354	3.417	1.919	1.919
306	Sonido en formato MP3	1181	3.401	3.416	1.934	1.966
307	Sonido en formato MP3	1188	3.416	3.447	1.95	1.95
308	Sonido en formato MP3	1190	3.447	3.432	1.997	1.95
309	Sonido en formato MP3	1205	3.463	3.479	1.997	1.981
310	Sonido en formato MP3	1206	3.463	3.494	1.997	1.981
311	Sonido en formato MP3	1213	3.495	3.51	2.074	1.997
312	Sonido en formato MP3	1233	3.541	3.573	2.028	2.028
313	Sonido en formato MP3	1246	3.588	3.619	2.043	2.044
314	Sonido en formato MP3	1248	3.588	3.604	2.06	2.059
315	Sonido en formato MP3	1249	3.588	3.604	2.059	2.06
316	Sonido en formato MP3	1260	3.619	3.65	2.091	2.074
317	Sonido en formato MP3	1264	3.635	3.666	2.075	2.075
318	Sonido en formato MP3	1274	3.666	3.682	2.09	2.106
319	Sonido en formato MP3	1275	3.681	3.713	2.091	2.09
320	Sonido en formato MP3	1277	3.682	3.713	2.09	2.121
321	Sonido en formato MP3	1283	3.697	3.713	2.106	2.106
322	Sonido en formato MP3	1286	3.729	3.729	2.106	2.122
323	Sonido en formato MP3	1299	3.744	3.775	2.137	2.153
324	Sonido en formato MP3	1303	3.76	3.776	2.153	2.153
325	Sonido en formato MP3	1306	3.776	3.775	2.137	2.153
326	Sonido en formato MP3	1309	3.76	3.775	2.153	2.153
327	Sonido en formato MP3	1310	3.76	3.79	2.153	2.153
328	Sonido en formato MP3	1315	3.79	3.806	2.169	2.168
329	Adobe Acrobat Document (.pdf)	1316	3.791	3.822	2.153	2.184

330	Sonido en formato MP3	1319	3.807	3.822	2.169	2.184
331	Sonido en formato MP3	1320	3.806	3.822	2.184	2.169
332	Sonido en formato MP3	1329	3.822	3.838	2.184	2.184
333	Sonido en formato MP3	1330	3.884	3.869	2.2	2.309
334	Sonido en formato MP3	1332	3.822	4.119	2.199	2.184
335	Sonido en formato MP3	1335	3.868	3.869	2.2	2.246
336	Sonido en formato MP3	1337	3.853	3.869	2.2	2.2
337	Sonido en formato MP3	1347	3.9	3.9	2.215	2.216
338	Sonido en formato MP3	1352	3.9	3.931	2.231	2.247
339	Sonido en formato MP3	1353	3.884	3.9	2.247	2.215
340	Sonido en formato MP3	1355	3.9	3.916	2.231	2.231
341	Sonido en formato MP3	1363	3.931	3.947	2.246	2.247
342	Sonido en formato MP3	1370	3.931	3.978	2.247	2.262
343	Sonido en formato MP3	1377	3.962	3.978	2.262	2.262
344	Sonido en formato MP3	1381	3.978	3.978	2.387	2.262
345	Sonido en formato MP3	1383	3.978	4.009	2.277	2.278
346	Sonido en formato MP3	1390	4.025	4.025	2.277	2.278
347	Sonido en formato MP3	1394	4.025	4.056	2.293	2.293
348	Sonido en formato MP3	1395	4.025	4.056	2.293	2.293
349	Sonido en formato MP3	1399	4.009	4.056	2.355	2.309
350	Sonido en formato MP3	1401	4.056	4.056	2.308	2.356
351	Sonido en formato MP3	1405	4.056	4.071	2.308	2.324
352	Sonido en formato MP3	1406	4.04	4.056	2.324	2.324
353	Sonido en formato MP3	1408	4.041	4.072	2.324	2.309
354	Sonido en formato MP3	1410	4.071	4.087	2.308	2.309
355	Sonido en formato MP3	1414	4.072	4.087	2.356	2.356
356	Sonido en formato MP3	1419	4.102	4.102	2.325	2.34
357	Sonido en formato MP3	1421	4.119	4.103	2.324	2.387
358	Sonido en formato MP3	1422	4.103	4.15	2.34	2.34
359	Sonido en formato MP3	1425	4.118	4.118	2.34	2.48
360	Sonido en formato MP3	1428	4.149	4.118	2.355	2.355
361	Sonido en formato MP3	1429	4.118	4.134	2.356	2.34
362	Sonido en formato MP3	1430	4.119	4.134	2.355	2.356
363	Sonido en formato MP3	1438	4.165	4.166	2.356	2.372
364	Sonido en formato MP3	1445	4.149	4.181	2.387	2.386
365	Sonido en formato MP3	1447	4.165	4.212	2.371	2.387
366	Sonido en formato MP3	1449	4.165	4.196	2.387	2.371
367	Sonido en formato MP3	1451	4.165	4.196	2.387	2.387
368	Sonido en formato MP3	1452	5.211	4.196	2.403	2.387
369	Sonido en formato MP3	1454	4.228	4.196	2.387	2.418
370	Sonido en formato MP3	1455	4.196	4.243	2.387	2.403
371	Sonido en formato MP3	1456	4.181	4.196	2.418	2.403
372	Sonido en formato MP3	1457	4.212	4.212	2.402	2.403
373	Sonido en formato MP3	1458	4.196	4.212	2.418	2.403
374	Sonido en formato MP3	1460	4.197	4.212	2.402	2.418
375	Sonido en formato MP3	1462	4.212	4.243	2.403	2.418
376	Sonido en formato MP3	1464	4.212	4.227	2.45	2.418
377	Sonido en formato MP3	1471	4.243	4.243	2.418	2.418
378	Sonido en formato MP3	1475	4.243	4.259	2.418	2.433
379	Sonido en formato MP3	1477	4.258	4.275	2.418	2.434
380	Sonido en formato MP3	1479	4.259	4.275	2.449	2.434
381	Sonido en formato MP3	1482	4.275	4.29	2.449	2.433
382	Sonido en formato MP3	1483	4.259	4.306	2.433	2.434
383	Sonido en formato MP3	1484	4.321	4.29	2.449	2.434
384	Sonido en formato MP3	1488	4.29	4.29	2.449	2.465
385	Sonido en formato MP3	1492	6.006	4.306	2.464	2.465
386	Sonido en formato MP3	1494	4.29	4.321	2.465	2.496
387	Sonido en formato MP3	1496	4.305	4.321	2.48	2.465
388	Sonido en formato MP3	1500	4.353	4.337	2.465	2.465
389	Sonido en formato MP3	1501	4.321	4.368	2.464	2.465
390	Sonido en formato MP3	1502	4.321	4.399	2.465	2.48
391	Sonido en formato MP3	1503	4.337	4.352	2.481	2.48
392	Sonido en formato MP3	1507	4.352	4.352	2.481	2.48
393	Sonido en formato MP3	1510	4.368	4.399	2.48	2.496
394	Sonido en formato MP3	1511	4.352	4.384	2.496	2.48
395	Sonido en formato MP3	1512	4.352	4.384	2.496	2.48
396	Sonido en formato MP3	1515	4.352	4.368	2.496	2.481
397	Sonido en formato MP3	1521	4.415	4.4	2.496	2.511
398	Sonido en formato MP3	1524	4.384	4.399	2.511	2.496
399	Sonido en formato MP3	1527	4.399	4.431	2.512	2.528
400	Sonido en formato MP3	1534	4.415	4.431	2.511	2.512
401	Sonido en formato MP3	1535	4.446	4.446	2.512	2.574
402	Sonido en formato MP3	1536	4.462	4.446	2.559	2.543
403	Sonido en formato MP3	1542	4.43	4.462	2.543	2.543
404	Sonido en formato MP3	1544	4.446	4.462	2.542	2.542
405	Sonido en formato MP3	1545	4.446	4.461	2.542	2.542
406	Sonido en formato MP3	1546	4.446	4.54	2.542	2.558
407	Sonido en formato MP3	1547	4.462	4.493	2.443	2.605
408	Sonido en formato MP3	1549	4.462	4.493	2.558	2.559
409	Sonido en formato MP3	1550	4.462	4.493	2.542	2.559
410	Sonido en formato MP3	1552	4.477	4.477	2.558	2.543
411	Sonido en formato MP3	1553	4.524	4.493	2.559	2.558
412	Sonido en formato MP3	1556	4.493	4.493	2.558	2.543
413	Sonido en formato MP3	1557	4.509	4.493	2.558	2.574
414	Sonido en formato MP3	1558	4.493	4.586	2.558	2.559
415	Sonido en formato MP3	1559	4.493	4.509	2.589	2.574

416	Sonido en formato MP3	1562	4.509	4.509	2.574	2.605
417	Sonido en formato MP3	1563	4.493	4.509	2.574	2.574
418	Sonido en formato MP3	1565	4.509	4.524	2.574	2.574
419	Sonido en formato MP3	1566	4.524	4.524	2.558	2.574
420	Sonido en formato MP3	1568	4.524	4.587	2.652	2.59
421	Sonido en formato MP3	1570	4.665	4.68	2.574	2.606
422	Sonido en formato MP3	1571	4.524	4.54	2.574	2.589
423	Sonido en formato MP3	1575	4.586	4.555	2.59	2.59
424	Sonido en formato MP3	1577	4.54	4.555	2.59	2.589
425	Sonido en formato MP3	1579	4.54	4.555	2.589	2.589
426	Sonido en formato MP3	1580	4.556	4.57	2.606	2.606
427	Sonido en formato MP3	1582	4.555	4.571	2.605	2.59
428	Sonido en formato MP3	1584	4.571	4.586	2.605	2.605
429	Sonido en formato MP3	1588	4.57	4.586	2.714	2.605
430	Imagen JPEG (.jpg)	1594	4.586	4.618	2.621	2.62
431	Sonido en formato MP3	1598	4.633	4.633	2.652	2.652
432	Sonido en formato MP3	1599	4.649	4.618	2.621	2.621
433	Sonido en formato MP3	1601	4.602	4.618	2.621	2.636
434	Sonido en formato MP3	1602	4.617	4.617	2.636	2.637
435	Sonido en formato MP3	1604	4.618	4.649	2.652	2.636
436	Imagen JPEG (.jpg)	1606	4.618	4.649	2.637	2.636
437	Sonido en formato MP3	1607	4.617	4.949	2.636	2.652
438	Sonido en formato MP3	1611	4.649	4.664	2.652	2.715
439	Sonido en formato MP3	1612	4.634	4.664	2.636	2.668
440	Sonido en formato MP3	1613	4.649	4.664	2.652	2.652
441	Sonido en formato MP3	1621	4.68	4.695	2.667	2.667
442	Sonido en formato MP3	1622	4.665	4.68	2.668	2.668
443	Sonido en formato MP3	1624	4.711	4.696	2.667	2.667
444	Sonido en formato MP3	1626	4.68	4.711	2.668	2.683
445	Sonido en formato MP3	1627	4.68	4.727	2.682	2.699
446	Sonido en formato MP3	1628	4.68	4.711	2.699	2.683
447	Sonido en formato MP3	1630	4.696	4.742	2.667	2.683
448	Sonido en formato MP3	1631	4.696	4.711	2.683	2.698
449	Sonido en formato MP3	1632	4.695	4.789	2.684	2.699
450	Sonido en formato MP3	1637	4.712	4.729	2.699	2.699
451	Sonido en formato MP3	1639	4.727	4.743	2.683	2.699
452	Sonido en formato MP3	1642	4.727	4.742	2.714	2.699
453	Sonido en formato MP3	1643	4.727	4.758	2.715	2.715
454	Sonido en formato MP3	1644	4.727	4.774	2.699	2.714
455	Sonido en formato MP3	1645	4.726	4.742	2.714	2.808
456	Sonido en formato MP3	1649	4.742	4.836	2.714	2.715
457	Sonido en formato MP3	1650	4.758	4.789	2.714	2.714
458	Sonido en formato MP3	1651	4.758	4.789	2.73	2.714
459	Sonido en formato MP3	1653	4.758	4.774	2.746	2.715
460	Sonido en formato MP3	1654	4.805	4.79	3.167	2.73
461	Sonido en formato MP3	1655	4.789	4.773	2.73	2.715
462	Sonido en formato MP3	1656	4.789	4.805	2.715	2.73
463	Sonido en formato MP3	1658	4.758	4.789	2.714	2.73
464	Sonido en formato MP3	1659	4.774	4.789	2.745	2.73
465	Sonido en formato MP3	1660	4.79	4.821	2.777	2.73
466	Sonido en formato MP3	1664	4.805	4.82	2.761	2.73
467	Sonido en formato MP3	1666	4.789	4.82	2.745	2.73
468	Sonido en formato MP3	1667	4.804	4.821	2.745	2.792
469	Sonido en formato MP3	1679	4.836	4.851	2.761	2.761
470	Sonido en formato MP3	1680	4.836	4.898	2.776	2.761
471	Sonido en formato MP3	1681	4.867	4.867	2.761	2.761
472	Sonido en formato MP3	1682	4.852	4.868	2.761	2.761
473	Sonido en formato MP3	1685	4.867	4.914	2.776	2.777
474	Sonido en formato MP3	1687	5.023	4.867	2.792	2.777
475	Sonido en formato MP3	1688	5.054	4.883	2.777	2.761
476	Sonido en formato MP3	1689	4.977	4.883	2.777	2.777
477	Sonido en formato MP3	1690	4.961	4.998	2.777	2.777
478	Sonido en formato MP3	1691	5.507	4.883	2.792	2.777
479	Sonido en formato MP3	1692	5.57	4.945	2.777	2.793
480	Sonido en formato MP3	1694	5.07	4.93	2.792	2.839
481	Sonido en formato MP3	1695	4.882	4.883	2.855	2.793
482	Sonido en formato MP3	1697	4.883	4.914	2.793	2.793
483	Sonido en formato MP3	1698	4.914	4.899	2.777	2.792
484	Sonido en formato MP3	1700	4.992	4.914	2.793	2.792
485	Sonido en formato MP3	1702	4.899	4.929	2.792	2.808
486	Sonido en formato MP3	1704	4.899	4.93	2.792	2.808
487	Sonido en formato MP3	1705	4.899	4.929	2.808	2.823
488	Sonido en formato MP3	1707	4.914	4.93	2.824	2.855
489	Sonido en formato MP3	1709	4.914	4.977	2.808	2.808
490	Sonido en formato MP3	1710	4.929	4.945	2.824	2.808
491	Sonido en formato MP3	1712	5.226	4.945	2.808	2.823
492	Sonido en formato MP3	1719	4.945	4.977	2.824	2.839
493	Sonido en formato MP3	1721	4.945	4.977	2.839	2.839
494	Sonido en formato MP3	1722	4.977	4.976	2.824	2.823
495	Sonido en formato MP3	1724	4.961	4.992	2.423	2.839
496	Sonido en formato MP3	1728	4.976	4.992	2.855	2.87
497	Sonido en formato MP3	1729	4.992	5.008	2.84	2.839
498	Sonido en formato MP3	1730	4.976	4.992	2.839	2.855
499	Sonido en formato MP3	1731	5.008	5.07	2.855	3.728
500	Sonido en formato MP3	1734	5.117	5.023	2.84	2.839
501	Sonido en formato MP3	1740	5.039	5.039	2.902	2.886

502	Sonido en formato MP3	1745	5.039	5.086	2.87	2.886
503	Sonido en formato MP3	1749	5.039	5.054	2.87	2.871
504	Sonido en formato MP3	1751	5.319	5.133	2.886	2.87
505	Sonido en formato MP3	1752	5.055	5.101	2.886	2.886
506	Sonido en formato MP3	1753	5.054	5.288	2.886	2.886
507	Sonido en formato MP3	1755	5.055	5.07	2.886	2.886
508	Sonido en formato MP3	1758	5.07	5.085	2.886	2.886
509	Sonido en formato MP3	1763	5.085	5.101	2.902	2.902
510	Sonido en formato MP3	1765	5.07	5.226	2.964	2.886
511	Sonido en formato MP3	1766	5.117	5.101	2.901	2.917
512	Sonido en formato MP3	1768	5.102	5.102	2.902	2.933
513	Sonido en formato MP3	1771	5.21	5.116	2.917	2.917
514	Sonido en formato MP3	1778	5.163	5.148	2.948	2.964
515	Sonido en formato MP3	1779	5.21	5.148	2.918	2.933
516	Sonido en formato MP3	1780	5.195	5.148	2.917	2.933
517	Sonido en formato MP3	1784	5.116	5.148	2.933	2.933
518	Sonido en formato MP3	1785	5.132	5.148	2.917	2.949
519	Sonido en formato MP3	1790	5.304	5.163	2.949	2.948
520	Sonido en formato MP3	1794	5.32	5.179	2.949	2.964
521	Sonido en formato MP3	1795	5.242	5.179	2.948	2.949
522	Sonido en formato MP3	1798	5.226	5.195	2.995	2.949
523	Sonido en formato MP3	1799	5.164	5.195	2.964	2.964
524	Sonido en formato MP3	1804	5.195	5.226	2.964	2.964
525	Sonido en formato MP3	1805	5.304	5.211	2.964	2.964
526	Sonido en formato MP3	1807	5.195	5.304	2.964	2.98
527	Sonido en formato MP3	1808	5.273	5.211	3.011	2.995
528	Sonido en formato MP3	1812	5.257	5.272	2.98	2.979
529	Sonido en formato MP3	1815	5.304	5.242	2.98	2.995
530	Sonido en formato MP3	1816	5.242	5.257	2.995	2.995
531	Sonido en formato MP3	1817	5.335	5.258	2.996	2.995
532	Microsoft PowerPoint (.pptx)	1819	5.257	5.257	2.979	2.98
533	Sonido en formato MP3	1821	5.351	5.273	2.995	2.979
534	Sonido en formato MP3	1825	5.273	5.257	3.011	3.011
535	Sonido en formato MP3	1829	5.398	5.319	2.995	3.011
536	Sonido en formato MP3	1832	5.382	5.304	3.011	3.011
537	Sonido en formato MP3	1835	5.382	5.304	3.027	3.027
538	Sonido en formato MP3	1841	5.413	5.319	3.026	3.026
539	Sonido en formato MP3	1843	5.397	5.319	3.073	3.042
540	Sonido en formato MP3	1860	5.444	5.366	3.088	3.057
541	Sonido en formato MP3	1864	5.46	5.382	3.073	3.074
542	Sonido en formato MP3	1865	5.492	5.382	3.073	3.057
543	Sonido en formato MP3	1871	5.382	5.414	3.12	3.073
544	Sonido en formato MP3	1872	5.46	5.444	3.089	3.088
545	Sonido en formato MP3	1874	5.445	5.413	3.073	3.105
546	Sonido en formato MP3	1877	5.522	5.444	3.089	3.088
547	Sonido en formato MP3	1878	5.428	5.429	3.089	3.089
548	Sonido en formato MP3	1879	5.492	5.429	3.089	3.088
549	Sonido en formato MP3	1880	5.538	5.444	3.089	3.105
550	Sonido en formato MP3	1884	5.46	5.444	3.104	3.104
551	Sonido en formato MP3	1887	5.428	5.444	3.104	3.089
552	Sonido en formato MP3	1889	5.553	5.46	3.152	3.167
553	Sonido en formato MP3	1890	5.554	5.476	3.104	3.104
554	Sonido en formato MP3	1891	5.585	5.46	3.12	3.12
555	Sonido en formato MP3	1892	5.554	5.476	3.12	3.12
556	Sonido en formato MP3	1894	5.553	5.476	3.104	3.12
557	Sonido en formato MP3	1896	5.647	5.491	3.151	3.12
558	Sonido en formato MP3	1903	5.569	5.491	3.276	3.135
559	Sonido en formato MP3	1906	5.632	5.834	3.135	3.136
560	Sonido en formato MP3	1907	5.585	5.507	3.12	3.151
561	Sonido en formato MP3	1908	5.585	5.507	3.136	3.135
562	Sonido en formato MP3	1909	5.491	5.522	3.151	3.152
563	Sonido en formato MP3	1910	5.632	5.507	3.135	3.136
564	Sonido en formato MP3	1912	5.523	5.553	3.151	3.151
565	Sonido en formato MP3	1914	5.516	5.523	3.198	3.136
566	Sonido en formato MP3	1916	5.522	5.538	3.152	3.167
567	Sonido en formato MP3	1919	5.522	5.538	3.152	3.151
568	Sonido en formato MP3	1920	5.647	5.538	3.151	3.167
569	Sonido en formato MP3	1921	5.616	5.569	3.166	3.151
570	Sonido en formato MP3	1922	5.522	5.553	3.167	3.167
571	Sonido en formato MP3	1923	5.6	5.756	3.166	3.166
572	Sonido en formato MP3	1925	5.585	5.663	3.152	3.182
573	Sonido en formato MP3	1926	5.554	5.569	3.167	3.182
574	Sonido en formato MP3	1927	5.647	5.554	3.167	3.166
575	Sonido en formato MP3	1929	5.616	5.585	3.182	3.183
576	Sonido en formato MP3	1930	5.679	5.663	3.166	3.167
577	Imagen JPEG (.jpg)	1934	5.663	5.569	3.214	3.213
578	Sonido en formato MP3	1937	5.662	5.585	3.229	3.182
579	Sonido en formato MP3	1939	5.725	5.601	3.183	3.182
580	Sonido en formato MP3	1942	5.679	5.663	3.198	3.198
581	Sonido en formato MP3	1946	5.616	5.647	3.198	3.198
582	Sonido en formato MP3	1947	5.616	5.663	3.214	3.198
583	Sonido en formato MP3	1949	5.757	5.631	3.214	3.198
584	Imagen JPEG (.jpg)	1951	5.741	5.648	3.198	3.213
585	Sonido en formato MP3	1954	5.709	5.647	3.229	3.214
586	Microsoft Excel (.xlsx)	1958	5.725	5.632	3.214	3.214
587	Sonido en formato MP3	1962	5.756	5.741	3.229	4.243

588	Sonido en formato MP3	1963	5.756	5.663	3.214	3.23
589	Sonido en formato MP3	1965	5.694	5.678	3.26	3.229
590	Sonido en formato MP3	1968	5.788	5.787	3.245	3.276
591	Sonido en formato MP3	1977	5.694	5.803	3.245	3.245
592	Sonido en formato MP3	1978	5.694	5.788	3.261	3.26
593	Imagen JPEG (.jpg)	1983	5.803	5.834	3.245	3.26
594	Imagen JPEG (.jpg)	1985	5.804	5.866	3.261	3.26
595	Sonido en formato MP3	1988	5.834	5.85	3.26	3.323
596	Sonido en formato MP3	1990	5.834	5.741	3.354	3.276
597	Sonido en formato MP3	1991	5.741	5.85	3.276	3.292
598	Imagen JPEG (.jpg)	1994	5.85	5.834	3.276	3.292
599	Sonido en formato MP3	1999	5.866	5.882	3.276	3.292
600	Imagen JPEG (.jpg)	2003	5.881	5.881	3.292	3.291
601	Sonido en formato MP3	2007	5.865	5.819	3.323	3.291
602	Sonido en formato MP3	2009	5.866	5.897	3.323	3.291
603	Sonido en formato MP3	2010	5.897	5.912	3.432	3.307
604	Sonido en formato MP3	2011	8.112	5.959	3.323	3.307
605	Imagen JPEG (.jpg)	2015	5.913	5.997	3.323	3.308
606	Sonido en formato MP3	2019	5.818	5.882	3.323	3.323
607	Sonido en formato MP3	2023	1.928	6.006	3.323	3.323
608	Sonido en formato MP3	2026	5.912	5.912	3.354	3.339
609	Sonido en formato MP3	2029	5.959	5.944	3.354	3.338
610	Sonido en formato MP3	2030	5.944	5.959	3.354	3.354
611	Microsoft PowerPoint (.pptx)	2034	6.006	5.959	3.354	3.354
612	Sonido en formato MP3	2036	5.99	5.959	3.338	3.354
613	Sonido en formato MP3	2036	5.897	5.896	3.354	3.369
614	Imagen JPEG (.jpg)	2042	6.022	5.896	3.37	3.354
615	Sonido en formato MP3	2046	6.037	5.99	3.37	3.386
616	Imagen JPEG (.jpg)	2051	6.022	5.975	3.385	3.37
617	Imagen JPEG (.jpg)	2056	6.022	5.99	3.386	3.37
618	Sonido en formato MP3	2058	6.053	6.037	3.401	3.386
619	Sonido en formato MP3	2060	5.928	6.038	3.417	3.386
620	Sonido en formato MP3	2061	6.053	6.006	3.4	3.401
621	Imagen JPEG (.jpg)	2063	6.052	6.037	3.448	3.401
622	Imagen JPEG (.jpg)	2069	6.037	6.1	3.401	3.401
623	Imagen JPEG (.jpg)	2074	6.1	6.084	3.401	3.416
624	Sonido en formato MP3	2077	6.116	6.084	3.416	3.416
625	Sonido en formato MP3	2080	5.99	6.146	3.495	3.447
626	Sonido en formato MP3	2084	6.068	6.1	3.557	3.417
627	Sonido en formato MP3	2087	6.147	6.115	3.479	3.447
628	Sonido en formato MP3	2091	6.116	6.147	3.463	3.432
629	Sonido en formato MP3	2094	6.115	6.13	3.495	3.447
630	Sonido en formato MP3	2103	6.224	6.162	3.51	3.479
631	Imagen JPEG (.jpg)	2106	6.162	6.068	3.463	3.51
632	Sonido en formato MP3	2111	6.162	6.115	3.463	4.836
633	Imagen JPEG (.jpg)	2113	6.084	6.303	3.51	3.479
634	Imagen JPEG (.jpg)	2118	6.209	6.209	3.494	3.479
635	Sonido en formato MP3	2121	6.1	6.224	3.494	3.479
636	Sonido en formato MP3	2122	6.116	6.256	3.495	3.51
637	Sonido en formato MP3	2124	6.115	6.224	3.572	3.494
638	Imagen JPEG (.jpg)	2128	6.334	6.24	3.542	3.495
639	Sonido en formato MP3	2131	6.131	6.209	3.495	3.51
640	Sonido en formato MP3	2135	6.162	6.287	3.495	3.495
641	Sonido en formato MP3	2137	6.162	6.162	3.541	3.51
642	Imagen JPEG (.jpg)	2140	6.349	6.381	3.51	3.525
643	Sonido en formato MP3	2144	6.349	6.349	3.525	3.526
644	Sonido en formato MP3	2146	6.286	6.286	3.526	3.525
645	Sonido en formato MP3	2149	6.225	6.193	3.525	3.541
646	Imagen JPEG (.jpg)	2152	6.209	6.334	3.541	3.541
647	Imagen JPEG (.jpg)	2162	6.349	6.333	3.572	3.604
648	Sonido en formato MP3	2169	6.349	6.256	3.557	3.588
649	Sonido en formato MP3	2175	6.365	6.396	3.572	3.572
650	Imagen JPEG (.jpg)	2179	6.287	6.287	3.573	3.588
651	Imagen JPEG (.jpg)	2185	6.365	6.411	3.651	3.604
652	Imagen JPEG (.jpg)	2191	6.412	6.412	3.682	3.588
653	Sonido en formato MP3	2194	6.411	6.412	3.651	3.604
654	Sonido en formato MP3	2201	6.443	6.443	3.619	3.619
655	Imagen JPEG (.jpg)	2212	6.505	6.537	3.634	3.635
656	Imagen JPEG (.jpg)	2217	6.521	6.252	3.697	3.681
657	Imagen JPEG (.jpg)	2222	6.536	6.614	3.666	3.65
658	Imagen JPEG (.jpg)	2227	6.63	6.427	3.697	3.682
659	Sonido en formato MP3	2232	6.443	6.599	3.682	3.666
660	Sonido en formato MP3	2236	6.677	6.552	3.682	3.666
661	Sonido en formato MP3	2242	6.599	6.474	3.682	3.698
662	Imagen JPEG (.jpg)	2244	6.567	6.63	3.697	3.697
663	Sonido en formato MP3	2249	6.489	6.583	3.681	3.697
664	Sonido en formato MP3	2256	6.505	6.521	3.713	3.697
665	Sonido en formato MP3	2260	6.552	6.677	3.728	3.728
666	Imagen JPEG (.jpg)	2264	6.661	6.817	3.728	3.729
667	Sonido en formato MP3	2273	6.693	6.708	3.744	3.744
668	Sonido en formato MP3	2275	6.708	6.677	3.744	3.744
669	Imagen JPEG (.jpg)	2282	6.583	6.724	3.791	3.822
670	Imagen JPEG (.jpg)	2286	6.693	6.692	3.76	3.76
671	Sonido en formato MP3	2290	6.739	6.677	3.76	3.776
672	Sonido en formato MP3	2291	6.692	6.755	3.775	3.759
673	Imagen JPEG (.jpg)	2295	6.708	6.723	4.976	3.775

674	Imagen JPEG (.jpg)	2298	6.708	6.739	3.791	3.775
675	Sonido en formato MP3	2306	6.723	6.661	3.791	3.791
676	Sonido en formato MP3	2314	6.786	6.817	3.806	3.806
677	Sonido en formato MP3	2317	6.739	6.771	3.806	3.806
678	Imagen JPEG (.jpg)	2321	6.724	6.802	3.822	3.822
679	Imagen JPEG (.jpg)	2327	6.833	6.817	3.916	3.837
680	Sonido en formato MP3	2332	6.818	6.879	3.822	3.838
681	Sonido en formato MP3	2336	6.896	6.833	3.837	3.822
682	Imagen JPEG (.jpg)	2340	6.724	6.911	3.884	3.916
683	Imagen JPEG (.jpg)	2345	6.864	6.942	3.853	3.853
684	Imagen JPEG (.jpg)	2348	6.833	6.864	3.868	3.853
685	Imagen JPEG (.jpg)	2352	6.895	6.958	3.853	3.853
686	Imagen JPEG (.jpg)	2359	6.942	6.927	3.884	3.885
687	Imagen JPEG (.jpg)	2365	6.957	6.849	3.931	3.9
688	Sonido en formato MP3	2369	6.942	6.973	3.9	3.9
689	Sonido en formato MP3	2373	6.942	6.848	3.9	3.915
690	Sonido en formato MP3	2377	7.082	6.942	3.916	3.9
691	Sonido en formato MP3	2380	7.004	6.879	3.916	3.916
692	Sonido en formato MP3	2382	6.957	7.036	3.916	3.915
693	Imagen JPEG (.jpg)	2385	6.989	6.88	3.931	4.009
694	Imagen JPEG (.jpg)	2388	6.879	7.004	3.931	3.931
695	Sonido en formato MP3	2390	6.974	6.927	3.931	3.931
696	Sonido en formato MP3	2394	6.989	7.051	3.932	3.947
697	Sonido en formato MP3	2407	6.926	6.957	3.963	3.962
698	Imagen JPEG (.jpg)	2410	6.957	7.005	4.025	3.947
699	Imagen JPEG (.jpg)	2416	6.958	6.973	3.963	4.009
700	Imagen JPEG (.jpg)	2420	7.005	7.192	3.978	4.071
701	Imagen JPEG (.jpg)	2425	7.005	7.254	3.993	4.041
702	Sonido en formato MP3	2428	7.02	7.082	3.978	4.024
703	Imagen JPEG (.jpg)	2431	7.192	7.02	3.994	4.056
704	Imagen JPEG (.jpg)	2436	7.067	7.161	4.01	4.025
705	Imagen JPEG (.jpg)	2439	7.051	7.051	4.009	4.025
706	Imagen JPEG (.jpg)	2443	7.052	7.176	4.056	4.072
707	Imagen JPEG (.jpg)	2447	7.067	7.286	4.025	4.04
708	Imagen JPEG (.jpg)	2451	7.285	7.067	4.025	4.04
709	Imagen JPEG (.jpg)	2454	7.098	7.098	4.056	4.056
710	Sonido en formato MP3	2457	7.176	7.347	4.056	4.04
711	Imagen JPEG (.jpg)	2460	7.098	7.082	4.041	4.056
712	Sonido en formato MP3	2464	7.113	7.27	4.056	4.072
713	Sonido en formato MP3	2468	7.129	7.238	4.056	4.071
714	Sonido en formato MP3	2470	7.129	7.223	4.056	4.071
715	Imagen JPEG (.jpg)	2474	7.301	7.238	4.056	4.103
716	Imagen JPEG (.jpg)	2477	7.3	7.238	4.119	4.103
717	Sonido en formato MP3	2481	7.348	7.161	4.087	4.087
718	Imagen JPEG (.jpg)	2485	7.301	7.161	4.087	4.087
719	Sonido en formato MP3	2492	7.348	7.207	4.118	4.15
720	Sonido en formato MP3	2494	7.191	7.301	4.102	4.119
721	Imagen JPEG (.jpg)	2500	7.207	7.208	4.102	4.212
722	Sonido en formato MP3	2509	7.441	7.394	4.166	4.212
723	Imagen JPEG (.jpg)	2515	7.41	7.254	4.134	4.149
724	Imagen JPEG (.jpg)	2518	7.254	7.41	4.15	4.15
725	Imagen JPEG (.jpg)	2525	7.285	7.426	4.149	4.181
726	Sonido en formato MP3	2528	7.488	7.394	4.149	4.165
727	Imagen JPEG (.jpg)	2532	7.285	7.301	4.165	4.181
728	Imagen JPEG (.jpg)	2536	7.472	7.426	4.228	4.18
729	Imagen JPEG (.jpg)	2539	7.504	7.55	4.197	4.18
730	Imagen JPEG (.jpg)	2543	7.332	7.347	4.181	4.197
731	Imagen JPEG (.jpg)	2547	7.41	7.347	4.227	4.212
732	Imagen JPEG (.jpg)	2551	7.472	7.363	4.181	4.196
733	Sonido en formato MP3	2554	7.457	7.613	4.274	4.212
734	Imagen JPEG (.jpg)	2558	7.504	7.394	4.212	4.212
735	Imagen JPEG (.jpg)	2562	7.519	7.582	4.213	4.228
736	Imagen JPEG (.jpg)	2565	7.519	7.628	4.212	4.228
737	Imagen JPEG (.jpg)	2569	7.41	7.519	4.228	4.243
738	Imagen JPEG (.jpg)	2574	7.628	7.519	4.274	4.227
739	Imagen JPEG (.jpg)	2577	7.504	7.426	4.228	4.243
740	Imagen JPEG (.jpg)	2581	7.629	7.597	4.259	4.259
741	Sonido en formato MP3	2585	7.55	7.457	4.243	4.259
742	Sonido en formato MP3	2590	7.645	7.582	4.259	4.274
743	Imagen JPEG (.jpg)	2594	7.519	7.503	4.275	4.368
744	Imagen JPEG (.jpg)	2597	7.644	7.488	4.274	4.306
745	Imagen JPEG (.jpg)	2600	7.675	7.535	4.274	4.306
746	Imagen JPEG (.jpg)	2603	7.55	7.769	4.274	4.306
747	Imagen JPEG (.jpg)	2606	7.519	7.691	4.29	4.29
748	Imagen JPEG (.jpg)	2609	7.675	7.706	4.306	4.321
749	Imagen JPEG (.jpg)	2615	7.534	7.66	4.29	4.305
750	Imagen JPEG (.jpg)	2619	7.566	7.675	4.305	4.321
751	Imagen JPEG (.jpg)	2621	7.582	7.582	4.321	4.321
752	Imagen JPEG (.jpg)	2624	7.566	7.582	4.321	4.321
753	Imagen JPEG (.jpg)	2628	7.722	7.737	4.336	4.352
754	Sonido en formato MP3	2632	7.768	7.597	4.337	4.352
755	Imagen JPEG (.jpg)	2638	7.659	7.769	4.336	4.352
756	Sonido en formato MP3	2645	7.8	7.754	4.368	4.353
757	Imagen JPEG (.jpg)	2649	7.628	7.644	4.384	4.399
758	Imagen JPEG (.jpg)	2653	7.8	7.815	4.368	4.368
759	Imagen JPEG (.jpg)	2656	7.878	7.691	4.353	4.384

760	Imagen JPEG (.jpg)	2658	7.831	7.816	4.383	4.384
761	Microsoft PowerPoint (.pptx)	2666	7.785	7.831	4.461	4.399
762	Imagen JPEG (.jpg)	2671	7.831	7.707	4.399	4.399
763	Imagen JPEG (.jpg)	2677	7.753	7.784	4.399	4.431
764	Imagen JPEG (.jpg)	2682	7.784	7.769	4.399	4.43
765	Imagen JPEG (.jpg)	2686	8.019	7.878	4.431	4.431
766	Imagen JPEG (.jpg)	2688	7.815	7.925	4.446	4.43
767	Imagen JPEG (.jpg)	2691	7.956	8.409	4.43	4.43
768	Imagen JPEG (.jpg)	2694	7.956	7.925	4.43	4.431
769	Imagen JPEG (.jpg)	2698	7.784	7.909	4.431	4.509
770	Imagen JPEG (.jpg)	2701	7.816	7.863	4.446	4.462
771	Imagen JPEG (.jpg)	2705	7.815	7.971	4.446	4.462
772	Imagen JPEG (.jpg)	2709	8.08	7.8	4.524	4.493
773	Imagen JPEG (.jpg)	2714	7.956	7.987	4.462	4.493
774	Imagen JPEG (.jpg)	2719	7.847	7.847	4.461	4.54
775	Imagen JPEG (.jpg)	2723	7.863	7.987	4.493	4.493
776	Imagen JPEG (.jpg)	2727	7.878	7.862	4.477	4.509
777	Imagen JPEG (.jpg)	2731	7.894	8.049	4.493	4.493
778	Imagen JPEG (.jpg)	2736	8.112	8.002	4.492	4.524
779	Imagen JPEG (.jpg)	2741	8.158	8.081	4.524	4.524
780	Imagen JPEG (.jpg)	2744	7.909	8.018	4.508	4.524
781	Imagen JPEG (.jpg)	2749	8.112	7.941	4.524	4.556
782	Imagen JPEG (.jpg)	2752	7.94	8.097	4.524	4.54
783	Imagen JPEG (.jpg)	2756	7.987	8.034	4.54	4.602
784	Imagen JPEG (.jpg)	2759	7.941	8.112	4.571	4.54
785	Imagen JPEG (.jpg)	2761	8.185	8.097	4.556	4.54
786	Sonido en formato MP3	2764	8.159	7.972	4.555	4.555
787	Sonido en formato MP3	2768	8.128	8.185	4.555	4.556
788	Imagen JPEG (.jpg)	2771	8.112	8.096	4.617	4.556
789	Imagen JPEG (.jpg)	2775	7.987	8.159	4.571	4.758
790	Imagen JPEG (.jpg)	2779	8.158	8.019	4.571	4.586
791	Sonido en formato MP3	2785	8.175	8.19	4.586	4.571
792	Sonido en formato MP3	2790	8.049	8.065	4.586	4.586
793	Imagen JPEG (.jpg)	2794	8.175	8.221	4.602	4.602
794	Imagen JPEG (.jpg)	2799	8.19	8.065	4.617	4.602
795	Sonido en formato MP3	2806	8.097	8.081	4.618	4.618
796	Imagen JPEG (.jpg)	2810	8.284	8.268	4.618	4.633
797	Sonido en formato MP3	2814	8.268	8.268	4.633	4.633
798	Sonido en formato MP3	2825	8.143	8.159	4.633	4.695
799	Imagen JPEG (.jpg)	2818	8.315	8.159	4.648	4.712
800	Imagen JPEG (.jpg)	2832	8.378	8.424	4.68	4.648
801	Imagen JPEG (.jpg)	2835	8.206	8.346	4.648	4.665
802	Imagen JPEG (.jpg)	2838	8.175	8.299	4.665	4.68
803	Imagen JPEG (.jpg)	2841	8.19	8.409	4.665	4.68
804	Imagen JPEG (.jpg)	2844	8.222	8.206	4.68	4.68
805	Imagen JPEG (.jpg)	2848	8.284	8.393	4.68	4.711
806	Imagen JPEG (.jpg)	2851	8.221	8.205	4.696	4.696
807	Imagen JPEG (.jpg)	2854	8.3	8.393	4.696	4.711
808	Imagen JPEG (.jpg)	2856	8.456	8.237	4.696	4.695
809	Imagen JPEG (.jpg)	2861	8.408	8.424	4.711	4.805
810	Sonido en formato MP3	2865	8.534	8.424	4.727	4.712
811	Imagen JPEG (.jpg)	2871	8.252	8.393	4.711	4.727
812	Imagen JPEG (.jpg)	2884	8.3	8.33	4.742	4.742
813	Imagen JPEG (.jpg)	2890	8.346	8.502	4.743	4.774
814	Imagen JPEG (.jpg)	2897	8.548	8.455	4.758	4.852
815	Imagen JPEG (.jpg)	2900	8.346	8.471	4.774	4.773
816	Imagen JPEG (.jpg)	2904	8.439	8.377	4.774	4.79
817	Imagen JPEG (.jpg)	2909	8.378	8.408	4.789	4.789
818	Imagen JPEG (.jpg)	2912	10.249	8.97	4.79	4.79
819	Imagen JPEG (.jpg)	2919	8.408	8.517	4.805	4.883
820	Imagen JPEG (.jpg)	2923	8.658	8.44	4.804	4.805
821	Imagen JPEG (.jpg)	2929	8.642	8.456	4.82	4.836
822	Imagen JPEG (.jpg)	2934	8.471	8.736	4.82	4.836
823	Imagen JPEG (.jpg)	2939	8.611	8.767	4.82	4.836
824	Imagen JPEG (.jpg)	2944	8.487	8.768	4.836	4.851
825	Imagen JPEG (.jpg)	2949	8.658	8.503	4.867	4.882
826	Imagen JPEG (.jpg)	2953	8.533	8.674	4.851	4.868
827	Imagen JPEG (.jpg)	2958	8.502	8.533	4.867	4.868
828	Imagen JPEG (.jpg)	2963	8.736	8.798	4.867	4.882
829	Imagen JPEG (.jpg)	2968	8.533	8.626	4.883	4.929
830	Imagen JPEG (.jpg)	2973	8.533	8.751	4.882	4.898
831	Sonido en formato MP3	2977	8.83	8.596	4.898	4.898
832	Imagen JPEG (.jpg)	2980	8.564	8.643	4.883	4.899
833	Sonido en formato MP3	2989	8.596	8.452	4.929	4.946
834	Sonido en formato MP3	2995	8.861	8.689	4.929	4.945
835	Imagen JPEG (.jpg)	3000	8.689	8.642	4.976	4.945
836	Imagen JPEG (.jpg)	3005	8.861	8.767	4.946	4.961
837	Imagen JPEG (.jpg)	3011	8.892	8.673	4.929	4.961
838	Imagen JPEG (.jpg)	3017	8.892	8.861	4.961	4.976
839	Sonido en formato MP3	3021	8.767	12.402	4.961	4.992
840	Imagen JPEG (.jpg)	3031	8.924	9.001	4.976	4.992
841	Imagen JPEG (.jpg)	3042	8.736	8.954	4.992	5.07
842	Imagen JPEG (.jpg)	3052	8.798	8.799	5.023	5.038
843	Imagen JPEG (.jpg)	3061	8.861	8.97	5.039	5.07
844	Imagen JPEG (.jpg)	3070	8.86	8.86	5.039	5.07
845	Sonido en formato MP3	3086	8.892	8.908	5.07	5.086

846	Imagen JPEG (.jpg)	3090	8.908	8.923	5.07	5.07
847	Imagen JPEG (.jpg)	3097	8.923	9.033	5.086	5.117
848	Imagen JPEG (.jpg)	3104	8.939	8.955	5.102	5.101
849	Imagen JPEG (.jpg)	3119	8.97	9.266	5.148	5.132
850	Imagen JPEG (.jpg)	3125	9.22	9.033	5.132	5.148
851	Imagen JPEG (.jpg)	3131	8.986	9.235	5.164	5.163
852	Imagen JPEG (.jpg)	3147	9.126	9.298	5.179	5.179
853	Imagen JPEG (.jpg)	3150	9.61	9.095	5.179	5.179
854	Imagen JPEG (.jpg)	3158	9.11	9.111	5.179	5.211
855	Microsoft Excel (.xlsx)	3166	9.344	9.345	5.195	5.21
856	Imagen JPEG (.jpg)	3171	9.142	9.142	5.21	5.21
857	Imagen JPEG (.jpg)	3184	9.173	9.188	5.241	5.257
858	Imagen JPEG (.jpg)	3190	9.47	9.297	5.257	5.257
859	Imagen JPEG (.jpg)	3201	9.438	9.423	5.288	5.289
860	Sonido en formato MP3	3208	9.266	9.266	5.288	5.289
861	Imagen JPEG (.jpg)	3209	9.407	9.266	5.288	5.289
862	Imagen JPEG (.jpg)	3223	9.282	9.516	5.382	5.32
863	Microsoft Excel (.xlsx)	3226	9.376	9.547	5.319	5.319
864	Imagen JPEG (.jpg)	3230	9.297	9.687	5.32	5.32
865	Imagen JPEG (.jpg)	3241	9.314	9.344	5.319	5.553
866	Imagen JPEG (.jpg)	3251	9.532	9.641	5.351	5.351
867	Imagen JPEG (.jpg)	3265	9.391	9.423	5.366	5.398
868	Imagen JPEG (.jpg)	3274	9.641	9.453	5.382	5.398
869	Imagen JPEG (.jpg)	3281	10.171	9.47	5.398	5.398
870	Sonido en formato MP3	3290	9.501	9.657	5.413	5.429
871	Imagen JPEG (.jpg)	3294	9.578	9.687	5.554	5.413
872	Sonido en formato MP3	3354	9.953	9.843	5.632	5.507
873	Sonido en formato MP3	3381	9.953	9.922	5.553	5.569
874	Sonido en formato MP3	3385	9.75	9.781	5.569	5.585
875	Sonido en formato MP3	3472	9.984	10.015	5.709	5.726
876	Sonido en formato MP3	3489	10.171	10.28	5.741	5.741
877	Sonido en formato MP3	3564	10.436	10.281	5.85	5.881
878	Sonido en formato MP3	3575	10.561	10.484	5.865	5.881
879	Imagen JPEG (.jpg)	3600	10.374	10.53	5.897	5.928
880	Sonido en formato MP3	3700	10.686	10.842	6.084	6.115
881	Sonido en formato MP3	3787	11.123	10.92	6.256	6.225
882	Sonido en formato MP3	3863	11.154	11.138	6.334	6.365
883	Sonido en formato MP3	3935	11.622	11.653	6.474	6.49
884	Sonido en formato MP3	4098	11.81	11.825	6.724	6.739
885	Imagen JPEG (.jpg)	4160	12.059	12.199	6.833	6.849
886	Imagen JPEG (.jpg)	4192	12.355	12.23	6.88	6.911
887	Imagen JPEG (.jpg)	4222	12.309	12.152	6.957	6.927
888	Imagen JPEG (.jpg)	4277	12.434	12.636	7.02	7.036
889	Imagen JPEG (.jpg)	4333	12.496	12.511	7.114	7.13
890	Sonido en formato MP3	4401	12.932	12.932	7.222	7.254
891	Sonido en formato MP3	4434	12.948	12.917	7.27	7.285
892	Sonido en formato MP3	4440	12.808	12.917	7.301	7.316
893	Imagen JPEG (.jpg)	4467	13.197	12.886	7.364	7.363
894	Imagen JPEG (.jpg)	4522	13.041	13.26	7.426	7.425
895	Sonido en formato MP3	4635	13.603	13.603	7.613	7.629
896	Sonido en formato MP3	4691	13.791	13.65	7.706	7.722
897	Sonido en formato MP3	4764	13.744	14.087	10.311	7.831
898	Microsoft PowerPoint (.pptx)	4775	14.024	13.775	7.863	7.894
899	Sonido en formato MP3	4858	13.977	41.48	7.987	8.018
900	Imagen JPEG (.jpg)	4896	14.133	14.181	8.066	8.05
901	Sonido en formato MP3	4911	14.149	14.461	8.096	8.081
902	Imagen JPEG (.jpg)	4929	14.352	14.383	8.097	8.112
903	Imagen JPEG (.jpg)	4937	14.415	14.352	8.44	8.143
904	Imagen JPEG (.jpg)	4970	14.336	14.461	8.237	8.19
905	Imagen JPEG (.jpg)	4989	14.851	14.508	8.284	8.642
906	Vídeo MP4	5038	14.664	14.633	8.424	8.533
907	Imagen JPEG (.jpg)	5087	14.867	14.789	8.502	8.393
908	Imagen JPEG (.jpg)	5127	14.757	14.913	8.518	8.424
909	Sonido en formato MP3	5378	15.741	15.616	8.986	8.845
910	Sonido en formato MP3	5597	16.364	16.287	9.204	9.204
911	Imagen BMP (.bmp)	5640	16.239	16.349	9.453	9.282
912	Vídeo MP4	5705	16.474	16.552	9.657	9.376
913	Adobe Acrobat Document (.pdf)	5801	16.77	16.848	9.563	9.532
914	Adobe Acrobat Document (.pdf)	5829	17.363	16.801	9.89	9.594
915	Microsoft PowerPoint 97-2003	5969	17.488	17.877	9.953	9.812
916	Sonido en formato MP3	6056	18.47	17.955	10.125	9.969
917	Imagen JPEG (.jpg)	6139	18.034	18.892	10.249	10.156
918	Sonido en formato MP3	6344	18.657	18.783	10.655	10.484
919	Sonido en formato MP3	6430	20.779	18.783	10.592	10.545
920	Sonido en formato MP3	6488	21.325	23.198	10.842	10.671
921	Imagen JPEG (.jpg)	6852	22.074	20.171	11.747	11.263
922	Microsoft PowerPoint (.pptx)	6963	20.031	20.109	11.419	11.45
923	Microsoft Word (.docx)	7130	21.825	20.639	11.918	11.716
924	Base de Datos Access (.accdb)	7296	21.013	21.076	12.09	11.997
925	Vídeo MP4	7338	21.856	21.622	13.167	12.293
926	Imagen JPEG (.jpg)	7586	22.557	22.168	12.636	12.667
927	Vídeo MP4	7626	23.151	22.183	12.729	12.698
928	Imagen JPEG (.jpg)	7639	23.072	22.496	12.761	15.397
929	Adobe Acrobat Document (.pdf)	7822	23.478	23.369	13.01	12.87
930	Adobe Acrobat Document (.pdf)	7933	23.353	24.071	13.026	13.057
931	Vídeo MP4	8136	24.461	23.712	13.635	13.416

932	Sonido en formato MP3	8160	24.383	24.32	13.65	13.416
933	Archivo WinRAR ZIP (.zip)	8384	25.1	24.149	14.134	13.79
934	Microsoft PowerPoint (.pptx)	8609	25.038	26.848	14.602	14.149
935	Executable Jar File (.jar)	8716	26.395	25.272	14.632	14.337
936	Microsoft Word 97-2003	8954	25.834	26.099	14.945	14.71
937	Vídeo MP4	9032	26.817	26.785	15.038	14.851
938	Microsoft PowerPoint 97-2003	9071	26.691	26.162	15.038	14.882
939	Microsoft Word 97-2003	9282	28.735	27.144	15.459	15.288
940	Vídeo MP4	9324	28.595	27.799	15.678	15.428
941	Vídeo MP4	9442	27.16	27.331	15.49	15.616
942	Vídeo MP4	9489	29.26	27.596	15.6	15.584
943	Vídeo MP4	9530	28.642	28.174	21.809	16.302
944	Adobe Acrobat Document (.pdf)	9560	29.063	28.08	15.912	15.881
945	Microsoft PowerPoint (.pptx)	9726	28.33	28.044	15.959	16.083
946	Vídeo MP4	9877	28.642	28.688	16.365	16.645
947	Microsoft PowerPoint 97-2003	10181	29.313	29.391	16.692	16.754
948	Sonido en formato MP3	10620	30.795	31.153	17.769	18.05
949	Microsoft PowerPoint (.pptx)	10623	30.545	30.638	17.425	17.582
950	Microsoft Excel (.xlsx)	10939	31.465	31.559	17.94	17.956
951	Microsoft Excel (.xlsx)	11412	32.947	32.979	18.735	18.985
952	Microsoft Excel (.xlsx)	11760	33.836	33.962	19.297	19.359
953	Imagen JPEG (.jpg)	13409	38.672	40.061	22.183	22.262
954	Adobe Acrobat Document (.pdf)	13601	39.14	39.219	22.604	22.323
955	Audio (.3gp)	14899	42.885	42.963	24.602	24.461
956	Sonido en formato MP3	16567	47.736	48.907	27.488	27.831
957	Vídeo MP4	16602	47.767	47.892	27.441	27.207
958	Aplicación Ejecutable (.exe)	19794	56.925	57.143	32.635	32.479
959	Microsoft PowerPoint (.pptx)	22613	65.629	65.224	37.471	37.05
960	Adobe Acrobat Document (.pdf)	24577	71.105	71.51	40.794	40.591
961	Sonido en formato MP3	24794	72.447	73.133	41.667	40.857
962	Sonido en formato MP3	25587	75.27	75.27	41.949	42.151
963	Sonido en formato MP3	26853	77.204	77.828	44.585	44.086
964	Windows Media (.avi)	27114	78.483	78.765	44.647	44.647
965	Sonido en formato MP3	34381	99.543	100.667	56.675	57.19
966	Sonido en formato MP3	34925	100.682	101.306	57.596	57.252
967	Sonido en formato MP3	33227	104.099	106.705	59.451	60.247
968	Vídeo MP4	36700	105.581	106.689	60.357	62.088
969	Sonido en formato MP3	37212	106.985	108.436	60.996	61.059
970	Sonido en formato MP3	37369	108.186	109.387	61.823	61.402
971	Sonido en formato MP3	37387	107.827	109.278	61.324	61.433
972	Adobe Acrobat Document (.pdf)	40309	116.876	116.938	66.113	66.144
973	Sonido en formato MP3	40549	117.156	117.687	66.471	67.392
974	Sonido en formato MP3	55130	159.323	160.072	90.356	90.573
975	Sonido en formato MP3	56196	163.239	163.816	93.382	92.586
976	Vídeo MP4	60123	172.973	175.001	98.92	98.717
977	Vídeo MP4	64782	186.436	188.23	106.454	107.078
978	Adobe Acrobat Document (.pdf)	67312	194.06	195.671	110.277	110.619
979	Windows Media (.avi)	70524	203.378	205.063	115.565	116.844
980	Sonido en formato MP3	72116	207.73	215.639	118.951	118.67
981	Vídeo MP4	73490	211.63	217.382	122.273	121.493
982	Windows Media (.avi)	73897	212.956	214.719	121.088	121.103
983	Vídeo MP4	81980	235.716	238.088	134.535	134.863
984	Adobe Acrobat Document (.pdf)	89630	257.884	260.567	146.843	146.733
985	Windows Media (.avi)	103783	298.507	303.826	170.54	170.306
986	Vídeo MP4	105397	309.808	304.169	173.051	173.8
987	Vídeo MP4	116639	337.444	337.023	192.083	192.207
988	Vídeo MP4	118101	340.11	382.966	197.36	196.394
989	Vídeo MP4	123418	355.337	358.722	202.38	202.083
990	Vídeo MP4	130476	375.243	378.831	214.438	213.69
991	Vídeo MP4	137884	397.021	400.421	226.2	226.184
992	Vídeo MP4	149177	432.729	433.775	244.421	250.502
993	Vídeo MP4	163041	471.573	472.837	268.5	268.976
994	Windows Media (.avi)	174322	527.834	503.429	287.699	286.756
995	Vídeo MP4	182279	529.449	525.783	300.278	299.208
996	Vídeo MP4	201408	584.143	581.491	341.669	330.331
997	Adobe Acrobat Document (.pdf)	207080	596.623	598.791	339.644	340.409
998	Windows Media (.avi)	210919	606.014	623.526	347.884	345.884
999	Vídeo MP4	227344	654.015	670.177	374.526	406.233
1000	Vídeo MP4	261006	756.562	752.81	428.236	431.84

HIKARI Ltd
International Publishers of Science, Technology and Medicine

HIKARI

Site Search: Go

CONTEMPORARY ENGINEERING SCIENCES

ISSN 1313-6569 (print) ISSN 1314-7641 (online)

P. D. Filio-Aguilar, R. Flores-Carapia, V. M. Silva-Garcia
SSL/TLS record protocol based on the triple DES-96 cryptosystem
Contemporary Engineering Sciences, Vol. 7, 2014, no. 20, 947-956
http://dx.doi.org/10.12988/ces.2014.48115

Contemporary Engineering Sciences, Vol. 7, 2014, no. 20, 947 - 956
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2014.48115

SSL/TLS Record Protocol Based on the Triple DES-96 Cryptosystem

P. D. Filio-Aguilar

Universidad Autónoma del Estado de México
Hermenegildo Galeana No. 3 Col. María Isabel,
Valle de Chalco. C.P. 56615 Edo. de México

R. Flores-Carapia

Instituto Politécnico Nacional, CIDETEC
Av."Juan de Dios Bátiz" s/n esq. Miguel Othón
de Mendizábal, Col. Nueva Industrial Vallejo, Del. Gustavo
A. Madero, México, D.F., C.P. 07700

V. M. Silva-García

Instituto Politécnico Nacional, CIDETEC
Av."Juan de Dios Bátiz" s/n esq. Miguel Othón
de Mendizábal, Col. Nueva Industrial Vallejo, Del. Gustavo
A. Madero, México, D.F., C.P. 07700

Abstract

Due to the absolute safety needed when sensitive information via the Internet is used, complex algorithms and cryptographic protocols such as SSL/TLS are required. This paper presents a comparison of the Triple DES cipher system that is part of the suite of encryption SSL/TLS and the cryptographic system Triple DES-96 having the following differences: Triple DES-96 begins with an information block of 96 bits, and Triple DES with a 64 block bits, besides, Triple DES-96 takes a variable permutation PV from the Triple DES keys and the number e is calculated. Furthermore, the expansion table E moves to the output

Anexo C. Constancias de ponencias.

Universidad Autónoma del Estado de México

Centro Universitario UAEM Temascaltepec

Otorga la presente

CONSTANCIA

A: C. PEDRO DAVID FILIO AGUILAR

Por su participación en el

VIII Coloquio de investigación 2012-B de la MACSCO

Llevado a cabo el 13 de Diciembre del 2012, en el Centro Universitario UAEM Temascaltepec

U A E M
CENTRO UNIVERSITARIO
TEMASCALTEPEC
DIFUSIÓN

CU Temascaltepec

Dr. En Edu. Manuel Antonio Pérez Chávez
Encargado del Despacho del Centro Universitario
UAEM Temascaltepec y su Extensión Tejupilco

Dr. En Edu. Daniel Cardoso Jiménez
Subdirector Académico

Universidad Autónoma del Estado de México
Centro Universitario UAEM Atlacomulco

S E O T O R G A L A P R E S E N T E

CONSTANCIA

A: PEDRO DAVID FILIO AGUILAR

*Por su **PARTICIPACIÓN** en el "IX Coloquio de Investigación de la Maestría en Ciencias de la Computación 2013-A", llevado a cabo en el Centro Universitario UAEM Atlacomulco, el día 30 de mayo del 2013.*

PATRIA, CIENCIA Y TRABAJO
"2013, 50 Aniversario Luctuoso del Poeta Heriberto Enríquez"

M. EN A. FIDENCIO OCHOA FLORES
Encargado del Despacho de la Dirección del
Centro Universitario UAEM Atlacomulco

UAEM | Universidad Autónoma del Estado de México

Centro Universitario UAEM Valle de Chalco

4° Coloquio Internacional de Cómputo e Informática

13 al 15 de noviembre de 2013

Otorga la presente

Constancia

Al Ing. Filio Aguilar, Dr. Samuel Olmos Peña, Dra. Cristina Juárez Lardín y al Dr. René Guadalupe Cruz Flores

Por su participación con la Ponencia "Sistema de Encriptación de Mensaje Utilizado 3DES y RSA Aleatoriamente" en el marco del 4to. Coloquio Internacional de Computación e Informática

Valle de Chalco, Estado de México a 13 de Noviembre de 2013.

Dr. René Guadalupe Cruz Flores Coordinador de Investigación	Dra. Magally Martínez Reyes Directora del Centro Universitario	M. en Ed. Anabelem Soberanes Martín Líder de C. A. Cómputo Aplicado

UAEM | Universidad Autónoma del Estado de México

CUTex
Centro Universitario UAEM Texcoco

Otorga a

Filio Aguilar Pedro David

el presente

Reconocimiento

Por disertar su tema de investigación en este Campus Universitario el 28 de noviembre de 2013, en el marco del X Coloquio de Investigación de la Maestría en Ciencias de la Computación 2013-B.

Patria, Ciencia y Trabajo

"2013, 50 Aniversario Luctuoso del Poeta Heriberto Enríquez"

CENTRO UNIVERSITARIO UAEM TEXCOCO

Dr. en Ed. Carlos G. Vega Vargas
Encargado del Despacho de la Dirección
DIRECCIÓN

UAEM | Universidad Autónoma del Estado de México

Centro Universitario UAEM Valle de Chalco

CENTRO UNIVERSITARIO
XICO
VALLE DE CHALCO

Simposio Académico de Ciencia Investigación y Tecnología

Otorga la presente

Constancia

A: Filio Aguilar Pedro David

Por su participación con la Ponencia "Sistema de Encriptación de Mensajes Utilizando 3DES Y RSA Aleatoriamente" en el marco del Simposio Académico de Ciencia, Investigación y Tecnología

Valle de Chalco, Estado de México a 5 de Diciembre de 2013.

M. en Ed. Anabelem Soberanes Martin
Subdirectora Académica del C.U.

Dra. Magally Martinez Reyes
Directora del Centro Universitario

Anexo D. Código para el Cifrado-Descifrado de texto con DES.

```
/*
 * The Des encryption and decryption Data Encryption Standard (DES)
 * Copyright (C) Filio-Aguilar D. - UAEM and Fores-Carapia R. - CIDETEC
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version 3
 * of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along withthis program. If not, see <http://www.gnu.org/licenses/>.
 */
package mx.com.uaem.des;

/**
 * Clase para cifrar y/o descifrar texto con el Data Encryption Standard (DES)
 *
 * @author David Filio
 * @author Rolando Flores
 * @version 2.1.0
 */
public class Des {

    // Variable que guarda los valores que toman las llaves
    private static final int K[][] = new int[2][16];

    // Tabla de permutación IP
    static final int[] IP = {
        58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4,
        62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32, 24, 16, 8,
        57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3,
        61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47, 39, 31, 23, 15, 7};

    // Tabla de permutación IP a la -1
    static final int[] IP_1 = {
        40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63, 31,
        38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13, 53, 21, 61, 29,
        36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59, 27,
        34, 2, 42, 10, 50, 18, 58, 26, 33, 1, 41, 9, 49, 17, 57, 25};

    // Tabla de Expansión E BIT-SELECTION TABLE
```

```
static final int[] E = {
   32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9, 8, 9, 10, 11, 12, 13,
   12, 13, 14, 15, 16, 17, 16, 17, 18, 19, 20, 21, 20, 21, 22,
   23, 24, 25, 24, 25, 26, 27, 28, 29, 28, 29, 30, 31, 32, 1};

// 8 Cajas S Boxes
static final int[][] S = {
   {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
      0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
      4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0,
      15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13
   },
   {15, 1, 8, 14, 6, 11, 3, 2, 9, 7, 2, 13, 12, 0, 5, 10,
      3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5,
      0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15,
      13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9
   },
   {10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
      13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1,
      13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7,
      1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12
   },
   {7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15,
      13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9,
      10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
      3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14
   },
   {2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9,
      14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6,
      4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14,
      11, 8, 12, 7, 1, 14, 2, 12, 6, 15, 0, 9, 10, 4, 5, 3
   },
   {12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11,
      10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8,
      9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6,
      4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13
   },
   {4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1,
      13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6,
      1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
      6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12
   },
   {13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
      1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2,
      7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8,
      2, 1, 14, 7, 4, 10, 18, 13, 15, 12, 9, 0, 3, 5, 6, 11
   }};

// Tabla de permutación P Function
static final int[] P = {
   16, 7, 20, 21, 29, 12, 28, 17, 1, 15, 23, 26, 5, 18, 31, 10,
```

```
    2, 8, 24, 14, 32, 27, 3, 9, 19, 13, 30, 6, 22, 11, 4, 25};

// Permuted Choice 1
static final int[] PC1 = {
    57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18,
    10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36,
    63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22,
    14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4};

// Numero de corrimientos a la izquierda
static final int[] LeftShifts = {
    1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1};

// Permuted Choice 2
static final int[] PC2 = {
    14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10, 23, 19, 12, 4, 26,
    8, 16, 7, 27, 20, 13, 2, 41, 52, 31, 37, 47, 55, 30, 40, 51,
    45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32};

/**
 * Método para generar las llaves
 *
 * @param key long contiene el valor de las llaves
 */
public static void setKey(int key[]) {
    int keyp[] = new int[2];
    int i, j, k, l, t, b;
    keyp[0] = 0;
    keyp[1] = 0;
    // Ciclo para recorrer PC1, ingresan 64 bits y salen 56 bits
    for (i = 0; i < 56; i++) {
        if (PC1[i] <= 32) {
            j = 0;
        } else {
            j = 1;
        }
        t = PC1[i] - 1;
        k = (t) % 32;

        b = key[j] >> 31 - k;
        b &= 1;
        if (i < 28) {
            j = 0;
        } else {
            j = 1;
        }
        k = i % 28;
        b <<= (27 - k);
        keyp[j] |= b;
    }
    // Ciclo para pasar la información sobre LeftShifts
```

```
    for (i = 0; i < 16; i++) {
        t = keyp[0] >> (28 - LeftShifts[i]);
        keyp[0] <<= LeftShifts[i];
        keyp[0] |= t;
        keyp[0] &= 0xfffffff;
        t = keyp[1] >> (28 - LeftShifts[i]);
        keyp[1] <<= LeftShifts[i];
        keyp[1] |= t;
        keyp[1] &= 0xfffffff;
        K[0][i] = 0;
        K[1][i] = 0;

        // Ciclo para pasar por PC2, ingresan 56 bits y salen 48 bits
        for (j = 0; j < 48; j++) {
            if (PC2[j] <= 28) {
                k = 0;
            } else {
                k = 1;
            }
            l = (PC2[j] - 1) % 28;
            b = keyp[k] >> 27 - l;
            b &= 1;
            if (j < 24) {
                k = 0;
            } else {
                k = 1;
            }
            l = j % 24;
            b <<= (23 - l);
            // El resultado de 48 bits se gurada en K
            K[k][i] |= b;
        }
    }
}

/**
 * Método para cifrar el texto plano
 *
 * @param txt char Contiene el texto cifrado
 * @return res resultado
 */
public static int[] cypher(int txt[]) {
    int res[] = new int[2];
    int txtp[] = new int[2];
    int i, b, j, k;
    txtp[0] = 0;
    txtp[1] = 0;
    // Ciclo para pasar la información por la permutación inicial IP
    for (i = 0; i < 64; i++) {
        if (IP[i] <= 32) {
            j = 0;
```

```
            } else {
                j = 1;
            }
            k = (IP[i] - 1) % 32;
            b = txt[j] >> 31 - k;
            b &= 1;
            if (i < 32) {
                j = 0;
            } else {
                j = 1;
            }
            k = i % 32;
            b <<= (31 - k);
            txtp[j] |= b;
        }
        // Ciclos para pasar la información por la permutación inicial IP_1
        for (i = 0; i < 16; i++) {
            b = f(txtp[1], i);
            b ^= txtp[0];
            txtp[0] = txtp[1];
            txtp[1] = b;
        }
        res[0] = 0;
        res[1] = 0;
        for (i = 0; i < 64; i++) {
            if (IP_1[i] <= 32) {
                j = 1;
            } else {
                j = 0;
            }
            k = (IP_1[i] - 1) % 32;
            b = txtp[j] >> 31 - k;
            b &= 1;
            if (i < 32) {
                j = 0;
            } else {
                j = 1;
            }
            k = i % 32;
            b <<= (31 - k);
            res[j] |= b;
        }
        // Imprimir en consola el texto cifrado
        System.out.println(binary(txtp[0], 32) + "    " + binary(txtp[1], 32));
        return res; // Se regresa el valor en res
    }

    /**
     * Metodo para descifrar el texto cifrado
     *
     * @param txt contiene el texto descifrado
```

```
 * @return res resultado
 */
public static int[] decypher(int txt[]) {
    int res[] = new int[2];
    int txtp[] = new int[2];
    int i, b, j, k;
    txtp[0] = 0;
    txtp[1] = 0;
    for (i = 0; i < 64; i++) {
        if (IP[i] <= 32) {
            j = 0;
        } else {
            j = 1;
        }
        k = (IP[i] - 1) % 32;
        b = txt[j] >> 31 - k;
        b &= 1;
        if (i < 32) {
            j = 1;
        } else {
            j = 0;
        }
        k = i % 32;
        b <<= (31 - k);
        txtp[j] |= b;
    }
    for (i = 15; i >= 0; i--) {
        b = f(txtp[0], i);
        b ^= txtp[1];
        txtp[1] = txtp[0];
        txtp[0] = b;

    }
    res[0] = 0;
    res[1] = 0;
    for (i = 0; i < 64; i++) {
        if (IP_1[i] <= 32) {
            j = 0;
        } else {
            j = 1;
        }
        k = (IP_1[i] - 1) % 32;
        b = txtp[j] >> 31 - k;
        b &= 1;
        if (i < 32) {
            j = 0;
        } else {
            j = 1;
        }
        k = i % 32;
        b <<= (31 - k);
```

```
            res[j] |= b;
        }
        // Imprimir en consola el texto descifrado
        System.out.println(binary(txtp[0], 32) + "    " + binary(txtp[1], 32));
        return res; // Se regresa el valor en res
    }

    /**
     * Método que realiza las operaciones de la función f
     *
     * @param data información después de salir de la permutación E
     * @param ki Contiene las subllaves
     * @return a resultado de la información al salir de la permutación P
     */
    private static int f(int data, int ki) {
        int i, j, k, r, a, b;
        int[] datap = new int[2];
        datap[0] = 0;
        datap[1] = 0;
        for (i = 0; i < 48; i++) {
            b = data >> 32 - E[i];  // lado R de 32 bits
            b &= 1;
            if (i < 24) {
                j = 0;
            } else {
                j = 1;
            }
            k = i % 24;
            b <<= (23 - k);
            datap[j] |= b;
        }
        datap[0] ^= K[0][ki];
        datap[1] ^= K[1][ki];
        r = 0;
        // Ciclos para pasar la información por las SBoxes
        for (i = 0; i < 4; i++) {
            j = datap[0] >> (18 - i * 6);
            j &= 0x3f;
            a = (j >> 5) & 1;
            a <<= 5;
            b = j & 1;
            b <<= 4;
            a |= b;
            j >>= 1;
            j &= 0xf;
            j |= a;
            b = S[i][j];
            b <<= (12 - i * 4);
            r |= b;
        }
        r <<= 16;
```

```
        k = 0;
        for (i = 0; i < 4; i++) {
            j = datap[1] >> (18 - i * 6);
            j &= 0x3f;
            a = (j >> 5) & 1;
            a <<= 5;
            b = j & 1;
            b <<= 4;
            a |= b;
            j >>= 1;
            j &= 0xf;
            j |= a;
            b = S[i + 4][j] << (12 - i * 4);
            k |= b;
        }
        r |= k;
        a = 0;
        // pasar por la permutación P
        for (i = 0; i < 32; i++) {
            b = r >> (32 - P[i]);
            b &= 1;
            b <<= (31 - i);
            a |= b;
        }
        return a; // Se regresa el valor en a, de 32 bits
    }

    /**
     * Método para convertir a binario
     *
     * @param x cadena de texto
     * @param bits cadena de texto en binario
     * @return
     */
    private static String binary(int x, int bits) {
        String r = "";
        int b;
        int i;
        for (i = 0; i < bits; i++) {
            b = x >> (bits - i - 1);
            b &= 1;
            r += b;
        }
        return r;
    }

    /**
     * Método para convertir a hexadecimal
     *
     * @param x cadena de texto en hexadecimal
     * @return
```

```
     */
    private static String hex(int x) {
        String r = "";
        int b;
        int i;
        for (i = 0; i < 8; i++) {
            b = x >> (28 - i * 4);
            b &= 0xf;
            r += (char) (b < 10 ? b + 48 : b + 55);
        }
        return r;
    }

    /**
     * Metodo principal que hace ejecutable la clase Des
     *
     * @param args
     */
    public static void main(String[] args) {
        int key[] = new int[2];
        int txt[] = new int[2];
        //key en hexadecimal
        key[0] = 0x13345779;
        key[1] = 0x9bbcdff1;
        //texto plano
        txt[0] = 0x01234567;
        txt[1] = 0x89abcdef;
        setKey(key); //ingresamos al metodo setKey los valores de key
        txt = cypher(txt); // Guarda en txt el texto cifrado
        System.out.println(hex(txt[0]) + "   " + hex(txt[1])); //texto cifrado
        System.out.println("");
        txt = decypher(txt); // Guarda en txt el texto descifrado
        System.out.println(hex(txt[0]) + "   " + hex(txt[1])); //texto descifrado
    }
}
```

Anexo E. Código para el cifrado-descifrado de archivos con Triple DES.

```
/*
 * The TripleDes encryption and decryption files
 * Copyright (C) Filio-Aguilar D. - UAEM and Fores-Carapia R. - CIDETEC
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version 2
 * of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
 */
package mx.com.uaem.des;

/**
 * Clase para cifrar y/o descifrar archivos con TripleDes
 *
 * @author David Filio
 * @author Rolando Flores
 * @version 2.1.0
 */
public class TripleDes {

   // key 1, key 2, key 3
   private final long K1[] = new long[16];
   private final long K2[] = new long[16];
   private final long K3[] = new long[16];

   // Tabla de permutación IP
   static final int[] IP = {
      58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4,
      62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32, 24, 16, 8,
      57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3,
      61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47, 39, 31, 23, 15, 7};

   // Tabla de permutación IP a la -1
   static final int[] IP_1 = {
      40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63, 31,
      38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13, 53, 21, 61, 29,
      36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59, 27,
      34, 2, 42, 10, 50, 18, 58, 26, 33, 1, 41, 9, 49, 17, 57, 25};
```

```
// Tabla de Expansión E BIT-SELECTION TABLE
static final int[] E = {
   32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9, 8, 9, 10, 11, 12, 13,
   12, 13, 14, 15, 16, 17, 16, 17, 18, 19, 20, 21, 20, 21, 22,
   23, 24, 25, 24, 25, 26, 27, 28, 29, 28, 29, 30, 31, 32, 1};

// 8 Cajas S Boxes
static final int[][] S = {
   {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
      0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
      4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0,
      15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13
   },
   {15, 1, 8, 14, 6, 11, 3, 2, 9, 7, 2, 13, 12, 0, 5, 10,
      3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5,
      0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15,
      13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9
   },
   {10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
      13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1,
      13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7,
      1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12
   },
   {7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15,
      13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9,
      10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
      3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14
   },
   {2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9,
      14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6,
      4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14,
      11, 8, 12, 7, 1, 14, 2, 12, 6, 15, 0, 9, 10, 4, 5, 3
   },
   {12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11,
      10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8,
      9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6,
      4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13
   },
   {4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1,
      13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6,
      1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
      6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12
   },
   {13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
      1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2,
      7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8,
      2, 1, 14, 7, 4, 10, 18, 13, 15, 12, 9, 0, 3, 5, 6, 11
   }};

// Tabla de permutación P Function
```

```
static final int[] P = {
   16, 7, 20, 21, 29, 12, 28, 17, 1, 15, 23, 26, 5, 18, 31, 10,
   2, 8, 24, 14, 32, 27, 3, 9, 19, 13, 30, 6, 22, 11, 4, 25};

// Permuted Choice 1
static final int[] PC1 = {
   57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18,
   10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36,
   63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22,
   14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4};

// Numero de corrimientos a la izquierda
static final int[] LeftShifts = {
   1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1};

// Permuted Choice 2
static final int[] PC2 = {
   14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10, 23, 19, 12, 4, 26,
   8, 16, 7, 27, 20, 13, 2, 41, 52, 31, 37, 47, 55, 30, 40, 51,
   45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32};

/**
 * Método para generar las Subllaves para cada uno de los tres ciclos
 *
 * @param key1 long llave 1
 * @param key2 long llave 2
 * @param key3 long llave 3
 */
public void setKey(long key1, long key2, long key3) {
   long keyp;
   long b, left, right, t;
   int i, j;
   keyp = 0;
   // Primer ciclo del cálculo de las 16 Subllaves a partir de K1
   for (i = 0; i < 56; i++) {
      b = key1 >> (64 - PC1[i]);
      b &= 1;
      b <<= (55 - i);
      keyp |= b;
   }
   right = keyp & 0xfffffff;
   left = keyp >> 28;
   left &= 0xfffffff;
   // Ciclo para pasar la información sobre LeftShifts
   for (i = 0; i < 16; i++) {
      t = left >> (28 - LeftShifts[i]);
      left <<= LeftShifts[i];
      left |= t;
      left &= 0xfffffff;
      t = right >> (28 - LeftShifts[i]);
      right <<= LeftShifts[i];
```

```
    right |= t;
    right &= 0xffffffff;
    K1[i] = 0;
    keyp = left << 28;
    keyp |= right;
    // Ciclo para pasar la información sobre PC2
    for (j = 0; j < 48; j++) {
        b = keyp >> (56 - PC2[j]);;
        b &= 1;
        b <<= (47 - j);
        K1[i] |= b;
    }
}
keyp = 0;
// Segundo ciclo del cálculo de las 16 Subllaves a partir de K2
for (i = 0; i < 56; i++) {
    b = key2 >> (64 - PC1[i]);
    b &= 1;
    b <<= (55 - i);
    keyp |= b;
}
right = keyp & 0xffffffff;
left = keyp >> 28;
left &= 0xffffffff;
// Ciclo para pasar la información sobre LeftShifts
for (i = 0; i < 16; i++) {
    t = left >> (28 - LeftShifts[i]);
    left <<= LeftShifts[i];
    left |= t;
    left &= 0xffffffff;
    t = right >> (28 - LeftShifts[i]);
    right <<= LeftShifts[i];
    right |= t;
    right &= 0xffffffff;
    K2[i] = 0;
    keyp = left << 28;
    keyp |= right;
    // Ciclo para pasar la información sobre PC2
    for (j = 0; j < 48; j++) {
        b = keyp >> (56 - PC2[j]);;
        b &= 1;
        b <<= (47 - j);
        K2[i] |= b;
    }
}
keyp = 0;
// Tercer ciclo del cálculo de las 16 Subllaves a partir de K3
for (i = 0; i < 56; i++) {
    b = key3 >> (64 - PC1[i]);
    b &= 1;
    b <<= (55 - i);
```

```
            keyp |= b;
        }
        right = keyp & 0xffffffff;
        left = keyp >> 28;
        left &= 0xffffffff;
        // Ciclo para pasar la información sobre LeftShifts
        for (i = 0; i < 16; i++) {
            t = left >> (28 - LeftShifts[i]);
            left <<= LeftShifts[i];
            left |= t;
            left &= 0xffffffff;
            t = right >> (28 - LeftShifts[i]);
            right <<= LeftShifts[i];
            right |= t;
            right &= 0xffffffff;
            K3[i] = 0;
            keyp = left << 28;
            keyp |= right;
            // Ciclo para pasar la información sobre PC2
            for (j = 0; j < 48; j++) {
                b = keyp >> (56 - PC2[j]);
                b &= 1;
                b <<= (47 - j);
                K3[i] |= b;
            }
        }
    }

    /**
     * Método para cifrar
     *
     * @param txt byte Contiene el texto cifrado
     * @return res resultado
     */
    public byte[] cypher(byte txt[]) {
        byte res[] = new byte[8];
        byte txtp[] = new byte[8];
        int i, b, j, k, left, right;
        long lr, rl;
        for (i = 0; i < 8; i++) {
            txtp[i] = 0;
        }
        // Ciclo para pasar la información por la permutación inicial IP
        for (i = 0; i < 64; i++) {
            j = (IP[i] - 1) / 8;
            k = (IP[i] - 1) % 8;
            b = txt[j] >> (7 - k);
            b &= 1;
            j = i / 8;
            k = i % 8;
            b <<= (7 - k);
```

```
            txtp[j] |= b;
        }
        left = right = 0;
        for (i = 0; i < 4; i++) {
            left <<= 8;
            right <<= 8;
            left |= txtp[i] & 0xffl;
            right |= txtp[4 + i] & 0xffl;
        }
        for (i = 0; i < 16; i++) {
            b = f(right, 1, i);
            b ^= left;
            left = right;
            right = b;
        }

        for (i = 15; i >= 0; i--) {
            b = f(left, 2, i);
            b ^= right;
            right = left;
            left = b;
        }
        for (i = 0; i < 16; i++) {
            b = f(right, 3, i);
            b ^= left;
            left = right;
            right = b;
        }
        for (i = 0; i < 8; i++) {
            res[i] = 0;
        }
        lr = (long) right << 32;
        rl = left & 0xffffffffL;
        lr |= rl;
        // Ciclo para pasar la información por la permutación inicial IP_1
        for (i = 0; i < 64; i++) {
            b = (int) (lr >> (64 - IP_1[i]));
            b &= 1;
            j = i / 8;
            k = i % 8;
            b <<= (7 - k);
            res[j] |= b;
        }
        return res; // Se regresa el valor obtenido en res
    }

    /**
     * Método para descifrar
     *
     * @param txt byte Contiene el texto descifrado
     * @return res resultado
```

```
 */
public byte[] decypher(byte txt[]) {
    byte res[] = new byte[8];
    byte txtp[] = new byte[8];
    int i, b, j, k, left, right;
    long lr, rl;
    for (i = 0; i < 8; i++) {
        txtp[i] = 0;
    }
    // Ciclo para pasar la información por la permutación inicial IP
    for (i = 0; i < 64; i++) {
        j = (IP[i] - 1) / 8;
        k = (IP[i] - 1) % 8;
        b = txt[j] >> (7 - k);
        b &= 1;
        j = i / 8;
        k = i % 8;
        b <<= (7 - k);
        txtp[j] |= b;
    }
    left = right = 0;
    for (i = 0; i < 4; i++) {
        left <<= 8;
        right <<= 8;
        left |= txtp[4 + i] & 0xffl;
        right |= txtp[i] & 0xffl;
    }
    for (i = 15; i >= 0; i--) {
        b = f(left, 3, i);
        b ^= right;
        right = left;
        left = b;
    }
    for (i = 0; i < 16; i++) {
        b = f(right, 2, i);
        b ^= left;
        left = right;
        right = b;
    }
    for (i = 15; i >= 0; i--) {
        b = f(left, 1, i);
        b ^= right;
        right = left;
        left = b;
    }
    for (i = 0; i < 8; i++) {
        res[i] = 0;
    }
    lr = left;
    lr <<= 32;
    rl = right & 0xffffffffL;
```

```
        lr |= rl;
        // Ciclo para pasar la información por la permutación inicial IP_1
        for (i = 0; i < 64; i++) {
            b = (int) (lr >> (64 - IP_1[i]));
            b &= 1;
            j = i / 8;
            k = i % 8;
            b <<= (7 - k);
            res[j] |= b;
        }
        return res; // Se regresa el valor obtenido en res
    }

    /**
     * Método que realiza las operaciones de la función f
     *
     * @param data int información después de salir de la permutación E
     * @param key int Contiene las llaves
     * @param ki int Contiene las subllaves
     * @return a resultado de la información al salir de la permutación P
     */
    private int f(int data, int key, int ki) {
        int i, j, r, a;
        long datap, b;
        datap = 0;

        for (i = 0; i < 48; i++) {
            b = data >> (32 - E[i]);
            b &= 1;
            b <<= (47 - i);
            datap |= b;
        }
        // Ejecuta una de las acciones en función de cada una de las tres llaves
        switch (key) {
            case 1:
                datap ^= K1[ki];
                break;
            case 2:
                datap ^= K2[ki];
                break;
            default:
                datap ^= K3[ki];
        }
        r = 0;
        // Ciclos para pasar la información por las SBoxes
        for (i = 0; i < 8; i++) {
            j = (int) (datap >> (42 - i * 6));
            j &= 0x3f;
            a = (j >> 5) & 1;
            a <<= 5;
            b = j & 1;
```

```
            b <<= 4;
            a |= b;
            j >>= 1;
            j &= 0xf;
            j |= a;
            b = S[i][j];
            b <<= (28 - i * 4);
            r |= b;
        }
        a = 0;
        // pasar por la permutación P
        for (i = 0; i < 32; i++) {
            b = r >> (32 - P[i]);
            b &= 1;
            b <<= (31 - i);
            a |= b;
        }
        return a; // Se regresa el valor en a
    }
}
```

Anexo F. Código para el Cifrado-Descifrado de archivos con Triple DES-96.

```
/*
* The TripleDes-96 encryption and decryption files
 * Copyright (C) Filio-Aguilar D. - UAEM and Fores-Carapia R. - CIDETEC
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version 2
 * of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
 */
package mx.com.uaem.des;

/**
 * Clase para cifrar y/o descifrar archivos con TripleDes96
 *
 * @author David Filio
 * @author Rolando Flores
 * @version 2.1.0
 */
public class TripleDes96 {

   // Variables que guardan los valores que toman las llaves
   private long K1[] = new long[16];
   private long K2[] = new long[16];
   private long K3[] = new long[16];

   // // Variables que guardan la permutación variable y su inversa
   private int IP[] = new int[96];
   private int IP_1[] = new int[96];

   // Tabla de permutación generada a partir de los valores del numero euler
   final int[] euler = {71, 82, 81, 82, 84, 59, 4, 52, 35, 36, 02, 87, 47,
      13, 52, 66, 24, 97, 75, 72, 47, 9, 36, 99, 95, 95, 74, 96, 69, 67,
      62, 77, 24, 07, 66, 30, 35, 35, 47, 59, 45, 71, 38, 21, 78, 52, 51,
      66, 42, 74, 27, 46, 63, 91, 93, 20, 03, 05, 99, 21, 81, 74, 13, 59,
      66, 29, 04, 35, 72, 90, 03, 34, 29, 52, 60, 59, 56, 30, 73, 81, 32,
      32, 86, 27, 94, 34, 90, 76, 32, 33, 82, 98, 80, 75, 31, 95
   };
```

```
// Permutacion resultante de combinar E y P
final int[] E = {25, 16, 7, 20, 21, 29, 21, 29, 12, 28, 17, 1,
   17, 1, 15, 23, 26, 5, 26, 5, 18, 31, 10, 2, 10, 2, 8, 24,
   14, 32, 14, 32, 27, 3, 9, 19, 9, 19, 13, 30, 6, 22, 6, 22,
   11, 4, 25, 16};

// 8 Cajas S Boxes
final int[][] S = {
   {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
      0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
      4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0,
      15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13
   },
   {15, 1, 8, 14, 6, 11, 3, 2, 9, 7, 2, 13, 12, 0, 5, 10,
      3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5,
      0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15,
      13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9
   },
   {10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
      13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1,
      13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7,
      1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12
   },
   {7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15,
      13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9,
      10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
      3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14
   },
   {2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9,
      14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6,
      4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14,
      11, 8, 12, 7, 1, 14, 2, 12, 6, 15, 0, 9, 10, 4, 5, 3
   },
   {12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11,
      10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8,
      9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6,
      4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13
   },
   {4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1,
      13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6,
      1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
      6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12
   },
   {13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
      1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2,
      7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8,
      2, 1, 14, 7, 4, 10, 18, 13, 15, 12, 9, 0, 3, 5, 6, 11
   }};
// Permuted Choice 1
final int[] PC1 = {
   57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18,
```

```
    10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36,
    63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22,
    14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4};

// Numero de corrimientos a la izquierda
final int[] LeftShifts = {
    1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1};

// Permuted Choice 2
final int[] PC2 = {
    14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10, 23, 19, 12, 4, 26,
    8, 16, 7, 27, 20, 13, 2, 41, 52, 31, 37, 47, 55, 30, 40, 51,
    45, 33, 48, 44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32};

/**
 * Método para pasar la información por la permutacion euler
 */
private void getPermutation() {
    int c[] = new int[96];
    int X[] = new int[96];
    int i;
    for (i = 0; i < 96; i++) {
        c[i] = euler[i] % (96 - i);
    }
    for (i = 0; i < 96; i++) {
        X[i] = i;
    }
    for (i = 0; i < 96; i++) {
        IP[i] = X[c[i]];
        X[c[i]] = X[95 - i];
    }
    for (i = 0; i < 96; i++) {
        IP_1[IP[i]] = i;
    }

}

/**
 * Método para generar las Subllaves para cada uno de los tres ciclos
 *
 * @param key1 long llave 1
 * @param key2 long llave 2
 * @param key3 long llave 3
 */
public void setKey(long key1, long key2, long key3) {
    long keyp;
    long b, left, right, t;
    int i, j;
    keyp = 0;
    getPermutation();
    // Primer ciclo del cálculo de las 16 Subllaves a partir de K1
```

```
for (i = 0; i < 56; i++) {
    b = key1 >> (64 - PC1[i]);
    b &= 1;
    b <<= (55 - i);
    keyp |= b;
}
right = keyp & 0xfffffff;
left = keyp >> 28;
left &= 0xfffffff;
// Ciclo para pasar la información sobre LeftShifts
for (i = 0; i < 16; i++) {
    t = left >> (28 - LeftShifts[i]);
    left <<= LeftShifts[i];
    left |= t;
    left &= 0xfffffff;
    t = right >> (28 - LeftShifts[i]);
    right <<= LeftShifts[i];
    right |= t;
    right &= 0xfffffff;
    K1[i] = 0;
    keyp = left << 28;
    keyp |= right;
    // Ciclo para pasar la información sobre PC2
    for (j = 0; j < 48; j++) {
        b = keyp >> (56 - PC2[j]);;
        b &= 1;
        b <<= (47 - j);
        K1[i] |= b;
    }
}
keyp = 0;
// Segundo ciclo del cálculo de las 16 Subllaves a partir de K2
for (i = 0; i < 56; i++) {
    b = key2 >> (64 - PC1[i]);
    b &= 1;
    b <<= (55 - i);
    keyp |= b;
}
right = keyp & 0xfffffff;
left = keyp >> 28;
left &= 0xfffffff;
// Ciclo para pasar la información sobre LeftShifts
for (i = 0; i < 16; i++) {
    t = left >> (28 - LeftShifts[i]);
    left <<= LeftShifts[i];
    left |= t;
    left &= 0xfffffff;
    t = right >> (28 - LeftShifts[i]);
    right <<= LeftShifts[i];
    right |= t;
    right &= 0xfffffff;
```

```
            K2[i] = 0;
            keyp = left << 28;
            keyp |= right;
            // Ciclo para pasar la información sobre PC2
            for (j = 0; j < 48; j++) {
                b = keyp >> (56 - PC2[j]);;
                b &= 1;
                b <<= (47 - j);
                K2[i] |= b;
            }
        }
        keyp = 0;
        // Tercer ciclo del cálculo de las 16 Subllaves a partir de K3
        for (i = 0; i < 56; i++) {
            b = key3 >> (64 - PC1[i]);
            b &= 1;
            b <<= (55 - i);
            keyp |= b;
        }
        right = keyp & 0xfffffff;
        left = keyp >> 28;
        left &= 0xfffffff;
        // Ciclo para pasar la información sobre LeftShifts
        for (i = 0; i < 16; i++) {
            t = left >> (28 - LeftShifts[i]);
            left <<= LeftShifts[i];
            left |= t;
            left &= 0xfffffff;
            t = right >> (28 - LeftShifts[i]);
            right <<= LeftShifts[i];
            right |= t;
            right &= 0xfffffff;
            K3[i] = 0;
            keyp = left << 28;
            keyp |= right;
            // Ciclo para pasar la información sobre PC2
            for (j = 0; j < 48; j++) {
                b = keyp >> (56 - PC2[j]);
                b &= 1;
                b <<= (47 - j);
                K3[i] |= b;
            }
        }
    }

    /**
     * Método para cifrar
     *
     * @param txt byte Contiene el texto cifrado
     * @return res resultado
     */
```

```
public byte[] cypher(byte txt[]) {
   byte res[] = new byte[12];
   byte txtp[] = new byte[12];
   int i, ii, j, k;
   long b, left, right;
   long lr[] = new long[2];
   long tlr[] = new long[2];
   for (i = 0; i < 12; i++) {
      txtp[i] = txt[i];
   }
   left = right = 0;
   for (i = 0; i < 6; i++) {
      left <<= 8;
      right <<= 8;
      left |= txtp[i] & 0xffl;
      right |= txtp[6 + i] & 0xffl;
   }
   // Ciclo para pasar la información por la permutación inicial IP
   for (i = 0; i < 16; i++) {
      if (i == 2) {
         lr[0] = left;
         lr[1] = right;
         tlr[0] = 0;
         tlr[1] = 0;
         for (ii = 0; ii < 96; ii++) {
            j = (IP[ii]) / 48;
            k = (IP[ii]) % 48;
            b = lr[j] >> (47 - k);
            b &= 1;
            j = ii / 48;
            k = ii % 48;
            b <<= (47 - k);
            tlr[j] |= b;
         }
         left = tlr[0];
         right = tlr[1];
      }
      b = f(right, 1, i);
      b ^= left;
      left = right;
      right = b;
   }

   for (i = 15; i >= 0; i--) {
      b = f(left, 2, i);
      b ^= right;
      right = left;
      left = b;

   }
```

```
    for (i = 0; i < 16; i++) {
        b = f(right, 3, i);
        b ^= left;
        left = right;
        right = b;
        if (i == 13) {
            lr[0] = left;
            lr[1] = right;
            tlr[0] = 0;
            tlr[1] = 0;
            // Ciclo para pasar la información por la permutación inicial IP_1
            for (ii = 0; ii < 96; ii++) {
                j = (IP_1[ii]) / 48;
                k = (IP_1[ii]) % 48;
                b = lr[j] >> (47 - k);
                b &= 1;
                j = ii / 48;
                k = ii % 48;
                b <<= (47 - k);
                tlr[j] |= b;
            }
            left = tlr[0];
            right = tlr[1];
        }
    }
    for (i = 0; i < 12; i++) {
        res[i] = 0;
    }
    for (i = 0; i < 6; i++) {
        res[i] = (byte) ((right >> (40 - i * 8)) & 0xffl);
    }
    for (i = 0; i < 6; i++) {
        res[i + 6] = (byte) ((left >> (40 - i * 8)) & 0xffl);
    }
    return res; // Se regresa el valor obtenido en res
}

/**
 * método para Descifrar
 *
 * @param txt byte
 * @return res resultado
 */
public byte[] decypher(byte txt[]) {

    byte res[] = new byte[12];
    byte txtp[] = new byte[12];
    int i, ii, j, k;
    long b, left, right;
    long lr[] = new long[2];
    long tlr[] = new long[2];
```

```
for (i = 0; i < 12; i++) {
   txtp[i] = txt[i];
}

left = right = 0;
for (i = 0; i < 6; i++) {
   left <<= 8;
   right <<= 8;
   left |= txtp[6 + i] & 0xffl;
   right |= txtp[i] & 0xffl;
}
// Ciclo para pasar la información por la permutación inicial IP
for (i = 15; i >= 0; i--) {
   if (i == 13) {
      lr[0] = left;
      lr[1] = right;
      tlr[0] = 0;
      tlr[1] = 0;
      for (ii = 0; ii < 96; ii++) {
         j = (IP[ii]) / 48;
         k = (IP[ii]) % 48;
         b = lr[j] >> (47 - k);
         b &= 1;
         j = ii / 48;
         k = ii % 48;
         b <<= (47 - k);
         tlr[j] |= b;
      }
      left = tlr[0];
      right = tlr[1];
   }
   b = f(left, 3, i);
   b ^= right;
   right = left;
   left = b;

}
for (i = 0; i < 16; i++) {
   b = f(right, 2, i);
   b ^= left;
   left = right;
   right = b;

}
for (i = 15; i >= 0; i--) {
   b = f(left, 1, i);
   b ^= right;
   right = left;
   left = b;
   if (i == 2) {
```

```
            lr[0] = left;
            lr[1] = right;
            tlr[0] = 0;
            tlr[1] = 0;
            // Ciclo para pasar la información por la permutación inicial IP_1
            for (ii = 0; ii < 96; ii++) {
               j = (IP_1[ii]) / 48;
               k = (IP_1[ii]) % 48;
               b = lr[j] >> (47 - k);
               b &= 1;
               j = ii / 48;
               k = ii % 48;
               b <<= (47 - k);
               tlr[j] |= b;
            }
            left = tlr[0];
            right = tlr[1];
         }

      }

      for (i = 0; i < 6; i++) {
         res[i] = (byte) ((left >> (40 - i * 8)) & 0xffl);
      }
      for (i = 0; i < 6; i++) {
         res[i + 6] = (byte) ((right >> (40 - i * 8)) & 0xffl);
      }
      return res; // Se regresa el valor obtenido en res
   }

   /**
    * Método que realiza las operaciones de la función f
    *
    * @param data long información después de salir de la permutación E
    * @param key int Contiene las llaves
    * @param ki int Contiene las subllaves
    * @return datap resultado de la información al salir de la permutación P
    */
   private long f(long data, int key, int ki) {
      int i, j, r, a;
      long datap, b;
      datap = data;
      // Ejecuta una de las acciones en función de cada una de las tres llaves
      switch (key) {
         case 1:
            datap ^= K1[ki];
            break;
         case 2:
            datap ^= K2[ki];
            break;
         default:
```

```
                datap ^= K3[ki];
            }
            r = 0;
            // Ciclos para pasar la información por las SBoxes
            for (i = 0; i < 8; i++) {
                j = (int) (datap >> (42 - i * 6));
                j &= 0x3f;
                a = (j >> 5) & 1;
                a <<= 5;
                b = j & 1;
                b <<= 4;
                a |= b;
                j >>= 1;
                j &= 0xf;
                j |= a;
                b = S[i][j];
                b <<= (28 - i * 4);
                r |= b;
            }
            datap = 0;
            // pasar por la permutación E
            for (i = 0; i < 48; i++) {
                b = r >> (32 - E[i]);
                b &= 1;
                b <<= (47 - i);
                datap |= b;
            }
            return datap;   //Se regresa el valor en datap
        }

        /**
         * Método para convertir a hexadecimal
         *
         * @param x long texto en hexadecimal
         * @return r
         */
        public String hex(long x) {
            String r = "";
            long b;
            int i;
            for (i = 0; i < 64; i += 4) {
                b = x >> (60 - i);
                b &= 0xf;
                r += (char) (b < 10 ? b + 48 : b + 55);
            }
            return r;
        }
    }
```

Anexo G. Código de la clase que ejecuta a Triple DES y a Triple DES-96.

```
/*
 * Main class that encrypts and decrypts files with TripleDes y TripleDes96
 * Copyright (C) Filio-Aguilar D. - UAEM and Fores-Carapia R. - CIDETEC
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License
 * as published by the Free Software Foundation; either version 3
 * of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along withthis program. If not, see <http://www.gnu.org/licenses/>.
 */
package mx.com.uaem.des;

//importaciones
import java.applet.Applet;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import javax.swing.JFileChooser;

/**
 * Clase principal que ejecuta Triple DES y Triple DE-96 para cifrar archivos
 *
 * @author David Filio
 * @author Rolando Flores
 * @version 2.1.0
 */
public class principal extends Applet {

   private JFileChooser fc;

   // Se iniicaliza el selector de archivos
   public void init() {
      this.setSize(600, 400);
      fc = new JFileChooser();
      try {
         java.awt.EventQueue.invokeAndWait(new Runnable() {
            public void run() {
               initComponents();
```

```
            }
        });
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

/**
 * This method is called from within the init() method to initialize the
 * form. WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jButton1 = new javax.swing.JButton();
    jButton2 = new javax.swing.JButton();
    jButton3 = new javax.swing.JButton();
    jButton4 = new javax.swing.JButton();

    jButton1.setText("Cifrar");
    jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
        }
    });

    jButton2.setText("Descifrar");
    jButton2.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton2ActionPerformed(evt);
        }
    });

    jButton3.setText("Cifrar 96");
    jButton3.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton3ActionPerformed(evt);
        }
    });

    jButton4.setText("Descifrar 96");
    jButton4.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton4ActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
    this.setLayout(layout);
    layout.setHorizontalGroup(
```

```
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(157, 157, 157)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 75,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton3))
                .addGap(119, 119, 119)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                    .addComponent(jButton4, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jButton2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap(235, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap(354, Short.MAX_VALUE)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton1)
                    .addComponent(jButton2))
                .addGap(48, 48, 48)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton4)
                    .addComponent(jButton3))
                .addGap(37, 37, 37))
        );
    }// </editor-fold>
    /**
     * Boton1, si se presiona abre el explorador y cifra con TripleDes el
     * archivo seleccionado
     *
     * @param evt ActionEvent
     */
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String a = "";
        byte[] bin;
        int len = 8;
        bin = new byte[len];
        if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {
            try {
                File file = fc.getSelectedFile();
                len = (int) file.length();
                bin = new byte[len];
                FileInputStream ois = new FileInputStream(file);
                BufferedInputStream bfs = new BufferedInputStream(ois);
                bfs.read(bin); // Lee el archivo
                bfs.close();
                // Se le agrega al achivo la extencion .enc para diferenciarlo
```

```
            a = file.getPath() + ".enc";
            System.out.println(a);
        } catch (IOException e) {
            System.out.println("error: " + e.toString());
        }
        // Se agregan los datos para cifrar el archivo
        int i, j;
        long key1, key2, key3;
        byte bk[] = new byte[8];
        // Se guarda en la variable ti el tiempo en que inicia el proceso
        // de cifrado
        long ti = System.currentTimeMillis();
        TripleDes tdes = new TripleDes(); //Se crea un objeto tipo TripleDes
        key1 = 0x133457799bbcdff1L; // Se ingresa el valor de la llave 1
        key2 = 0xfedcba9876543210L; // Se ingresa el valor de la llave 2
        key3 = 0x9876542310fedcbaL; // Se ingresa el valor de la llave 3
        tdes.setKey(key1, key2, key3); // Se pasan a tdes las tres llaves
        len = len - len % 8;
        for (i = 0; i < len; i += 8) {
            for (j = 0; j < 8; j++) {
                bk[j] = bin[i + j];
            }
            bk = tdes.cypher(bk);
            for (j = 0; j < 8; j++) {
                bin[i + j] = bk[j];
            }
        }
        // Se imprime en consola el tiempo actual menos el valor de ti
        // para obtener el tiempo que tardo en cifrar
        System.out.println("Time is: " + (System.currentTimeMillis() - ti));
        try {
            File file1 = new File(a);
            FileOutputStream os = new FileOutputStream(file1);
            BufferedOutputStream bos = new BufferedOutputStream(os);
            bos.write(bin); // Se escribe el archivo cifrado
            bos.close();
        } catch (IOException e) {
            System.out.println(e);
        }
    }
    System.out.println("Termino..."); //Imprime en consola el fin del proceso
}
/**
 * Boton2 si se presiona abre el explorador y descifra con TripleDes el
 * archivo seleccionado
 *
 * @param evt ActionEvent
 */
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    String a = "";
    byte[] bin;
```

```
int len = 8;
bin = new byte[len];
if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {
   try {
      File file = fc.getSelectedFile();
      len = (int) file.length();
      bin = new byte[len];
      FileInputStream ois = new FileInputStream(file);
      BufferedInputStream bfs = new BufferedInputStream(ois);
      bfs.read(bin); // Lee el archivo
      bfs.close();
      a = file.getPath();
      a = a.substring(0, a.length() - 4);
      System.out.println(a);
   } catch (IOException ex) {
      System.out.println(ex);
   } catch (Exception ex) {
      System.out.println(ex);
   }
   // Se agregan los datos para descifrar el archivo
   int i, j;
   long key1, key2, key3;
   byte bk[] = new byte[8];
   // Se guarda en la variable ti el tiempo en que inicia el proceso
   // de descifrado
   long ti = System.currentTimeMillis();
   TripleDes tdes = new TripleDes(); //Se crea un objeto tipo TripleDes
   key1 = 0x133457799bbcdff1L; // Se ingresa el valor de la llave 1
   key2 = 0xfedcba9876543210L; // Se ingresa el valor de la llave 2
   key3 = 0x9876542310fedcbaL; // Se ingresa el valor de la llave 3
   tdes.setKey(key1, key2, key3);  // Se pasan a tdes las tres llaves
   len = len - len % 8;
   for (i = 0; i < len; i += 8) {
      for (j = 0; j < 8; j++) {
         bk[j] = bin[i + j];
      }
      bk = tdes.decypher(bk);
      for (j = 0; j < 8; j++) {
         bin[i + j] = bk[j];
      }
   }
   // Se imprime en consola el tiempo actual menos el valor de ti
   // para obtener el tiempo que tardo en descifrar
   System.out.println("Time is: " + (System.currentTimeMillis() - ti));
   try {
      File file1 = new File(a);
      FileOutputStream os = new FileOutputStream(file1);
      BufferedOutputStream bos = new BufferedOutputStream(os);
      bos.write(bin); // Se escribe el archivo descifrado
      bos.close();
   } catch (IOException e) {
```

```
            System.out.println(e);
        }
    }
    System.out.println("Termino..."); //Imprime en consola el fin del proceso
}
/**
 * Boton3 si se presiona abre el explorador y cifra con TripleDes-96 el
 * archivo seleccionado
 *
 * @param evt ActionEvent
 */
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    String a = "";
    byte[] bin;
    int len = 8;
    bin = new byte[len];
    if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {
        try {
            File file = fc.getSelectedFile();
            len = (int) file.length();
            bin = new byte[len];
            FileInputStream ois = new FileInputStream(file);
            try (BufferedInputStream bfs = new BufferedInputStream(ois)) {
                bfs.read(bin); // Lee el archivo
                bfs.close();
            }
            // Se le agrega al achivo la extencion .enc para diferenciarlo
            a = file.getPath() + ".enc";
            System.out.println(a);
        } catch (IOException e) {
            System.out.println("error: " + e.toString());
        }
        // Datos para cifrar el archivo
        int i, j;
        long key1, key2, key3;
        byte bk[] = new byte[12];
        // Se guarda en la variable ti el tiempo en que
        // inicia el proceso de cifrado
        long ti = System.currentTimeMillis();
        TripleDes96 tdes = new TripleDes96();//Se crea un objeto tipo TripleDes-96
        key1 = 0x133457799bbcdff1L; // Se ingresa el valor de la llave 1
        key2 = 0xfedcba9876543210L; // Se ingresa el valor de la llave 2
        key3 = 0x9876542310fedcbaL; // Se ingresa el valor de la llave 3
        tdes.setKey(key1, key2, key3); // Se pasan a tdes las tres llaves
        len = len - len % 12;
        for (i = 0; i < len; i += 12) {
            for (j = 0; j < 12; j++) {
                bk[j] = bin[i + j];
            }
            bk = tdes.cypher(bk);
            for (j = 0; j < 12; j++) {
```

```
                bin[i + j] = bk[j];
            }
        }
        // Se imprime en consola el tiempo actual menos el valor de ti
        // para obtener el tiempo que tardo en cifrar
        System.out.println("Time is: " + (System.currentTimeMillis() - ti));
        try {
            File file1 = new File(a);
            FileOutputStream os = new FileOutputStream(file1);
            BufferedOutputStream bos = new BufferedOutputStream(os);
            bos.write(bin); // Se escribe el archivo cifrado
            bos.close();
        } catch (IOException e) {
            System.out.println(e);
        }
    }
    System.out.println("Termino..."); //Imprime en consola el fin del proceso
}
/**
 * Boton4 al presionarlo abre el explorador y descifra con TripleDes el
 * archivo seleccionado
 *
 * @param evt ActionEvent
 */
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    String a = "";
    byte[] bin;
    int len = 8;
    bin = new byte[len];
    if (fc.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {
        try {
            File file = fc.getSelectedFile();
            len = (int) file.length();
            bin = new byte[len];
            FileInputStream ois = new FileInputStream(file);
            BufferedInputStream bfs = new BufferedInputStream(ois);
            bfs.read(bin); // Lee el archivo
            bfs.close();
            a = file.getPath();
            a = a.substring(0, a.length() - 4);
            System.out.println(a);
        } catch (IOException ex) {
            System.out.println(ex);
        } catch (Exception ex) {
            System.out.println(ex);
        }
        // Se agregan los datos para descifrar el archivo
        int i, j;
        long key1, key2, key3;
        byte bk[] = new byte[12];
        // Se guarda en la variable ti el tiempo en que inicia el proceso
```

```
            // de descifrado
            long ti = System.currentTimeMillis();
            TripleDes96 tdes = new TripleDes96(); //Se crea un objeto tipo TripleDes
            key1 = 0x133457799bbcdff1L; // Se ingresa el valor de la llave 1
            key2 = 0xfedcba9876543210L; // Se ingresa el valor de la llave 2
            key3 = 0x9876542310fedcbaL; // Se ingresa el valor de la llave 3
            tdes.setKey(key1, key2, key3);  // Se pasan a tdes las tres llaves
            len = len - len % 12;
            for (i = 0; i < len; i += 12) {
               for (j = 0; j < 12; j++) {
                  bk[j] = bin[i + j];
               }
               bk = tdes.decypher(bk);
               for (j = 0; j < 12; j++) {
                  bin[i + j] = bk[j];
               }
            }
            // Se imprime en consola el tiempo actual menos el valor de ti
            // para obtener el tiempo que tardo en descifrar
            System.out.println("Time is: " + (System.currentTimeMillis() - ti));
            try {
               File file1 = new File(a);
               FileOutputStream os = new FileOutputStream(file1);
               BufferedOutputStream bos = new BufferedOutputStream(os);
               bos.write(bin); // Se escribe el archivo descifrado
               bos.close();
            } catch (IOException e) {
               System.out.println(e);
            }
         }
         System.out.println("Termino..."); //Imprime en consola el fin del proceso
      }

      // Variables declaration - do not modify
      private javax.swing.JButton jButton1;
      private javax.swing.JButton jButton2;
      private javax.swing.JButton jButton3;
      private javax.swing.JButton jButton4;
      // End of variables declaration
   }
```