ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

NUEVO ALGORITMO CRIPTOGRÁFICO CON LA INCORPORACIÓN DE LA ESTEGANOGRAFÍA EN IMÁGENES

AUTOR: PABLO MARTÍ MÉNDEZ NARANJO

Proyecto de investigación, presentado ante el Instituto de Posgrado y Educación Continua de la ESPOCH, como requisito parcial para la obtención del grado de

MAGISTER EN SEGURIDAD TELEMÁTICA

RIOBAMBA - ECUADOR

OCTUBRE 2015

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

CERTIFICACIÓN:

El Tribunal del PROYECTO DE INVESTIGACIÓN CERTIFICA QUE:

El trabajo de titulación titulado "NUEVO ALGORITMO CRIPTOGRÁFICO CON LA INCORPORACIÓN DE LA ESTEGANOGRAFÍA EN IMÁGENES", de responsabilidad del Ing. Pablo Martí Méndez Naranjo, ha sido prolijamente revisado y se autoriza su presentación.

Tribunal:

Dr. Juan Mario Vargas Guambo	
PRESIDENTE	FIRMA
Ing. Jorge Ernesto Huilca Palacios	
DIRECTOR	FIRMA
Ing. Andrés Santiago Cisneros Barahona	
MIEMBRO	FIRMA
Ing. Paúl Patricio Romero Riera	
MIEMBRO	FIRMA
DOCUMENTALISTA SISBIB ESPOCH	FIRMA

Riobamba, Octubre 2015

DERECHOS INTELECTUALES

Yo, Pablo Martí Méndez Naranjo, declaro que soy responsable de las ideas, doctrinas y resultados expuestos en el presente Proyecto de Investigación, y que el patrimonio intelectual generado por la misma pertenece exclusivamente a la Escuela Superior Politécnica de Chimborazo.

060333501-9

DEDICATORIA

Este trabajo está dedicado a las personas que a lo largo de mi vida me han proporcionado una orientación personal y profesional.

La fe, el esfuerzo y optimismo dedicado a lo largo de los años de estudio, son el fruto de quienes creyeron en mí, apoyándome en todo sentido extendiendo la mano a través de la educación.

Con mucho cariño a mis padres, quienes me brindaron su apoyo moral y económico en cada etapa de mi vida.

Pablo Martí

AGRADECIMIENTO

Agradezco a Dios por su protección.

A mis padres por guiarme día a día y permitirme cumplir mis anhelos.

A la Escuela Superior Politécnica de Chimborazo, Institución que contribuyó a mi formación.

A las autoridades y docentes, en especial al Director y los Miembros del Tribunal del Trabajo de Titulación que me orientaron hasta alcanzar esta meta.

A mi familia y amigos por su apoyo.

Pablo Martí

ÍNDICE DE CONTENIDO

LISTA DE TABLAS xi

LISTA DE FIGURAS xiv

LISTA DE ANEXOS xxii

RESUMEN xxiii

SUMMARY xxiv

CAPÍTULO I

INTRODUCCIÓN

1.1. **Problema de investigación** 1

1.1.1. ***Planteamiento del problema*** 1

1.1.2. ***Formulación del problema*** 3

1.1.3. ***Sistematización del problema*** 4

1.2. **Justificación de la investigación** 4

1.2.1. ***Justificación teórica*** 4

1.2.2. ***Justificación metodológica*** 4

1.2.3. ***Justificación práctica*** 5

1.3. **Objetivos** 5

1.3.1. ***Objetivo general*** 5

1.3.2. ***Objetivos específicos*** 5

1.4. **Hipótesis** 5

1.5. **Estructura del proyecto de investigación** 6

CAPÍTULO II

MARCO DE REFERENCIA

2.1. **Estado del arte** 7

2.2. **Criptografía** 13

2.2.1. ***Antecedentes*** 13

2.2.1.1. *Criptología* 13

2.2.2. ***Tipos de criptografía*** 15

2.2.2.1. *Criptografía clásica* 15

2.2.2.2. *Criptografía moderna* 16

2.2.3. ***Determinación del algoritmo criptográfico base*** 35

2.3. **Esteganografía** 38

2.3.1. ***Antecedentes*** 38

2.3.2. ***Técnicas esteganográficas en imágenes*** 39

2.3.2.1. *Enmascaramiento y filtrado* 40

2.3.2.2. *Inserción de bits en el objeto contenedor* 40

2.3.2.3. *Creación de un fichero contenedor propio partiendo de la información a ocultar* 40

2.3.2.4. *Bit Menos Significativo (Least Significant Bit - LSB)* 40

2.3.3. ***Determinación de la técnica esteganográfica en imágenes*** 42

2.4. **Análisis y diseño de algoritmos** 42

CAPÍTULO III

DISEÑO DE INVESTIGACIÓN

3.1. **Tipo de investigación** 46

3.2. **Diseño de la investigación** 46

3.3. **Métodos y técnicas** 46

3.3.1. ***Métodos*** 47

3.3.2. ***Técnicas*** 47

3.4. **Instrumentos** 47

3.5. **Validación de instrumentos** 48

3.6. **Implementación del algoritmo criptográfico base** 50

3.6.1. ***Desarrollo de la aplicación*** 50

3.6.2. ***Proceso de cifrado*** 54

3.6.3. ***Proceso de descifrado*** 57

3.6.4. ***Validación del algoritmo*** 59

3.6.4.1. *Desarrollo de la aplicación* 60

3.7. **Creación e implementación del nuevo algoritmo criptográfico** 65

3.7.1. ***Propuesta de mejora*** 65

3.7.2. ***Desarrollo de la aplicación*** 68

3.7.3. ***Nuevo proceso de cifrado*** 71

3.7.4. ***Nuevo proceso de descifrado*** 74

3.7.5. ***Análisis y diseño de algoritmos*** 77

3.8. **Implementación del algoritmo esteganográfico en imágenes** 79

3.8.1. ***Desarrollo de la aplicación*** 79

3.8.2. ***Proceso de embebido*** 83

3.8.3. ***Proceso de extracción*** 85

3.9. **Integración de los algoritmos criptográficos y esteganográficos** 87

3.9.1. ***Prototipo I*** 88

3.9.1.1. *Desarrollo de la aplicación* 88

3.9.1.2. *Cifrado y embebido* 93

3.9.1.3. *Extracción y descifrado* 94

3.9.2. ***Prototipo II*** 95

3.9.2.1. *Desarrollo de la aplicación* 95

3.9.2.2. *Cifrado y embebido* 100

3.9.2.3. *Extracción y descifrado* 101

3.10. **Definición de los escenarios de pruebas** 102

3.11. **Hipótesis** 103

3.11.1. *Determinación de variables* 103

3.11.2. *Operacionalización conceptual* 103

3.11.3. *Operacionalización metodológica* 103

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. **Desarrollo de las pruebas** 105

4.1.1. ***Prototipo I*** 105
4.1.1.1. *Cifrado y embebido* 105
4.1.1.2. *Extracción y descifrado* 110
4.1.2. ***Prototipo II*** 114
4.1.2.1. *Cifrado y embebido* 114
4.1.2.2. *Extracción y descifrado* 118
4.2. **Análisis y comparación de resultados** 122
4.2.1. ***Comparación de resultados criptográficos*** 122
4.2.1.1. *Clave de 128 bits* 122
4.2.1.2. *Clave de 192 bits* 123
4.2.1.3. *Clave de 256 bits* 123
4.2.2. ***Comparación de resultados esteganográficos*** 123
4.2.2.1. *Prototipo I* 123
4.2.2.2. *Prototipo II* 132
4.3. **Prueba de hipótesis** 141
4.3.1. ***Pruebas*** 141
4.3.1.1. *Análisis de características de los algoritmos* 141
4.3.1.2. *Criptoanálisis de los mensajes cifrados de los algoritmos* 145
4.3.2. ***Escalas de calificación*** 164
4.3.2.1. *Indicador 1: No. claves utilizadas* 164
4.3.2.2. *Indicador 2: No. Rondas* 165
4.3.2.3. *Indicador 3: No. funciones usadas por el algoritmo* 165
4.3.2.4. *Indicador 4: No. funciones ejecutadas por el algoritmo* 166
4.3.2.5. *Indicador 5: Entropía* 166
4.3.2.6. *Indicador 6: Histograma* 166
4.3.2.7. *Indicador 7: N-gramas* 167
4.3.2.8. *Indicador 8: Autocorrelación* 167
4.3.2.9. *Indicador 9: Fuerza bruta* 168
4.3.3. ***Ponderación de indicadores*** 168

4.3.3.1. *Indicador 1: No. Claves utilizadas* 169

4.3.3.2. *Indicador 2: No. Rondas* 169

4.3.3.3. *Indicador 3: No. funciones usadas por el algoritmo* 170

4.3.3.4. *Indicador 4: No. funciones ejecutadas por el algoritmo* 171

4.3.3.5. *Indicador 5: Entropía* 172

4.3.3.6. *Indicador 6: Histogramas* 173

4.3.3.7. *Indicador 7: N-gramas* 174

4.3.3.8. *Indicador 8: Autocorrelación* 175

4.3.3.9. *Indicador 9: Fuerza bruta* 176

4.3.4. ***Comprobación de hipótesis*** 177

4.3.4.1. *Estadística descriptiva* 177

4.3.4.2. *Estadística inferencial* 179

CONCLUSIONES 185

RECOMENDACIONES 187

BIBLIOGRAFÍA

ANEXOS

LISTA DE TABLAS

Tabla 1-2: Resultados obtenidos de la correlación entre la adyacencia de los pixeles 9
Tabla 2-2: Resultados obtenidos de la comparación de imágenes 11
Tabla 3-2: Resultados obtenidos de la comparación de imágenes y tamaños 12
Tabla 4-2: Ventajas y desventajas de la criptografía asimétrica 17
Tabla 5-2: Ventajas y desventajas de la criptografía simétrica 19
Tabla 6-2: Relación entre longitud de clave, tamaño de bloque y rondas 22
Tabla 7-2: Ventajas y desventajas del algoritmo DES 31
Tabla 8-2: Tabla comparativa entre algoritmos de criptografía simétrica 37

Tabla 1-3: Datos para ejecución de la aplicación desarrollada para cifrado 55
Tabla 2-3: Datos para ejecución de la aplicación desarrollada para descifrado 58
Tabla 3-3: Datos para ejecución de la aplicación con librerías definidas para cifrado 62
Tabla 4-3: Datos para ejecución de la aplicación con librerías definidas para descifrado 63
Tabla 5-3: Datos para ejecución de la aplicación desarrollada para cifrado 2NAES 72
Tabla 6-3: Datos para ejecución de la aplicación desarrollada para descifrado 2NAES 75
Tabla 7-3: Datos para ejecución de la aplicación desarrollada para embebido 83
Tabla 8-3: Datos para ejecución de la aplicación desarrollada para extracción 85
Tabla 9-3: Operacionalización conceptual 103
Tabla 10-3: Operacionalización metodológica 104

Tabla 1-4: Datos para ejecución de cifrado y embebido con el Prototipo I con clave de 128 bits 105
Tabla 2-4: Mensaje cifrado y embebido con el Prototipo I con clave de 128 bits 106
Tabla 3-4: Datos para ejecución del Prototipo I con clave de 192 bits 107
Tabla 4-4: Mensaje cifrado y embebido con el Prototipo I con clave de 192 bits 107
Tabla 5-4: Datos para ejecución del Prototipo I con clave de 256 bits 108
Tabla 6-4: Mensaje cifrado y embebido con el Prototipo I con clave de 256 bits 109
Tabla 7-4: Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 128 bits 110
Tabla 8-4: Mensaje extraído y descifrado con el Prototipo I con clave de 128 bits 110
Tabla 9-4: Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 192 bits 111
Tabla 10-4: Mensaje extraído y descifrado con el Prototipo I con clave de 192 bits 112
Tabla 11-4: Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 256 bits 112
Tabla 12-4: Mensaje extraído y descifrado con el Prototipo I con clave de 256 bits 113

Tabla 13-4: Datos para ejecución del Prototipo II con clave de 128 bits..... 114
Tabla 14-4: Mensaje cifrado y embebido con el Prototipo II con clave de 128 bits..... 115
Tabla 15-4: Datos para ejecución del Prototipo II con clave de 192 bits..... 115
Tabla 16-4: Mensaje cifrado y embebido con el Prototipo II con clave de 192 bits..... 116
Tabla 17-4: Datos para ejecución del Prototipo II con clave de 256 bits..... 117
Tabla 18-4: Mensaje cifrado y embebido con el Prototipo II con clave de 256 bits..... 117
Tabla 19-4: Datos para ejecución del Prototipo II con clave de 128 bits..... 118
Tabla 20-4: Mensaje extraído y descifrado con el Prototipo II con clave de 128 bits 119
Tabla 21-4: Datos para ejecución del Prototipo II con clave de 192 bits..... 119
Tabla 22-4: Mensaje extraído y descifrado con el Prototipo II con clave de 192 bits 120
Tabla 23-4: Datos para ejecución del Prototipo II con clave de 256 bits..... 121
Tabla 24-4: Mensaje extraído y descifrado con el Prototipo II con clave de 256 bits 121
Tabla 25-4: Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 128 bits 122
Tabla 26-4: Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 192 bits 123
Tabla 27-4: Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 256 bits 123
Tabla 28-4: Tamaños de las imágenes con el Prototipo I con clave de 128 bits..... 124
Tabla 29-4: Tamaños de las imágenes con el Prototipo I con clave de 192 bits..... 127
Tabla 30-4: Tamaños de las imágenes con el Prototipo I con clave de 256 bits..... 129
Tabla 31-4: Tamaños de las imágenes con el Prototipo II con clave de 128 bits 132
Tabla 32-4: Tamaños de las imágenes con el Prototipo II con clave de 192 bits 135
Tabla 33-4: Tamaños de las imágenes con el Prototipo II con clave de 256 bits 138
Tabla 34-4: Determinación de los indicadores definidos para comparar los algoritmos 142
Tabla 35-4: Resultados de comparación de indicadores..... 144
Tabla 36-4: Valores promedio de resultados de los indicadores..... 144
Tabla 37-4: Datos para realizar las pruebas..... 146
Tabla 38-4: Caracteres diferentes y valores de entropía de mensajes cifrados..... 150
Tabla 39-4: Valor promedio de resultados del Indicador entropía..... 150
Tabla 40-4: Número de caracteres del histograma de los mensajes cifrados..... 153
Tabla 41-4: Valor promedio de resultados del Indicador histograma..... 154
Tabla 42-4: Detalle de n-gramas de los mensajes cifrados..... 156
Tabla 43-4: Resumen de n-gramas de los mensajes cifrados..... 156
Tabla 44-4: Valor promedio de resultados del Indicador n-gramas máximo..... 157
Tabla 45-4: Número de caracteres que concuerdan en la correlación de los mensajes cifrados..... 160

Tabla 46-4: Valor promedio de resultados del Indicador autocorrelación 160
Tabla 47-4: Número de caracteres del histograma de los mensajes cifrados 163
Tabla 48-4: Valor promedio de resultados del Indicador fuerza bruta 164
Tabla 49-4: Tabla de escalas para el Indicador 1: No. claves utilizadas 164
Tabla 50-4: Tabla de escalas para el Indicador 2: No. rondas 165
Tabla 51-4: Tabla de escalas para el Indicador 3: No. funciones usadas por el algoritmo ... 165
Tabla 52-4: Tabla de escalas para el Indicador 4: No. funciones ejecutadas por el algoritmo 166
Tabla 53-4: Tabla de escalas para el Indicador 5: Entropía 166
Tabla 54-4: Tabla de escalas para el Indicador 6: Histogramas 167
Tabla 55-4: Tabla de escalas para el Indicador 7: N-gramas 167
Tabla 56-4: Tabla de escalas para el Indicador 8: Autocorrelación 168
Tabla 57-4: Tabla de escalas para el Indicador 9: Fuerza bruta 168
Tabla 58-4: Códigos del Indicador 1: No. Claves utilizadas 169
Tabla 59-4: Códigos del Indicador 2: No. Rondas 170
Tabla 60-4: Códigos del Indicador 3: No. Funciones utilizadas por el algoritmo 170
Tabla 61-4: Códigos del Indicador 4: No. Funciones ejecutadas por el algoritmo 171
Tabla 62-4: Códigos del Indicador 5: Entropía 172
Tabla 63-4: Códigos del Indicador 6: Histogramas 173
Tabla 64-4: Códigos del Indicador 7: N-gramas 174
Tabla 65-4: Códigos del Indicador 8: Autocorrelación 175
Tabla 66-4: Códigos del Indicador 9: Fuerza Bruta 176
Tabla 67-4: Resultados de indicadores 178
Tabla 68-4: Tabla de contingencia de frecuencias observadas 180
Tabla 69-4: Tabla de contingencia de frecuencias esperadas 181
Tabla 70-4: Cálculo de X^2 182
Tabla 71-4: Tabla de distribución de X^2 183

LISTA DE FIGURAS

Figura 1-1: Proceso de la criptografía 2
Figura 2-1: Proceso de la esteganografía 2
Figura 1-2: Propuesta del esquema híbrido 8
Figura 2-2: Propuesta de algoritmo para embeber la información 10
Figura 3-2: Propuesta de algoritmo para extraer la información 10
Figura 4-2: Propuesta del modelo esteganográfico 12
Figura 5-2: Clasificación de la criptología 13
Figura 6-2: Clasificación de la criptografía 15
Figura 7-2: Proceso de la criptografía asimétrica 16
Figura 8-2: Proceso de la criptografía simétrica 19
Figura 9-2: Matriz de Estado 21
Figura 10-2: Matriz para clave 21
Figura 11-2: Proceso del algoritmo AES de cifrado y descifrado 22
Figura 12-2: Proceso AddRoundKey 24
Figura 13-2: Proceso SubByte 24
Figura 14-2: Inversos multiplicativos en hexadecimal 25
Figura 15-2: Transformación 25
Figura 16-2: Sustitución S-BOX 26
Figura 17-2: Proceso ShiftRow 26
Figura 18-2: Matrices de Estado S y S' 26
Figura 19-2: Desplazamiento de la segunda fila del proceso ShiftRow 27
Figura 20-2: Desplazamiento de la tercera fila del proceso ShiftRow 27
Figura 21-2: Desplazamiento de la cuarta fila del proceso ShiftRow 27
Figura 22-2: Proceso MixColumn 28
Figura 23-2: Representación de matriz de etapa 28
Figura 24-2: Representación de forma gráfica de etapa 28
Figura 25-2: Tabla inversa S-BOX 29
Figura 26-2: Representación de matriz de etapa 30
Figura 27-2: Proceso DES 32
Figura 28-2: Función (F) Feisel de DES 33
Figura 29-2: Generación de claves DES 34
Figura 30-2: Proceso 3DES 35
Figura 31-2: Proceso de la técnica LSB 42
Figura 32-2: Comparativa de acuerdo el orden de complejidad 44

Figura 1-3: Logo de FlexHEX 48
Figura 2-3: Logo de Guiffy Image Diff 49
Figura 3-3: Logo de Cryptool 49
Figura 4-3: Paquetes de la aplicación 50
Figura 5-3: Clases de la aplicación 51
Figura 6-3: Interfaz de usuario de la aplicación Algoritmo AES base 51
Figura 7-3: Splash Aplicación Algoritmo AES base 53
Figura 8-3: Proceso de cifrado del algoritmo AES base 54
Figura 9-3: Captura del archivo de texto "Origen.txt" 55
Figura 10-3: Captura del resultado de cifrado de la aplicación desarrollada 56
Figura 11-3: Captura de interfaz para exportación del mensaje cifrado 56
Figura 12-3: Captura del archivo exportado con el mensaje cifrado 56
Figura 13-3: Proceso de descifrado del algoritmo AES base 57
Figura 14-3: Captura del archivo de texto "Cifrado.txt" 58
Figura 15-3: Captura del resultado de descifrado de la aplicación desarrollada 59
Figura 16-3: Captura de interfaz para exportación del mensaje descifrado 59
Figura 17-3: Captura del archivo exportado con el mensaje descifrado 59
Figura 18-3: Paquetes de la aplicación 60
Figura 19-3: Clases de la aplicación 61
Figura 20-3: Interfaz de usuario de la aplicación 61
Figura 21-3: Splash Aplicación Algoritmo con librerías 62
Figura 22-3: Resultado de cifrado con la aplicación con librerías 63
Figura 23-3: Resultado de cifrado con la aplicación desarrollada 63
Figura 24-3: Resultado de descifrado con la aplicación con librerías 64
Figura 25-3: Resultado de descifrado con la aplicación desarrollada 64
Figura 26-3: Función ShiftColumn 65
Figura 27-3: Ejemplo de función ShiftColumn 66
Figura 28-3: Ejemplo matriz de código decimal de la clave A 66
Figura 29-3: Ejemplo matriz de código decimal de la clave A 66
Figura 30-3: Ejemplo matriz invertida de la clave A 66
Figura 31-3: Ejemplo matriz de código decimal de la clave B 67
Figura 32-3: Nuevo algoritmo de cifrado 2NAES 67
Figura 33-3: Nuevo algoritmo de descifrado 2NAES 67
Figura 34-3: Paquetes de la aplicación 68
Figura 35-3: Clases de la aplicación 68
Figura 36-3: Interfaz de usuario de la aplicación Nuevo Algoritmo Criptográfico (2NAES) 69
Figura 37-3: Splash Aplicación Nuevo algoritmo 2NAES 70

Figura 38-3: Proceso de cifrado del nuevo algoritmo 2NAES 71
Figura 39-3: Captura del archivo de texto “Origen.txt” 72
Figura 40-3: Captura del resultado de cifrado de la aplicación desarrollada 73
Figura 41-3: Captura de interfaz para exportación del mensaje cifrado 73
Figura 42-3: Captura del archivo exportado con el mensaje cifrado 73
Figura 43-3: Proceso de descifrado del nuevo algoritmo 2NAES 74
Figura 44-3: Captura del archivo de texto “Cifrado.txt” 75
Figura 45-3: Captura del resultado de descifrado de la aplicación desarrollada 76
Figura 46-3: Captura de interfaz para exportación del mensaje descifrado 76
Figura 47-3: Captura del archivo exportado con el mensaje descifrado 76
Figura 48-3: Complejidad del algoritmo AES base 77
Figura 49-3: Complejidad del nuevo algoritmo 2NAES 78
Figura 50-3: Paquetes de la aplicación 79
Figura 51-3: Clases de la aplicación 80
Figura 52-3: Interfaz de usuario de la aplicación Esteganografía LSB 81
Figura 53-3: Splash Aplicación Esteganografía LSB 82
Figura 54-3: Proceso de embebido del algoritmo esteganográfico LSB en imágenes 83
Figura 55-3: Captura del resultado de embebido de la aplicación desarrollada 84
Figura 56-3: Captura del resultado de guardar la imagen embebida de la aplicación desarrollada 84
Figura 57-3: Imagen “Base.bmp” 85
Figura 58-3: Imagen “Base_embebido.bmp” 85
Figura 59-3: Proceso de extracción del algoritmo esteganográfico LSB en imágenes 85
Figura 60-3: Captura del resultado de extracción de la aplicación desarrollada 86
Figura 61-3: Captura de interfaz para exportación del mensaje extraído 86
Figura 62-3: Confirmación de exportación del mensaje 87
Figura 63-3: Captura del archivo extraído con el mensaje descifrado 87
Figura 64-3: Paquetes de la aplicación 88
Figura 65-3: Clases de la aplicación 89
Figura 66-3: Interfaz de usuario de la aplicación Prototipo I 90
Figura 67-3: Splash Aplicación Prototipo I LSB-AES 92
Figura 68-3: Proceso de cifrado y embebido del Prototipo I 93
Figura 69-3: Proceso de extracción y descifrado del Prototipo I 94
Figura 70-3: Paquetes de la aplicación 95
Figura 71-3: Clases de la aplicación 96
Figura 72-3: Interfaz de usuario de la aplicación Prototipo II 96
Figura 73-3: Splash Aplicación Prototipo II LSB-2NAES 98

Figura 74-3: Proceso de cifrado y embebido del Prototipo II 100
Figura 75-3: Proceso de extración y descifrado del Prototipo II 101

Figura 1-4: Captura del resultado de cifrado y embebido del Prototipo I con clave de 128 bits 106
Figura 2-4: Captura del resultado de cifrado y embebido del Prototipo I con clave de 192 bits 108
Figura 3-4: Captura del resultado de cifrado y embebido del Prototipo I con clave de 256 bits 109
Figura 4-4: Captura del resultado de descifrado y extracción del Prototipo I con clave de 128 bits 111
Figura 5-4: Captura del resultado de descifrado y extracción del Prototipo I con clave de 192 bits 112
Figura 6-4: Captura del resultado de descifrado y extracción del Prototipo I con clave de 256 bits 113
Figura 7-4: Captura del resultado de cifrado y embebido del Prototipo II con clave de 128 bits 115
Figura 8-4: Captura del resultado de cifrado y embebido del Prototipo II con clave de 192 bits 116
Figura 9-4: Captura del resultado de cifrado y embebido del Prototipo II con clave de 256 bits 118
Figura 10-4: Captura del resultado de descifrado y extracción del Prototipo II con clave de 128 bits 119
Figura 11-4: Captura del resultado de descifrado y extracción del Prototipo II con clave de 192 bits 120
Figura 12-4: Captura del resultado de descifrado y extracción del Prototipo II con clave de 256 bits 122
Figura 13-4: Imagen “Base.bmp” con el Prototipo I con clave de 128 bits 124
Figura 14-4: Imagen “Base_embebido.bmp” con el Prototipo I con clave de 128 bits 124
Figura 15-4: Tamaño de la imagen “Base.bmp” con el Prototipo I con clave de 128 bits ... 124
Figura 16-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo I con clave de 128 bits 124
Figura 17-4: Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 128 bits 125
Figura 18-4: Resultado de comparación de imágenes 125
Figura 19-4: Diferencia en los pixeles entre las imágenes comparadas 126
Figura 20-4: Imagen “Base.bmp” con el Prototipo I con clave de 192 bits 126

Figura 21-4: Imagen “Base_embebido.bmp” con el Prototipo I con clave de 192 bits 126
Figura 22-4: Tamaño de la imagen “Base.bmp” con el Prototipo I con clave de 192 bits ... 127
Figura 23-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo I con clave de 192 bits 127
Figura 24-4: Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 192 bits 128
Figura 25-4: Resultado de comparación de imágenes 128
Figura 26-4: Diferencia en los pixeles entre las imágenes comparadas 129
Figura 27-4: Imagen “Base.bmp” con el Prototipo I con clave de 256 bits 129
Figura 28-4: Imagen “Base_embebido.bmp” con el Prototipo I con clave de 256 bits 129
Figura 29-4: Tamaño de la imagen “Base.bmp” con el Prototipo I con clave de 256 bits ... 130
Figura 30-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo I con clave de 256 bits 130
Figura 31-4: Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 256 bits 130
Figura 32-4: Resultado de comparación de imágenes 131
Figura 33-4: Diferencia en los pixeles entre las imágenes comparadas 131
Figura 34-4: Imagen “Base.bmp” con el Prototipo II con clave de 128 bits 132
Figura 35-4: Imagen “Base_embebido.bmp” con el Prototipo II con clave de 128 bits 132
Figura 36-4: Tamaño de la imagen “Base.bmp” con el Prototipo II con clave de 128 bits .. 133
Figura 37-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo II con clave de 128 bits 133
Figura 38-4: Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 128 bits 133
Figura 39-4: Resultado de comparación de imágenes 134
Figura 40-4: Diferencia en los pixeles entre las imágenes comparadas 134
Figura 41-4: Imagen “Base.bmp” con el Prototipo II con clave de 192 bits 135
Figura 42-4: Imagen “Base_embebido.bmp” con el Prototipo II con clave de 192 bits 135
Figura 43-4: Tamaño de la imagen “Base.bmp” con el Prototipo II con clave de 192 bits .. 136
Figura 44-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo II con clave de 192 bits 136
Figura 45-4: Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 192 bits 136
Figura 46-4: Resultado de comparación de imágenes 137
Figura 47-4: Diferencia en los pixeles entre las imágenes comparadas 137
Figura 48-4: Imagen “Base.bmp” con el Prototipo II con clave de 256 bits 138
Figura 49-4: Imagen “Base_embebido.bmp” con el Prototipo II con clave de 256 bits 138

Figura 50-4: Tamaño de la imagen “Base.bmp” con el Prototipo II con clave de 256 bits.. 139
Figura 51-4: Tamaño de la imagen “Base_embebido.bmp” con el Prototipo II con clave de 256 bits 139
Figura 52-4: Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 256 bits 139
Figura 53-4: Resultado de comparación de imágenes 140
Figura 54-4: Diferencia en los pixeles entre las imágenes comparadas 140
Figura 55-4: Resultados de la comparación de los indicadores del 1 al 4 145
Figura 56-4: Mensaje cifrado con el Prototipo I con clave de 128 bits 146
Figura 57-4: Mensaje cifrado con el Prototipo II con clave de 128 bits 146
Figura 58-4: Mensaje cifrado con el Prototipo I con clave de 192 bits 147
Figura 59-4: Mensaje cifrado con el Prototipo II con clave de 192 bits 147
Figura 60-4: Mensaje cifrado con el Prototipo I con clave de 256 bits 147
Figura 61-4: Mensaje cifrado con el Prototipo II con clave de 256 bits 147
Figura 62-4: Alfabeto utilizado para realizar las pruebas de criptoanálisis 148
Figura 63-4: Entropía del mensaje cifrado con el Prototipo I con clave de 128 bits 149
Figura 64-4: Entropía del mensaje cifrado con el Prototipo II con clave de 128 bits 149
Figura 65-4: Entropía del mensaje cifrado con el Prototipo I con clave de 192 bits 149
Figura 66-4: Entropía del mensaje cifrado con el Prototipo II con clave de 192 bits 149
Figura 67-4: Entropía del mensaje cifrado con el Prototipo I con clave de 256 bits 149
Figura 68-4: Entropía del mensaje cifrado con el Prototipo II con clave de 256 bits 149
Figura 69-4: Resultados promedios del Indicador 5: Entropía 150
Figura 70-4: Histograma del mensaje cifrado con el Prototipo I con clave de 128 bits 151
Figura 71-4: Histograma del mensaje cifrado con el Prototipo II con clave de 128 bits 151
Figura 72-4: Histograma del mensaje cifrado con el Prototipo I con clave de 192 bits 152
Figura 73-4: Histograma del mensaje cifrado con el Prototipo II con clave de 192 bits 152
Figura 74-4: Histograma del mensaje cifrado con el Prototipo I con clave de 256 bits 153
Figura 75-4: Histograma del mensaje cifrado con el Prototipo II con clave de 256 bits 153
Figura 76-4: Resultados promedios del Indicador 6: Histograma 154
Figura 77-4: N-grama del mensaje cifrado con el Prototipo I con clave de 128 bits 155
Figura 78-4: N-grama del mensaje cifrado con el Prototipo II con clave de 128 bits 155
Figura 79-4: N-grama del mensaje cifrado con el Prototipo I con clave de 192 bits 155
Figura 80-4: N-grama del mensaje cifrado con el Prototipo II con clave de 192 bits 155
Figura 81-4: N-grama del mensaje cifrado con el Prototipo I con clave de 256 bits 156
Figura 82-4: N-grama del mensaje cifrado con el Prototipo II con clave de 256 bits 156
Figura 83-4: Resultados promedios del Indicador 7: N-gramas 157

Figura 84-4: Autocorrelación del mensaje cifrado con el Prototipo I con clave de 128 bits 157
Figura 85-4: Autocorrelación del mensaje cifrado con el Prototipo II con clave de 128 bits 158
Figura 86-4: Autocorrelación del mensaje cifrado con el Prototipo I con clave de 192 bits 158
Figura 87-4: Autocorrelación del mensaje cifrado con el Prototipo II con clave de 192 bits 158
Figura 88-4: Autocorrelación del mensaje cifrado con el Prototipo I con clave de 256 bits 159
Figura 89-4: Autocorrelación del mensaje cifrado con el Prototipo II con clave de 256 bits 159
Figura 90-4: Resultados promedios del Indicador 8: Autocorrelación 160
Figura 91-4: Interfaz para ingresar el patrón para determinar la clave por fuerza bruta 161
Figura 92-4: Opciones de análisis para determinar la clave por fuerza bruta 161
Figura 93-4: Progreso del análisis por fuerza bruta con el Prototipo I con clave de 128 bits 162
Figura 94-4: Progreso del análisis por fuerza bruta con el Prototipo II con clave de 128 bits 162
Figura 95-4: Progreso del análisis por fuerza bruta con el Prototipo I con clave de 192 bits 162
Figura 96-4: Progreso del análisis por fuerza bruta con el Prototipo II con clave de 192 bits 162
Figura 97-4: Progreso del análisis por fuerza bruta con el Prototipo I con clave de 256 bits 163
Figura 98-4: Progreso del análisis por fuerza bruta con el Prototipo II con clave de 256 bits 163
Figura 99-4: Resultado parcial de la ejecución por fuerza bruta 163
Figura 100-4: Resultados promedios del Indicador 9: Fuerza bruta 164
Figura 101-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 1: No.Claves utilizadas 169
Figura 102-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 2: No.Rondas 170
Figura 103-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 3: No.Funciones usadas por el algoritmo 171
Figura 104-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 4: No.Funciones ejecutadas por el algoritmo 172
Figura 105-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 5: Entropía 173

Figura 106-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 6: Histogramas 174
Figura 107-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 7: N-gramas 175
Figura 108-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 8: Autocorrelación 176
Figura 109-4: Resultados obtenidos (de acuerdo a la escala) del Indicador 9: Fuerza bruta 177
Figura 110-4: Resultados de la comparación por Indicador 178
Figura 111-4: Resultados totales de la comparación 179
Figura 112-4: Curva de X^2 184

LISTA DE ANEXOS

Anexo A: Código fuente principal

RESUMEN

El trabajo de investigación “Nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes”, integra dos campos relacionados con la seguridad: criptografía y esteganografía con el objetivo de incrementar la seguridad de la información, la primera cifra el mensaje y la segunda oculta el mensaje tras un medio multimedia. Se compararon los indicadores considerados en las variables y se aplicó la estadística descriptiva e inferencial para la demostración de la hipótesis. El software utilizado para la investigación fue: Netbeans como ambiente de desarrollo, FlexHEX para comparar el código hexadecimal de las imágenes, Guiffy Image Diff para comparar las diferencias entre imágenes pixel a pixel y Cyptool que permitió realizar pruebas de criptoanálisis. El algoritmo criptográfico seleccionado como base fue el AES (Advanced Encryption Standard), de acuerdo al estudio realizado en base a los parámetros de comparación y para la técnica esteganográfica en imágenes se seleccionó la técnica LSB (Least Significant Bit). Se desarrolló, implementó y comparó los resultados obtenidos analizando las características de los algoritmos y ejecutando criptoanálisis a los mensajes encriptados entre el Prototipo II que utiliza el nuevo algoritmo criptográfico denominado 2NAES (2 New Advanced Encryption Standard) y el Prototipo I que utiliza el algoritmo AES base, a los cuales se les incorporó la técnica esteganográfica en imágenes LSB. Se concluye, que el nuevo algoritmo criptográfico 2NAES con la incorporación de la técnica LSB, mejoró la seguridad en un 52% en comparación con el algoritmo criptográfico AES base ya que el mensaje es más difuso. Se recomienda buscar nuevas modificaciones a los algoritmos criptográficos e incorporarlos con técnicas esteganográficas con el objetivo de mejorar la seguridad.

Palabras clave: <CRIPTOGRAFÍA> , <ESTEGANOGRAFÍA> , <ADVANCED ENCRYPTION STANDARD (AES)> , <LEAST SIGNIFICANT BIT (LSB)> , <CRIPTOANÁLISIS> , <2 NEW ADVANCED ENCRYPTION STANDARD (2NAES)>

SUMMARY

The research paper "New cryptographic algorithm with the addition of steganography in images", integrates two security-related fields: cryptography and steganography in order to increase the information security, the first encrypts the message and the second hides message after a multimedia environment. The indicators considered in the variables were compared and descriptive and inferential statistics to demonstrate the hypothesis was applied. The software used for the research was: NetBeans development environment, FlexHEX to compare the hexadecimal code of the images, Image Guiffy Diff to compare differences between images pixel by pixel and Cyptool which allowed testing of cryptanalysis. The cryptographic algorithm was selected as the basis AES (Advanced Encryption Standard), according to the study based on comparison parameters and for the steganographic technique images the LSB (Least Significant Bit) technique was selected. It was developed, implemented and compared the results obtained by analyzing the characteristics of algorithms and running cryptanalysis to encrypted between Prototype II messages using the new cryptographic algorithm called 2NAES (2 new Advanced Encryption Standard) and the Prototype I using the AES algorithm base, to which were incorporated the images LSB steganographic technique. It is concluded that the new cryptographic algorithm 2NAES by incorporating the LSB technique, improved security by 52% compared to the AES cryptographic algorithm basis because the message is more diffuse. It is recommended to seek further amendments to the cryptographic algorithms and incorporate with steganographic techniques in order to improve safety.

Keywords: <CRYPTOGRAPHY>, <STEGANOGRAPHY> , <ADVANCED ENCRYPTION STANDARD (AES)>, <LEAST SIGNIFICANT BIT (LSB)>, <CRYPTANALYSIS> , <2 NEW ADVANCED ENCRYPTION STANDARD (2NAES)>

CAPÍTULO I

INTRODUCCIÓN

En este Capítulo, se determina el enfoque u orientación general, identificación del problema, justificación, objetivos e hipótesis que se desea comprobar con el desarrollo de la investigación.

1.1. Problema de investigación

1.1.1. Planteamiento del problema

La seguridad en la transmisión de información a través de canales inseguros es de vital importancia en las comunicaciones, por lo que surge el término de seguridad informática, para brindar una mayor confianza de la información, ya que existen intrusos que desean acceder ella para conocerla y/o utilizarla con propósitos maliciosos, por lo que se propone mejorar los algoritmos criptográficos y combinar sus fortalezas con las técnicas de esteganografía. (**GABA & KUMAR**, 2013 , p. 395)

La criptografía y la esteganografía son dos campos en la seguridad informática: la primera cifra el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada uno de estos campos por separado no asegura la información, pero si se combinan ambas técnicas para cifrar y ocultar el mensaje tras un medio multimedia, mejoraría el nivel de seguridad. (**SEGURA & DÍAZ**, 2014 , http://www.boletin.upiita.ipn.mx/index.php/ciencia/215-cyt-numero-33/109-implementacion-del-algoritmo-esteganografico-lsb-least-significant-bit-estandar-en-archivos-de-audio-mp3)

El término **criptografía** proviene del griego *kryptos* (oculto) y *graphos* (escritura), lo que etimológicamente significa *"escritura oculta"*, es la ciencia que utilizando algoritmos específicos permite convertir los datos originales en criptogramas que son enviados por un canal inseguro en el que únicamente el destinatario puede descifrar los datos y obtener el mensaje original.

Las principales propiedades que se deben asegurar con la criptografía son: confidencialidad, integralidad, vinculación, autenticación. (**SAINI & VERMA**, 2013 , p. 607)

El origen cifra el mensaje utilizando una clave y el destino la descifra utilizando la clave dependiendo del tipo de criptografía utilizada ya sea simétrica o asimétrica, como se muestra en la Figura 1-1.

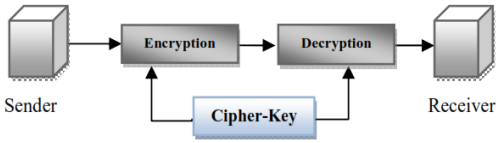


Figura 1-1 Proceso de la criptografía
Fuente: Saini & Verma, 2013

El término **esteganografía** proviene del griego *steganos* (cubierto) y *graphos* (escritura), lo que etimológicamente significa *"escritura cubierta"*, es la ciencia de ocultar información con técnicas específicas dentro de un archivo multimedia evitando la revelación de la información oculta para que *pasen inadvertidos*, el emisor embebe el mensaje en el archivo multimedia y el receptor lo extrae para obtener el mensaje oculto. (**SAINI & VERMA**, 2013 , p. 607)

El origen embebe el mensaje y el destino lo extrae (puede utilizarse clave una), como se muestra en la Figura 2-1.

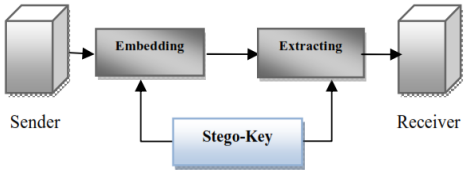


Figura 2-1 Proceso de la esteganografía
Fuente: Saini & Verma, 2013

Existen varias técnicas utilizadas para ocultar la información dentro de archivos multimedia como: documentos, imágenes, audio, vídeo, otros, por ejemplo: enmascaramiento y filtrado, algoritmos y transformaciones, inserción en el bit menos significativo (LSB), entre otros. (**CASTILLO, CASTILLO, & NÚÑEZ**, 2013 , http://ccns.jimdo.com/encriptaci%C3%B3n-de-datos/)

Actualmente se han realizado varias investigaciones previas acerca del tema en cuestión, entre ellas:

- La investigación: "*A Hybrid Approach for Image Security by Combining Encryption and Steganography"* (**SAINI & VERMA**, 2013 , pp. 607-611), menciona que debido a la necesidad que existe de asegurar la información antes de que sea transmitida, existen varios algoritmos de criptografía que han sido desarrollados para cumplir este fin. En la investigación, primero cifra la imagen con una nueva versión del algoritmo criptográfico y lo combina con la

esteganografía para mejorar el nivel de seguridad contra posibles ataques, sin embargo, no incluye el aporte de los autores para posibles investigaciones futuras.

- La investigación: *"Implementation of Steganography Using CES Technique"* (**GABA & KUMAR**, 2013 , pp. 395-399), menciona que debido al crecimiento de las redes y el avance de la tecnología es necesario incrementar la complejidad en la protección de información para que tengan seguridad, para lo que existen 2 grandes áreas: la criptografía y la esteganografía. La criptografía altera la estructura de los datos y la esteganografía oculta la información detrás de un medio multimedia. En la investigación, se utiliza el algoritmo criptográfico CES para el intercambio de información usando la técnica de pre proceso que comprende reducir el tamaño del texto y luego alterarlo utilizando una clave, lo que permite ocultar una mayor cantidad de información con técnicas esteganográficas para proteger la misma, sin embargo, la implementación de un pre procesamiento adicional requiere de una cantidad mayor complejidad de desarrollo, implementación y de recursos.
- La investigación: *"Security Improvisation in Image Steganography using DES"* (**KUMAR, HEMRAJANI, & KISHORE**, 2013 , pp. 1094-1099), menciona que debido a la evolución de las tecnologías de internet y sus aplicaciones requieren de un alto nivel de seguridad en los datos sobre canales de comunicación. La esteganografía en imágenes es una técnica digital para ocultar información atrás de una imagen. La técnica del Bit Menos Significativo (Least Significant Bit - LSB), es una de las más populares debido a su capacidad y alta capacidad de ocultación. Las técnicas esteganográficas se basan en estrategias de embebido con menos consideraciones para el pre procesamiento, sin embargo, el algoritmo original no provee el pre procesamiento requerido para una mayor seguridad, no ofrecen flexibilidad, robustez y alto nivel de seguridad. Se plantea una técnica de esteganografía en imágenes basado en DES (Data Encryption Standard) usando la fortaleza de mapeo s-box y clave secreta.

Por lo que el enfoque original de la presente investigación, que se diferencia de investigaciones anteriores es que se orienta a la creación e implementación de un nuevo algoritmo de cifrado (utilizando un algoritmo criptográfico existente como base) que será aplicado a archivos de tipo texto y posterior a esto, la ocultación de información con técnicas de esteganografía en imágenes, lo que mejorará la seguridad.

1.1.2. Formulación del problema

¿Cuál sería el nivel de mejora en la seguridad al implementar un nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes?

1.1.3. Sistematización del problema

- ¿Cuáles son los algoritmos criptográficos existentes en la actualidad?
- ¿Cuáles son las características, ventajas y desventajas de los algoritmos criptográficos existentes?
- ¿Cómo mejorar el rendimiento de un algoritmo criptográfico existente para crear uno mejor?
- ¿Cuáles son las técnicas utilizadas en la actualidad para la esteganográficas en imágenes?
- ¿Cómo combinar la criptografía con la esteganografía en imágenes?

1.2. Justificación de la investigación

1.2.1. Justificación teórica

La presente investigación plantea la integración de dos campos relacionados con la seguridad: la criptografía y la esteganografía con el objetivo de incrementar la seguridad de la información.

La criptografía es una medida de seguridad utilizada para almacenar o transmitir información sensible, para que ésta no pueda ser obtenida con facilidad por terceros ya que está cifrada. Además, existe un proceso de descifrado a través del cual la información puede ser descifrada de nuevo a su estado original.

La esteganografía permite que los mensajes estén ocultos atrás de un medio multimedia como documentos, imágenes, audio, video, entre otros, y pasen inadvertidos, protegiendo la información.

1.2.2. Justificación metodológica

De la criptografía se utiliza la gran ventaja de cifrar la información, para lo que se selecciona un algoritmo criptográfico existente como base, del cual se analizan las características y ventajas de un algoritmo criptográficos existentes y se selecciona uno como base para crear uno nuevo, posteriormente a los algoritmos criptográficos se incorpora la esteganografía en imágenes ocultando la información con una técnica esteganográfica seleccionada, con el objetivo de que en el caso de ser interceptando y extraída la información por atacantes que desean utilizarla, no pueda ser interpretada ya que está cifrada con un nuevo algoritmo criptográfico, mejorando la seguridad.

1.2.3. Justificación práctica

Los experimentos se realizarán en dos escenarios de prueba utilizando los prototipos desarrollados, el Prototipo I utiliza el algoritmo criptográfico base y el Prototipo II utiliza el nuevo algoritmo criptográfico, a los cuales se les incorpora la técnica esteganográfica en imágenes, posteriormente se compararán los mensajes cifrados por estos prototipos, con la finalidad de demostrar la mejora de la seguridad.

1.3. Objetivos

1.3.1. Objetivo general

- Implementar un nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes

1.3.2. Objetivos específicos

- Analizar los algoritmos criptográficos existentes para la selección de uno de ellos como base.
- Determinar las técnicas existentes esteganográficas en imágenes para la selección de una de ellas.
- Implementar un nuevo algoritmo criptográfico, en base al algoritmo existente escogido, con la incorporación de la esteganografía en imágenes.
- Verificar de nivel de seguridad en los escenarios de prueba del proceso implementado con el nuevo algoritmo y la incorporación de la esteganografía en imágenes, en comparación con el algoritmo base.

1.4. Hipótesis

La implementación del nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes mejora la seguridad en comparación con el algoritmo criptográfico base.

1.5. Estructura del proyecto de investigación

La estructura del proyecto de investigación se compone de lo siguiente:

- En el **CAPÍTULO I INTRODUCCIÓN.** Se determina el enfoque u orientación general, identificación del problema, justificación, objetivos e hipótesis que se desea comprobar con el desarrollo de la investigación.

- En el **CAPÍTULO II MARCO DE REFERENCIA.** Se profundiza el estado del arte acerca de las ciencias en estudio y se determina el algoritmo base para la creación del nuevo algoritmo criptográfico y la técnica esteganográfica en imágenes que se utilizará.

- En el **CAPÍTULO III DISEÑO DE INVESTIGACIÓN**. Se detalla el tipo de investigación, diseño, métodos, técnicas, instrumentos con su respectiva validación, adicionalmente se crea e implementa el algoritmo criptográfico base, el nuevo algoritmo criptográfico, la técnica esteganográfica, los prototipos (integran los algoritmos criptográficos y la técnica esteganográfica) y se definen los escenarios de pruebas.

- En el **CAPÍTULO IV RESULTADOS Y DISCUSIÓN**. Se desarrollan las pruebas en los escenarios establecidos, se analizan, comparan los resultados obtenidos y se comprueba la hipótesis planteada.

- En las **CONCLUSIONES** y **RECOMENDACIONES.** Se enuncian los resultados obtenidos y las sugerencias luego de realizar la investigación.

- En la **BIBLIOGRAFÍA.** Se enlistan las referencias utilizadas para el desarrollo de la investigación.

- En los **ANEXOS.** Se adjunta la información adicional utilizada.

CAPÍTULO II

MARCO DE REFERENCIA

En este Capítulo, se profundiza el estado del arte acerca de las ciencias en estudio y se determina el algoritmo base para la creación del nuevo algoritmo criptográfico y la técnica esteganográfica en imágenes que se utilizará.

2.1. Estado del arte

Actualmente se han realizado investigaciones relacionadas con las ciencias en estudio, las cuales han aportado al presente trabajo de investigación para:

- Comprender de mejor manera la teoría relacionada con la criptografía y la esteganografía.
- Identificar la descripción del problema y los objetivos planteados en las investigaciones realizadas.
- Conocer el proceso desarrollado por los autores para realizar las investigaciones.
- Determinar las métricas, indicadores o parámetros que los autores utilizan para realizar las pruebas.
- Determinar el aporte que realizan los autores en las investigaciones realizadas.
- Observar los resultados y conclusiones obtenidas en base a las pruebas realizadas que hacen los autores.

Entre las principales investigaciones, se mencionan:

Investigación: "A Hybrid Approach for Image Security by Combining Encryption and Steganography" (**SAINI & VERMA**, 2013 , pp. 607-611)

La motivación de los autores del artículo científico nace de la necesidad que existe de asegurar la información antes de que sea transmitida, existen varios algoritmos de criptografía que han sido desarrollados para cumplir este fin.

El proceso desarrollado se resume en lo siguiente:

- Determinación de los tipos de transformaciones del algoritmo criptográfico
- Propuesta e implementación de la modificación del algoritmo criptográfico
- Realización de pruebas con varias imágenes de diferentes dimensiones con las cuales se generan y se recuperas las imágenes cifradas para medir la eficiencia del algoritmo modificado
- Análisis de los resultados obtenidos en las pruebas desarrolladas
- Definición de conclusiones y recomendaciones.

La propuesta del esquema híbrido se muestra en la Figura 1-2.

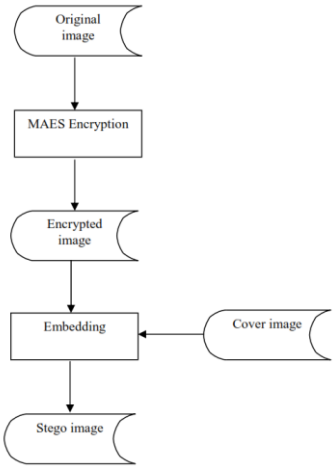


Figura 1-2 Propuesta del esquema híbrido
Fuente: Saini & Verma, 2013

Para las pruebas realizadas se utilizan imágenes con diferentes dimensiones (128x128, 256x256, 512x512) y tipo de color (color o escala de grises). Se analizan los histogramas de la imagen original y de la imagen cifrada comparándolas entre sí, obteniendo una variación significativa entre ellas.

Se calcula la correlación entre la adyacencia de los pixeles en tres direcciones: horizontal, vertical y diagonal usando la fórmula correspondiente, con lo que se obtienen los resultados mostrados en la Tabla 1-2.

Tabla 1-2 Resultados obtenidos de la correlación entre la adyacencia de los pixeles

Image Size	Direction	Plain Image	Cipher Image
128*128	Horizontal	0.9034	0.00788
	Vertical	0.9403	-0.0011
	Diagonal	0.8953	-0.0016
256*256	Horizontal	0.9667	-0.00394
	Vertical	0.9458	-0.00388
	Diagonal	0.9821	-0.00474
512*512	Horizontal	0.9775	-0.03986
	Vertical	0.9845	-0.02198
	Diagonal	0.9634	-0.0672

Fuente: Saini & Verma, 2013

Para la prueba de medidas de calidad se compara los algoritmos criptográficos, con los parámetros MSE (Mean Square Error) y PSNR (Peak Signal to Noise Ratio) que miden la calidad de la imagen esteganográfica, para la imagen flores.jpg en sus diferentes dimensiones.

Con los resultados obtenidos se concluye que con la modificación del algoritmo criptográfico se obtiene un mejor rendimiento, logrando un método seguro de cifrado y ocultación de la información.

Sin embargo, la investigación realizada no incluye el aporte de los autores para posibles investigaciones futuras.

Investigación: "Implementation of Steganography Using CES Technique" (**GABA & KUMAR**, 2013 , pp. 395-399)

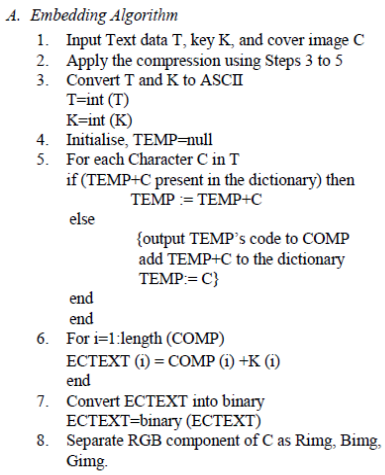
La motivación de los autores del artículo científico nace debido al crecimiento de las redes y el avance de la tecnología, es necesario incrementar la complejidad de la protección de información para que tengan seguridad, por lo que existen dos grandes áreas: la criptografía y la esteganografía. La criptografía altera la estructura de los datos y la esteganografía oculta la información detrás de un medio digital.

En la investigación se crea el algoritmo criptográfico CES para intercambio de información utilizando la técnica de pre proceso que comprende reducir el tamaño del texto y luego alterarlo utilizando una clave, lo que permite ocultar una mayor cantidad de información con técnicas esteganográficas para proteger la misma, sin embargo, la implementación de un pre procesamiento adicional requiere de una cantidad mayor complejidad de desarrollo, implementación y de recursos.

El proceso desarrollado se resume en lo siguiente:

- Compresión de los datos para reducir el tamaño y la cantidad de texto para ser ocultada dentro de la imagen.
- Modificación de los datos utilizando una clave
- Separación de los componentes RGB (Red, Green, Blue) de la imagen y estos componentes son transformados separadamente a un dominio de frecuencia usando DCT (Discrete Cosine Transform)
- El texto modificado es ocultado en el 8vo coeficiente de DCT en el componente azul ya que es el menos observable por el ojo humano, disminuyendo la distorsión y mejorando la calidad de la imagen.

Se definen los algoritmos para embeber y extraer la información, como se muestra en la Figura 2-2 y Figura 3-2.

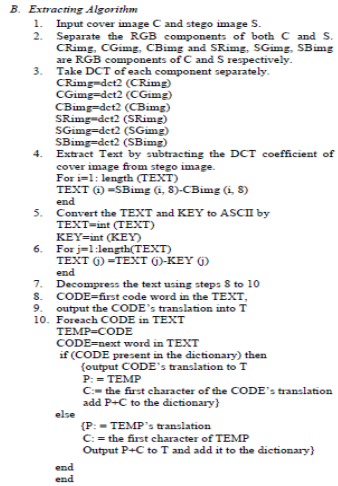


A. Embedding Algorithm

1. Input Text data T, key K, and cover image C
2. Apply the compression using Steps 3 to 5
3. Convert T and K to ASCII
   T=int (T)
   K=int (K)
4. Initialise, TEMP=null
5. For each Character C in T
   if (TEMP+C present in the dictionary) then
   TEMP := TEMP+C
   else
   {output TEMP's code to COMP
   add TEMP+C to the dictionary
   TEMP:= C}
   end
   end
6. For i=1:length (COMP)
   ECTEXT (i) = COMP (i) +K (i)
   end
7. Convert ECTEXT into binary
   ECTEXT=binary (ECTEXT)
8. Separate RGB component of C as Rimg, Bimg, Gimg.

Figura 2-2 Propuesta de algoritmo para embeber la información

Fuente: Gaba & Kumar, 2013



B. Extracting Algorithm

1. Input cover image C and stego image S.
2. Separate the RGB components of both C and S. CRimg, CGimg, CBimg and SRimg, SGimg, SBimg are RGB components of C and S respectively.
3. Take DCT of each component separately.
   CRimg=dct2 (CRimg)
   CGimg=dct2 (CGimg)
   CBimg=dct2 (CBimg)
   SRimg=dct2 (SRimg)
   SGimg=dct2 (SGimg)
   SBimg=dct2 (SBimg)
4. Extract Text by subtracting the DCT coefficient of cover image from stego image.
   For i=1: length (TEXT)
   TEXT (i) =SBimg (i, 8)-CBimg (i, 8)
   end
5. Convert the TEXT and KEY to ASCII by
   TEXT=int (TEXT)
   KEY=int (KEY)
6. For j=1:length(TEXT)
   TEXT (j) =TEXT (j)-KEY (j)
   end
7. Decompress the text using steps 8 to 10
8. CODE=first code word in the TEXT,
9. output the CODE's translation into T
10. Foreach CODE in TEXT
    TEMP=CODE
    CODE=next word in TEXT
    if (CODE present in the dictionary) then
    {output CODE's translation to T
    P: = TEMP
    C:= the first character of the CODE's translation
    add P+C to the dictionary}
    else
    {P: = TEMP's translation
    C: = the first character of TEMP
    Output P+C to T and add it to the dictionary}
    end
    end

Figura 3-2 Propuesta de algoritmo para extraer la información

Fuente: Gaba & Kumar, 2013

Para las pruebas realizadas se comparan las imágenes originales con las que contienen la información oculta en base a las métricas de MSE (Mean Square Error) y PSNR (Peak Signal to Noise Ratio) que miden la calidad de la imagen esteganográfica, mientras menor sea el valor de MSE significa que existe un menor error entre las dos imágenes.

Se realizaron las pruebas en tres imágenes y los resultados realizados con las imágenes esteganográficas y los histogramas se muestran en la Tabla 2-2:

Tabla 2-2 Resultados obtenidos de la comparación de imágenes

Cover Image	Stego Image	Plain Text	Key	Compressed Text	Altered Text (with key)	PSNR (In dB)	MSE	Bit change %
Football.jpg (Size: 10.8 KB)	Ballstego.jpg (Size: 10.8 KB)	It was a mind blowing and mind boggling event	beautiful	Ju!xbt!b!njoe!cmpx cphhm !fwfou	¬Ú ïÔÝ × ÐÎÐÚ ÌÒáã ØÛÊÎÍ ÏÝÛÚ×	59.28	0.1538	17.495
Greens.jpg (Size: 22.5 KB)	Gstego.jpg (Size: 22.5 KB)	Rain in spain falls mainly in plain	accepted	Sbjo! !tq gbmmt!n mz qm	ÁÍÒ ØÒ ÌÒáÔØ Ñ Òë ÔÍ	62.74	0.0889	10.356
Peppers.png (Size: 99.4 KB)	Pstego.png (Size: 99.4 KB)	He got all wet in rain on Wednesday	superior	If!hpu!bmm!xf jo!sb !p Xfeoftebz	½Û ÍáÞ Óáã ŸØ Ûá èÒ Ù ÉÚÚßÊæÎÑì	60.01	0.1282	15.332

COMPARISION OF CES RESULTS WITH P. BHARTI AND LSB TECHNIQUE

Image	LSB	P. Bharti's [2]technique	CES Technique
A	35.1	37	59.28
B	36.3	41	62.74

Fuente: Gaba & Kumar, 2013

Se concluye que el algoritmo implementado con pre procesamiento permite ocultar mayor cantidad de información sin distorsión en la imagen ya que utiliza el componente color azul que provee un mayor valor de PSNR por lo que esta técnica es más robusta y segura.

Sin embargo, la investigación planteada en este artículo científico tiene un alto nivel matemático y de conocimiento multimedia en la definición de los algoritmos utilizados para criptografía y esteganografía, lo cual es complejo de entender sin una mayor preparación en el área.

Investigación: "Security Improvisation in Image Steganography using DES" (**KUMAR, HEMRAJANI, & KISHORE, 2013** , pp. 1094-1099)

La motivación de los autores del artículo científico nace debido a la increíble evolución de las tecnologías de internet y sus aplicaciones requieren de un alto nivel de seguridad en los datos enviados sobre canales de comunicación. La esteganografía en imágenes es una técnica digital para ocultar información atrás de una imagen. La técnica del bit menos significativo (Least Significant-bit - LSB), es una de las más populares debido a su capacidad y alta capacidad de ocultación, por lo que se plantea una técnica de esteganografía en imágenes basado en el algoritmo DES (Data Encryption Standard) usando la fortaleza de mapeo s-box y clave secreta.

El proceso desarrollado se resume en lo siguiente:

- Definición de la función de codificación
- Determinación del método para la recuperación de la imagen

La propuesta del modelo esteganográfico se basa en SDES, S-box y clave secreta, como se muestra en la Figura 4-2.

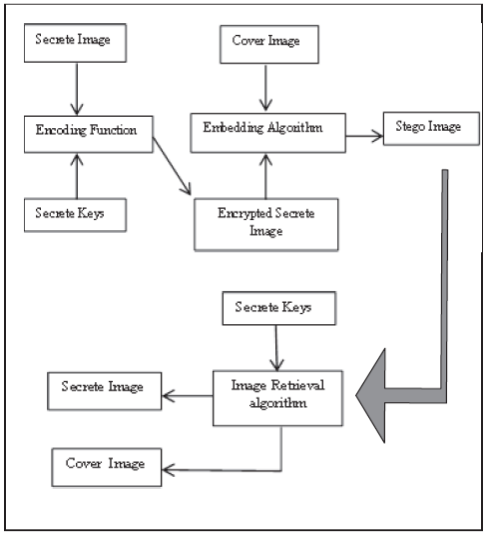


Figura 4-2 Propuesta del modelo esteganográfico
Fuente: Kumar, Hemrajani, & Kishore, 2013

Luego de realizar las pruebas de laboratorio con imágenes, los resultados obtenidos con las métricas establecidas se muestran en la Tabla 3-2:

Tabla 3-2 Resultados obtenidos de la comparación de imágenes y tamaños

Name of Image	Size (Pixel)	Capacity	PSNR In DB
Baboon	64× 64	25 %	54.58
Cameraman	64× 64	25 %	55.01
Lena	64× 64	25 %	59.28
Pirate	64× 64	25 %	51.63
Living room	64× 64	25 %	50.19
Women-darkhair	64× 64	25 %	52.85

Fuente: Kumar, Hemrajani, & Kishore, 2013

Se concluye que en la propuesta basada en el algoritmo DES la fortaleza del mapeo S-box y la clave secreta para cifrar la imagen secreta, mejora la seguridad y la calidad de la imagen en comparación con el algoritmo existente. La esteganografía combinada con la criptografía es una herramienta muy poderosa que permite la comunicación secreta de forma segura.

Sin embargo, la investigación planteada en este artículo científico requiere un alto nivel matemático de los algoritmos utilizados para criptografía y esteganografía, lo cual es complejo de entender sin una mayor preparación en el área.

2.2. Criptografía

2.2.1. Antecedentes

En la actualidad, debido a la evolución de las tecnologías de internet y sus aplicaciones, requieren de un alto nivel de seguridad en los datos que son enviados sobre canales de comunicación. (**RAMAIYA, HEMRAJANI, & SAXENA**, 2013 , p. 1094)

2.2.1.1. Criptología

El término **criptología** proviene del griego *kryptos (oculto)* y *logos (estudio)*, lo que etimológicamente significa *"estudio de lo oculto"*, es la ciencia que trata los problemas que se relacionan con la seguridad en el intercambio de mensajes en clave entre el emisor y el receptor a través de un canal de comunicaciones (ese canal suele ser una red de computadoras).

Como se ilustra en la Figura 5-2, la criptología está dentro de la teoría de la información que es parte de las matemáticas. (**PANTOJA**, 2015 , http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto)

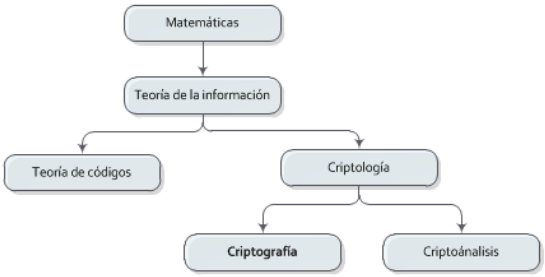


Figura 5-2 Clasificación de la criptología
Fuente: http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto

Según la Red Académica y de Investigación Española, la criptología está dividida en dos grandes ramas que son: (**RED ACADÉMICA DE INVESTIGACIÓN ESPAÑOLA**, 2013 , http://www.rediris.es/cert/doc/unixsec/node29.html)

- Criptoanálisis
- Criptografía

Los cuales se detallan a continuación:

2.2.1.2. Criptoanálisis

El término **criptoanálisis** proviene del griego *kryptos (oculto)* y *analýein (desatar)*, se encarga del estudio de los métodos para obtener una información que ha sido cifrada utilizando algoritmos y protocolos criptográficos cuyo objetivo principal es romper o forzar el código. (**COMUNIDAD EXPRESIÓN BINARIA**, 2011 , http://www.expresionbinaria.com/glosario/criptoanalisis/)

Los principales ataques sobre algoritmos criptográficos son:

- **Criptoanálisis lineal:** es una forma general de criptoanálisis sobre la base de la búsqueda de aproximaciones afines para la acción de un sistema de cifrado. Se aplica a sistemas de cifrado de bloques y sistemas de cifrado de flujo.
- **Criptoanálisis diferencial:** es un conjunto de técnicas para identificar diferencias a través de la red de transformaciones, con la finalidad de determinar los lugares en los que el sistema de cifrado tiene un comportamiento al azar y el uso de estas propiedades para recuperar la clave secreta. (**COMUNIDAD DOCSETOOLS**, 2013 , http://docsetools.com/articulos-noticias-consejos/article_129171.html)
- **Criptoanálisis truncado diferencial:** es una generalización de criptoanálisis diferencial, la variante truncada considera las diferencias que se determinan sólo parcialmente.
- **Fuerza bruta:** es una técnica que se basa en probar todas las combinaciones posibles hasta encontrar la palabra o texto legible en el que fue cifrado, para obtener el criptograma. (**DMK**, 2010 , http://www.redinfocol.org/atacando-por-fuerza-bruta-bruteforce-1/)

2.2.1.3. Criptografía

El término **criptografía** proviene del griego *kryptos (oculto)*, *y graphos (escritura)*, lo que etimológicamente significa *"escritura oculta"*. Se ocupa de las técnicas de cifrado destinadas a alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados para conseguir la confidencialidad de los mensajes. (**CASTILLO, CASTILLO, & NÚÑEZ**, 2013 , http://ccns.jimdo.com/encriptaci%C3%B3n-de-datos/)

Propiedades

Las propiedades de la criptografía son:

- **Confidencialidad:** garantiza únicamente personal autorizado pueda acceder a la información.
- **Integridad:** garantiza que la información esté completa y sin modificaciones con respecto a la original.
- **No repudio:** proporciona protección para que las entidades implicadas en la comunicación no puedan negar haber participado en la misma.
- **Autenticación:** permite verificar la identidad del comunicante.

2.2.2. Tipos de criptografía

Existen dos tipos de criptografía: clásica y moderna, como se muestra en la Figura 6-2. (**PANTOJA**, 2015 , http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto)

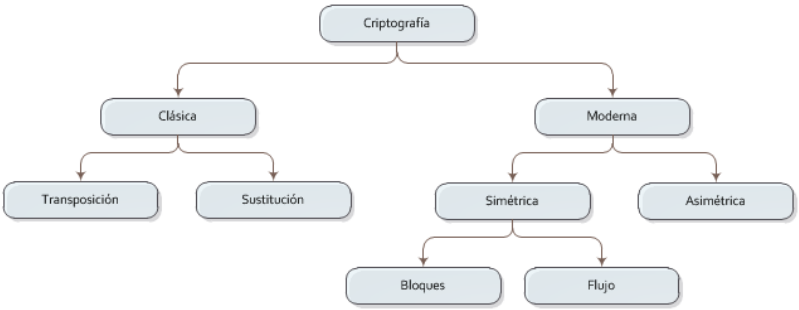


Figura 6-2 Clasificación de la criptografía
Fuente: http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto

2.2.2.1. Criptografía clásica

En los métodos criptográficos clásicos, la criptografía se puede clasificar en dos tipos:

- Trasposición
- Sustitución

Estos métodos criptográficos clásicos se detallan a continuación: (**PANTOJA**, 2015 , http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto)

2.2.2.1.1. Transposición

El método de trasposición consiste en cambiar el orden de los símbolos que forman parte del texto original, para disminuir las redundancias del texto original en el texto cifrado.

2.2.2.1.2. Sustitución

El método de sustitución consiste en reemplazar unos caracteres por otros, permite añadir confusión al mensaje cifrado y evitar mantener una relación entre el texto original y el texto cifrado.

Los métodos de transposición y sustitución no son muy utilizados debido a la sencillez de los mismos, por lo que pueden ser vulnerables a ataques y la información puede estar comprometida.

2.2.2.2. Criptografía moderna

En los métodos criptográficos modernos, la criptografía puede clasificarse en dos tipos:

- Criptografía asimétrica o de clave pública
- Criptografía simétrica o de clave privada.

2.2.2.2.1. Criptografía asimétrica o de clave pública

La criptografía de clave asimétrica o de clave pública, utiliza dos claves diferentes en cada uno de los extremos de la comunicación: una clave pública y otra clave privada (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/16/criptografia-algoritmos-de-cifrado-de-clave-asimetrica/), como se muestra en la Figura 7-2.

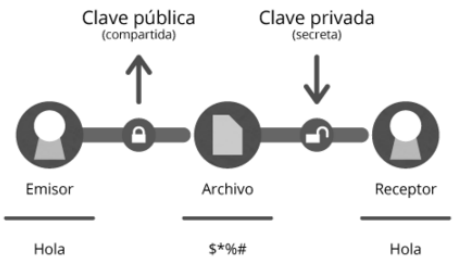


Figura 7-2 Proceso de la criptografía asimétrica
Fuente: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQyJk9rLS3vY5nlorIfyJN9BG8tKltstH3xQhBBactuJs4H9H0q

La clave pública puede ser accesible a todos los usuarios del sistema de comunicación que se desee y la clave privada debe ser protegida y guardada de forma segura, secreta.

Los algoritmos asimétricos están basados en funciones matemáticas simples de resolver en un sentido, pero en el sentido inverso su resolución es compleja, a menos que se conozca la clave con la que fue cifrada. Las claves públicas y privadas se generan simultáneamente y tienen relación entre sí. (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/16/criptografia-algoritmos-de-cifrado-de-clave-asimetrica/)

Además, es posible firmar documentos con la clave privada y verificando la identidad con la pública, con la finalidad de certificar que el emisor es quien dice ser. (**GUTIERREZ**, 2013 , http://www.genbetadev.com/seguridad-informatica/tipos-de-criptografia-simetrica-asimetrica-e-hibrida)

Ventajas y desventajas

Las principales ventajas y desventajas de la criptografía de clave asimétrica o de clave pública se muestran en la Tabla 4-2. (**COMUNIDAD ECURED**, 2014 , http://www.ecured.cu/index.php/Criptograf%C3%ADa_asim%C3%A9trica)

Tabla 4-2 Ventajas y desventajas de la criptografía asimétrica

Ventaja(s)	Desventaja(s)
• La distribución de las claves es más simple y segura (el propietario posee la clave privada y la clave pública es la que se distribuye)	• Requiere de mayor tiempo y recursos para realizar el proceso. • Las claves deben ser de mayor tamaño en comparación con las simétricas. • El mensaje cifrado ocupa un mayor espacio que el original.

Realizado por: Méndez Pablo, 2015

Entre los principales algoritmos de clave asimétrica, se mencionan:

- Diffie-Hellman
- RSA (Rivest, Shamir y Adleman)
- DSA (Digital Signature Algorithm)

2.2.2.2.1.1. Diffie-Hellman

El algoritmo criptográfico Diffie-Hellman fue desarrollado por Whitfiled Diffie y Martín Hellman en 1976 (**MORALES**, 2014 , http://gaussianos.com/criptografia-cifrado-de-clave-publica-i/). Es utilizado para generar una clave privada simétrica a ambos extremos de un canal de comunicación inseguro. Se emplea para obtener la clave secreta con la que posteriormente se cifrará la información, junto con un algoritmo de cifrado simétrico. Su seguridad radica en la dificultad de calcular el logaritmo discreto de números grandes. (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/16/criptografia-algoritmos-de-cifrado-de-clave-asimetrica/)

2.2.2.2.1.2. RSA (Rivest, Shamir y Adleman)

El algoritmo criptográfico RSA (Rivest Shamir Adleman) fue desarrollado en el año de 1977 por Ron Rivest, Adi Shamir y Len Adleman. (**MORALES**, 2014 , http://gaussianos.com/criptografia-cifrado-de-clave-publica-ii/)

El cifrado RSA es un algoritmo de clave pública o asimétrica por bloques, que tiene dos claves:

- **Clave pública:** se distribuye a los usuarios que el propietario de ella desee.
- **Clave privada**: es guardada en secreto por su propietario.

Cuando se envía un mensaje, el emisor utiliza la clave pública de cifrado del receptor para cifrar el mensaje y una vez que recibe el receptor, utiliza su clave privada para descifrarlo.

2.2.2.2.1.3. DSA (Digital Signature Algorithm)

El algoritmo de firma digital DSA (Digital Signature Algorithm), lo propuso el National Institute of Standards and Technology (NIST) en 1991 y fue adoptado por los Federal Information Processing Standards (FIPS) en 1993, este algoritmo sirve para firmar y no para cifrar información. Desde entonces se ha revisado cuatro veces. (**SSL247**, 2015 , https://www.ssl247.es/certificats-ssl/rsa-dsa-ecc)

2.2.2.2.2. Criptografía simétrica o de clave privada

La criptografía simétrica o de clave privada, utiliza una única clave, la cual permite cifrar y descifrar la información transmitida a través del canal inseguro (**DE LUZ**, 2010 ,

http://www.redeszone.net/2010/11/16/criptografia-algoritmos-de-cifrado-de-clave-asimetrica/), como se muestra en la Figura 8-2.

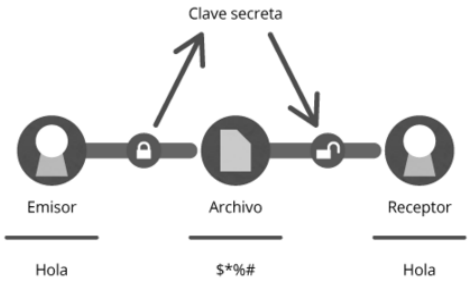


Figura 8-2 Proceso de la criptografía simétrica
Fuente: http://img.genbetadev.com/2013/01/criptografia-asimetrica.png

El punto débil del sistema es el canal de comunicación entre el emisor y el receptor, porque es vulnerable a ataques de intercepción de la clave que se ha transmitido sin seguridad, por ejemplo: cuando se la envía por correo electrónico, a través de una llamada o mensaje telefónico, entre otros. (**GUTIERREZ**, 2013 , http://www.genbetadev.com/seguridad-informatica/tipos-de-criptografia-simetrica-asimetrica-e-hibrida)

El emisor cifra la información, la envía a través del canal inseguro y el receptor descifra esa información con la misma clave del emisor.

Los algoritmos criptográficos son públicos por lo que su fortaleza depende de la complejidad interna y de la longitud de la clave que se utiliza para evitar posibles ataques de fuerza bruta. (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/)

Ventajas y desventajas

Las principales ventajas y desventajas de la criptografía de clave simétrica o de clave privada se muestran en la Tabla 5-2. (**MASON**, 2014 , http://www.ehowenespanol.com/ventajas-desventajas-criptografia-clave-simetrica-info_276624/)

Tabla 5-2 Ventajas y desventajas de la criptografía simétrica

Ventaja(s)	Desventaja(s)
• Fácil de usar. • Útil para el cifrado de archivos, ya que utiliza la misma clave para cifrar y descifrar. • Rápida • Utiliza menos recursos.	• Necesidad de comunicar la clave pública. • Dificultad para gestionar un gran número de claves. • Posible vulnerabilidad a ataques de fuerza bruta y de diccionario.

Realizado por: Méndez Pablo, 2015

Entre los principales algoritmos de clave simétrica, se mencionan:

- AES (Advanced Encryption Standard)
- DES (Data Encryption Standard)
- 3-DES (Triple Data Encryption Standard)

2.2.2.2.2.1. AES (Advanced Encryption Standard)

El algoritmo AES se lo conoce también como Rijndael, fue desarrollado los criptólogos belgas, Joan Daemen y Vincent Rijmen, estudiantes de la Katholieke Universiteit Leuven, es un tipo de cifrado por bloques que ha sido utilizado como un estándar de cifrado porque ha sido considerado seguro. Este cifrado se puede implementar en sistemas hardware y software. (**COMUNIDAD WIKIPEDIA**, 2015), https://es.wikipedia.org/wiki/Advanced_Encryption_Standard)

El sistema criptográfico AES tiene un tipo de clave simple, y posee claves de longitudes variables de 128 bits, de 192 bits y de 256 bits. Desde 2006, el AES es uno de los algoritmos criptográficos simétricos más utilizados, ya que garantiza la seguridad contra ataques conocidos, posee un diseño simple y su implementación es fácil en escenarios con dispositivos móviles que posean recursos limitados y en dispositivos con mayores recursos. Además, es utilizado en varios protocolos como SSL, aplicaciones como voz sobre IP (VoIP), entre otros. (**ANGEL**, 2005 , http://computacion.cs.cinvestav.mx/~jjangel/aes/AES_v2005_jjaa.pdf)

Resiste a criptoanálisis diferencial, truncado diferencial y lineal, lo cual es una gran ventaja porque lo hace resistente a este tipo de ataques. El tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo), para claves de 128 bits es de $5x10^{21}$ años.

Entre las principales ventajas se mencionan: (**ROBERTSON**, 2014 , http://www.ehowenespanol.com/ventajas-algoritmos-rijndael-info_290285/)

- **Flexibilidad:** variedad de bloques y tamaños de claves, por lo que puede ser implementado en varios sistemas.
- **Seguridad:** debido a la fuerte estructura algebraica que posee, lo que le permite tener mayor seguridad que un algoritmo promedio. Se puede acceder fácilmente a componentes del programa, con la finalidad de detectar y resolver problemas de seguridad.
- **Memoria:** requiere una menor cantidad de memoria para realizar los procesos, por lo que lo hace una buena alternativa para programas software y hardware.

- **La ventaja pública:** tiene una característica de código público, lo cual permite aumentar la seguridad y la conveniencia para el usuario.

El algoritmo criptográfico base AES, está formado por un conjunto de rondas con iteraciones de cuatro funciones matemáticas diferentes e invertibles para producir una información cifrada. La información generada por cada función es un resultado intermedio, que se conoce como Estado o Estado Intermedio. (**VÁSQUEZ**, 2007 , http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf)

El algoritmo representa el Estado como un matriz rectangular de bytes, que posee cuatro filas y Nb columnas. Siendo el número de columnas Nb en función del tamaño del bloque: (**VIDAL**, 2015)

$$N_b = \text{tamaño del bloque utilizado en bits} / 32$$

Si se tiene un bloque con 128 bits se tendría una matriz de cuatro filas y $N_b = 128/32 = 4$ columnas, como se muestra en la Figura 9-2.

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

Figura 9-2 Matriz de Estado
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

La clave del sistema se representa con una estructura análoga a la del Estado, es decir, se representa mediante una matriz rectangular de bytes de cuatro filas y N_k columnas.
Siendo el número de columnas N_k en función del tamaño de la clave:

$$N_k = \text{tamaño de la clave en bits} / 32.$$

Si se tiene una clave con 128 bits se dispondría de una matriz de cuatro filas y $N_r = 128/32 = 4$ columnas, como se muestra en la Figura 10-2.

k_{00}	k_{01}	k_{02}	k_{03}
k_{10}	k_{11}	k_{12}	k_{13}
k_{20}	k_{21}	k_{22}	k_{23}
k_{30}	k_{31}	k_{32}	k_{33}

Figura 10-2 Matriz para clave
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

En resumen, la relación existente entre longitud de clave, tamaño de bloque y rondas, se muestra en la Tabla 6-2.

Tabla 6-2 Relación entre longitud de clave, tamaño de bloque y rondas

Clave	Longitud de clave (Nk)	Tamaño de bloque (Nb)	Número de rondas (Nr)
128 bits	4	4	10
192 bits	6	4	12
256 bits	8	4	14

Realizado por: Méndez Pablo, 2015

Las funciones utilizadas en AES son las siguientes:

- AddRoundKey o etapa de adición de clave.
- SubByte o sustituciones de bytes.
- ShiftRow o etapa de desplazamiento de filas.
- MixColumn o etapas de mezcla de columnas.
- KeySchedule o expansión de clave.

Los procesos de cifrado y descifrado del algoritmo AES se muestran en la Figura 11-2.

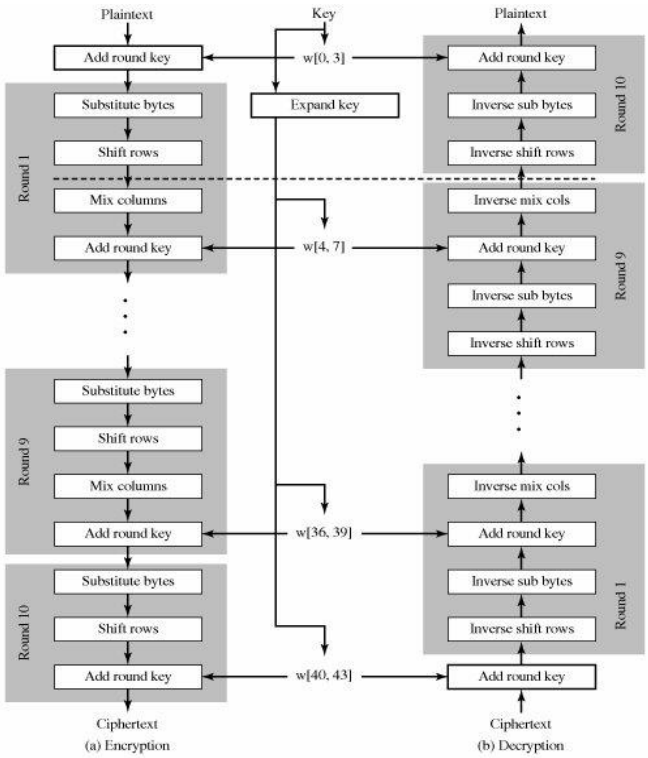


Figura 11-2 Proceso del algoritmo AES de cifrado y descifrado
Fuente: http://flylib.com/books/3/190/1/html/2/images/05fig01.jpg

El proceso del algoritmo AES consiste en dos partes:

- Proceso de cálculo de subclaves
- Proceso de cifrado

Cálculo de subclaves

Basándose en el principio de la criptografía moderna, mediante el cual se establece que la seguridad de un algoritmo sólo debe depender de la clave utilizada, se utilizan diferentes subclaves K_i tanto en el cifrado como en el descifrado para que el resultado del algoritmo dependa de una información externa al sistema: la clave del usuario.

Estas subclaves (RoundKey) se derivan de la clave principal K mediante el uso de dos funciones: una de expansión y otra de selección.

Siendo n el número de rondas que aplique el algoritmo, el número total de bits para subclaves que se necesitan para todas las rondas es igual al tamaño del bloque utilizado multiplicado por n + 1.

Puede expresarse como (n + 1) * Nb bytes. Es decir, por ejemplo, para un tamaño de bloque de 128 bits y 10 rondas, se necesitan 1408 bits de subclaves: $(128 * 11 = 1408 \text{ bits})$.

Por tanto, lógicamente, el número de claves que se generan depende del número de rondas empleadas (N_r).

Función AddRoundKey

En esta función se procede a realizar un XOR byte a byte entre la matriz de Estado y la matriz de la clave o subclave, dependiendo de la ronda en la que se encuentre.

Como se muestra en la Figura 12-2, cada byte del Estado se combina con un byte de la subclave usando la operación XOR ($\oplus$).

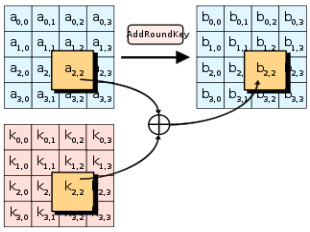


Figura 12-2 Proceso AddRoundKey
Fuente: http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

De esta manera toma la matriz [a$_{i\ j}$] y [k$_{i\ j}$] y aplicando el XOR da como resultado la matriz $[a_{i\,j} \oplus k_{i\,j}]$

Una vez acaba la función MixColumn, se vuelve a proceder con esta función de AddRoundKey, creándose de nuevo un Estado intermedio 1 pero en la siguiente ronda, o bien de estar en la última ronda se creará el bloque de salida.

En este caso la matriz utilizada para realizar el XOR es la matriz de subclave.

Función SubByte

En esta función, etapa o tratamiento se procede a realizar una sustitución no lineal que se aplica a cada byte de la matriz de Estado de forma independiente, generando un nuevo byte, es decir, a cada elemento de la matriz de estado se le sustituye por otro byte que depende del primero mencionado. Esta sustitución se lleva a cabo utilizando tablas o matrices S-Box invertibles.

Como se muestra en la Figura 13-2, cada byte en el Estado es reemplazado con su entrada en una tabla de búsqueda fija de 8 bits, S; bij = S (aij).

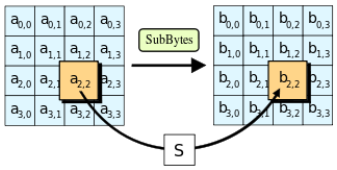


Figura 13-2 Proceso SubByte
Fuente: http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

Esta sustitución tiene dos apartados o transformaciones:

- Cada byte es considerado como un elemento del GF (2^8) que genera el polinomio irreducible $m(x) = x^8 + x^4 + x^3 + x + 1$ y sustituido por su inversa multiplicativa. El valor cero en este caso no varía pues no tiene recíproco, como se muestra en la Figura 14-2.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	8D	F6	CB	52	7B	D1	E8	4F	29	C0	B0	E1	E5	C7
1	74	B4	AA	4B	99	2B	60	5F	58	3F	FD	CC	FF	40	EE	B2
2	3A	6E	5A	F1	55	4D	A8	C9	C1	0A	98	15	30	44	A2	C2
3	2C	45	92	6C	F3	39	66	42	F2	35	20	6F	77	BB	59	19
4	1D	FE	37	67	2D	31	F5	69	A7	64	AB	13	54	25	E9	09
5	ED	5C	05	CA	4C	24	87	BF	18	3E	22	F0	51	EC	61	17
6	16	5E	AF	D3	49	A6	36	43	F4	47	91	DF	33	93	21	3B
7	79	B7	97	85	10	B5	BA	3C	B6	70	D0	06	A1	FA	81	82
8	83	7E	7F	80	96	73	BE	56	9B	9E	95	D9	F7	02	B9	A4
9	DE	6A	32	6D	D8	8A	84	72	2A	14	9F	88	F9	DC	89	9A
A	FB	7C	2E	C3	8F	B8	65	48	26	C8	12	4A	CE	E7	D2	62
B	0C	E0	1F	EF	11	75	78	71	A5	8E	76	3D	BD	BC	86	57
C	0B	28	2F	A3	DA	D4	E4	0F	A9	27	53	04	1B	FC	AC	E6
D	7A	07	AE	63	C5	DB	E2	EA	94	8B	C4	D5	9D	F8	90	6B
E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

Figura 14-2 Inversos multiplicativos en hexadecimal
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

- Después se aplica la siguiente transformación afín en GF (2^8), siendo $x_0, x_1,..., x_7$ los bits del byte correspondiente, y a su vez $y_0, y_1,..., y_7$ los bits del byte correspondiente al resultado, como se muestra en la Figura 15-2.

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Figura 15-2 Transformación
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

El resultado final de estas dos transformaciones se puede expresar en una tabla de sustitución denominada S-Box aplicada a cada byte, sabiendo que este está expresado en forma hexadecimal, de manera que el elemento más a la izquierda deberá situarse en la columna verde y el elemento más a la derecha deberá colocarse en la fila azul, siendo el resultado final el lugar donde coincidan o se unan, como se muestra en la Figura 16-2.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Figura 16-2 Sustitución S-BOX
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

Función ShiftRow

En esta función, etapa o tratamiento se procede a realizar un desplazamiento a la izquierda cíclicamente de las filas que conforman la matriz de estado actual, es decir, rotar los bytes de las filas de la matriz de estado resultante de la transformación anterior a la izquierda.

Cada fila se desplaza un número de posiciones diferentes, este número de rondas o rotaciones dependerá del tamaño del bloque N_b, explicado en la estructura del algoritmo.

Como se muestra en la Figura 17-2, los bytes en cada fila del Estado son rotados de manera cíclica hacia la izquierda. El número de lugares que cada byte es rotado difiere para cada fila.

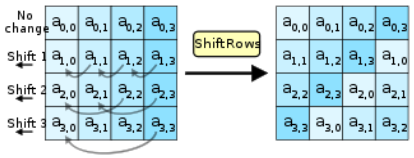


Figura 17-2 Proceso ShiftRow
Fuente: http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

De forma gráfica, en un bloque de 128, se muestra en la Figura 18-2.

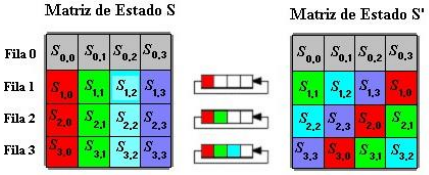


Figura 18-2 Matrices de Estado S y S'
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

La primera fila que en este caso es la fila 0 queda igual.

La segunda fila desplaza un byte, como se muestra en las Figura 19-2.

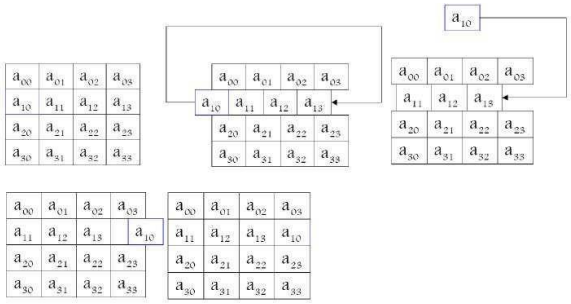


Figura 19-2 Desplazamiento de la segunda fila del proceso ShiftRow
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

La tercera columna desplaza dos bytes circularmente, como se muestra en la Figura 20-2.

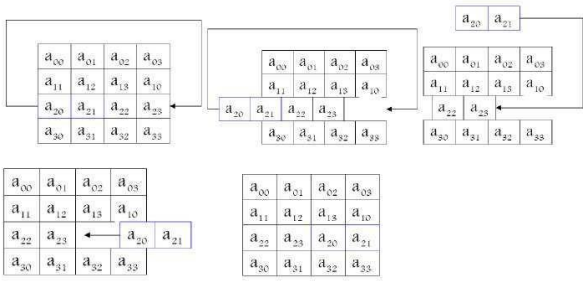


Figura 20-2 Desplazamiento de la tercera fila del proceso ShiftRow
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

Finalmente, la cuarta fila desplaza 3 bytes circularmente, como se muestra en la Figura 21-2.

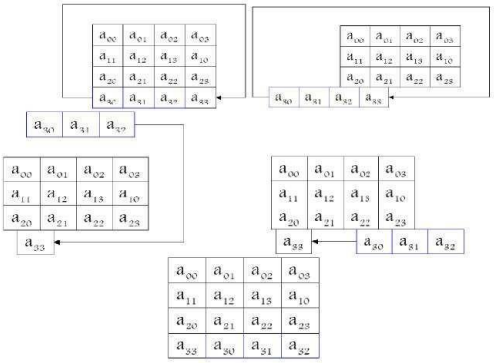


Figura 21-2 Desplazamiento de la cuarta fila del proceso ShiftRow
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

Función MixColumn

En esta función, etapa o tratamiento se procede sobre los bytes de una misma columna de la matriz de estado resultante de la transformación anterior.

Como se muestra en la Figura 22-2, cada columna del Estado es multiplicada por un polinomio constante c(x).

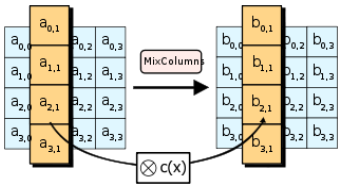


Figura 22-2 Proceso MixColumn
Fuente: http://es.wikipedia.org/wiki/Advanced_Encryption_Standard

Las columnas son tratadas como polinomios, cuyos coeficientes pertenecen a GF (2^8).

La transformación consiste en multiplicar las columnas en módulo $x^4 + 1$ por el polinomio $c(x) = 03\,x^3 + 01\,x^2 + 01\,x + 02$. De forma matemática se expresa:

$$S'(x) = c(x) \approx S(x)$$

Donde:

S'(x): Representa la matriz de estado resultante de esta etapa

S(x): Representa la matriz de Estados de entrada.

La fórmula se puede expresar en forma matricial como se muestra en la Figura 23-2.

$$\begin{pmatrix} S'_{0,C} \\ S'_{1,C} \\ S'_{2,C} \\ S'_{3,C} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} S_{0,C} \\ S_{1,C} \\ S_{2,C} \\ S_{3,C} \end{pmatrix}.$$

Figura 23-2 Representación de matriz de etapa
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

De forma gráfica, en un bloque de 128, se muestra en la Figura 24-2.

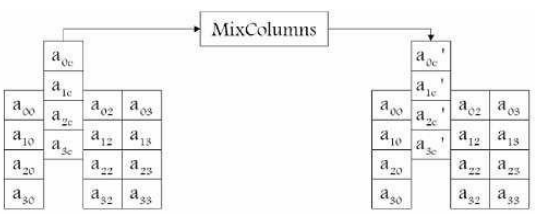


Figura 24-2 Representación de forma gráfica de etapa
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

Proceso de descifrado

El proceso de descifrado es muy similar al cifrado, sólo hay que hacer el proceso inverso, es decir, invertir el orden de todas las operaciones realizadas y hacer las transformaciones inversas. (**VÁSQUEZ**, 2007 , http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf)

En este proceso, las subclaves utilizadas, van desde la última generada en el proceso de cifrado hasta la primera (que corresponderá con bytes de la clave elegida para cifrar).

Las funciones matemáticas utilizadas son invertibles, para el proceso de descifrado se utilizarán las funciones inversas a las utilizadas en el proceso de cifrado.

Función AddRoundKey

La función AddRoundKey desempeña la misma función salvo que en vez de empezar en la primera ronda empezará por la última y acabará por la primera, como se muestra en la Figura 25-2.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Figura 25-2 Tabla inversa S-BOX
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

Función Inv-SubByte

La función Inv-SubByte realizará la aplicación inversa de la correspondiente S-box correspondiente a cada byte de la matriz de estado. De esta manera se obtiene otra tabla con los valores inversos a los valores de la tabla S-box.

Función Inv-ShiftRow

La función Inv-ShiftRow será inversa a la función ShiftRow en la que en vez de desplazar las filas de la matriz hacia la izquierda se desplazarán a la derecha, el mismo número de posiciones que se hubieran desplazado con anterioridad.

Función Inv-MixColumn

Por último, la función Inv-MixColumn será la inversa de la función MixColumn en la que se deberá operar sobre los bytes de una misma columna, considerando las columnas como polinomios con coeficientes en GF (2^8), que serán multiplicados por el polinomio $d(x) = 0\,B\,x^3 + 0\,D\,x^2 + 09\,x + 0\,E$, que es el inverso de $c(x)$. De forma matemática:

$$S(x)=d(x) \approx S'(x)$$

Donde:

S(x): Representa la matriz de Estado resultante de esta etapa

S'(x): Representa la matriz de Estados de entrada.

La fórmula se puede expresar en forma matricial como se muestra en la Figura 26-2.

$$\begin{pmatrix} S'_{0,C} \\ S'_{1,C} \\ S'_{2,C} \\ S'_{3,C} \end{pmatrix} = \begin{pmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{pmatrix} \cdot \begin{pmatrix} S_{0,C} \\ S_{1,C} \\ S_{2,C} \\ S_{3,C} \end{pmatrix}.$$

Figura 26-2 Representación de matriz de etapa
Fuente: http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

2.2.2.2.2.2. DES (Data Encryption Standard)

El algoritmo criptográfico DES fue desarrollado en el año 1977, es un tipo de cifrado en bloque simétrico, su seguridad es media y utiliza un tipo de clave simple. Su arquitectura está basada en un sistema monoalfabético, al que se aplican sucesivas permutaciones y sustituciones al texto original. (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/)

Se realiza una permutación inicial a la información de 64 bits, luego una permutación con entrada de 8 bits y otra de sustitución de entrada de 5 bits, todo este proceso se lo aplica en 16 rondas de cifrado.

Se utiliza una clave simétrica de 64 bits, los 56 primeros bits son empleados para el cifrado y los 8 bits restantes se usan para comprobación de errores durante el proceso. La clave efectiva es de 56 bits, por tanto, se tiene 2^{56} combinaciones de posibles claves, por lo que la fuerza bruta es casi imposible.

El tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo), para claves de 56 bits es de 400 días.

Las principales ventajas y desventajas del algoritmo DES se muestran en la Tabla 7-2. (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/)

Tabla 7-2 Ventajas y desventajas del algoritmo DES

Ventaja(s)	Desventaja(s)
• Es uno de los sistemas más conocidos. • Implementación rápida y simple.	• No se permite una clave de longitud variable. • Es vulnerable al criptoanálisis diferencial y lineal. • La longitud de clave de 56 bits es vulnerable ya que es corta.

Realizado por: Méndez Pablo, 2015

Proceso DES

La estructura básica del algoritmo, consta de 16 fases idénticas de proceso, denominadas rondas. Se realiza una permutación inicial (PI) y una permutación final (PF), que son funciones inversas entre sí (PI "deshace" la acción de PF, y viceversa). Antes de las rondas, el bloque es dividido en dos mitades de 32 bits y procesadas alternativamente, este entrecruzamiento se conoce como esquema Feistel. (**COMUNIDAD WIKIPEDIA**, 2015 , http://es.wikipedia.org/wiki/Data_Encryption_Standard)

La estructura de Feistel asegura que el cifrado y el descifrado sean procesos muy similares, la única diferencia es que las subclaves se aplican en orden inverso cuando se descifra. El resto del algoritmo es idéntico. Esto simplifica enormemente la implementación, en especial sobre hardware, al no haber necesidad de algoritmos distintos para el cifrado y el descifrado.

El símbolo rojo "⊕" representa la operación OR exclusivo (XOR). La función-F mezcla la mitad del bloque con parte de la clave. La salida de la función-F se combina entonces con la otra mitad del bloque, y los bloques son intercambiados antes de la siguiente ronda.

Tras la última ronda, las mitades no se intercambian; ésta es una característica de la estructura de Feistel que hace que el cifrado y el descifrado sean procesos parecidos.
En la Figura 27-2, se ilustra el proceso del algoritmo DES.

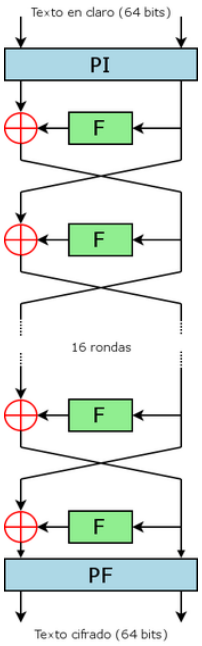


Figura 27-2 Proceso DES
Fuente: http://es.wikipedia.org/wiki/Data_Encryption_Standard

Función (F) de Feisel

La función-F, representada en la Figura 28-2, opera sobre medio bloque (32 bits) cada vez y consta de cuatro pasos: (**COMUNIDAD WIKIPEDIA**, 2015 , http://es.wikipedia.org/wiki/Data_Encryption_Standard)

- **Expansión**: la mitad del bloque de 32 bits se expande a 48 bits mediante la permutación de expansión, denominada E en el diagrama, duplicando algunos de los bits.
- **Mezcla**: el resultado se combina con una subclave utilizando una operación XOR. Dieciséis subclaves, una para cada ronda, se derivan de la clave inicial mediante la generación de subclaves descrita más abajo.
- **Sustitución**: tras mezclarlo con la subclave, el bloque es dividido en ocho trozos de 6 bits antes de ser procesados por las S-Cajas, o cajas de sustitución. Cada una de las ocho S-Cajas reemplaza sus seis bits de entrada con cuatro bits de salida, de acuerdo con una trasformación

no lineal, especificada por una tabla de búsqueda. Las S-cajas constituyen el núcleo de la seguridad de DES, sin ellas, el cifrado sería lineal, y fácil de romper.

- **Permutación**: finalmente, las 32 salidas de las S-cajas se reordenan de acuerdo a una permutación fija; la P-Caja

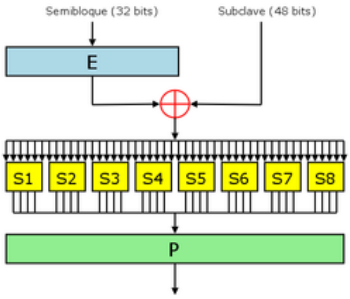


Figura 28-2 Función (F) Feisel de DES
Fuente: http://es.wikipedia.org/wiki/Data_Encryption_Standard

Alternando la sustitución de las S-Cajas, y la permutación de bits de la P-Caja y la expansión-E proporcionan las llamadas "confusión y difusión" respectivamente, un concepto identificado por Claude Shannon en los 40 como una condición necesaria para un cifrado seguro y práctico. (**COMUNIDAD WIKIPEDIA**, 2015 , http://es.wikipedia.org/wiki/Data_Encryption_Standard)

Generación de claves

En la Figura 29-2 se muestra la generación de claves para el cifrado, el algoritmo que se encarga de proporcionar las subclaves. Primero, se seleccionan 56 bits de la clave de las 64 iniciales mediante la Elección Permutada 1 (PC-1), los ocho bits restantes pueden descartarse o utilizarse como bits de comprobación de paridad.

Los 56 bits se dividen entonces en dos mitades de 28 bits; a continuación, cada mitad se trata independientemente. En rondas sucesivas, ambas mitades se desplazan hacia la izquierda uno o dos bits (dependiendo de cada ronda), y entonces se seleccionan 48 bits de subclave mediante la Elección Permutada 2 (PC-2), 24 bits de la mitad izquierda y 24 de la derecha. Los desplazamientos (indicados por "<<<" en el diagrama) implican que se utiliza un conjunto diferente de bits en cada subclave; cada bit se usa aproximadamente en 14 de las 16 subclaves.

La generación de claves para descifrado es similar, se debe generar las claves en orden inverso.

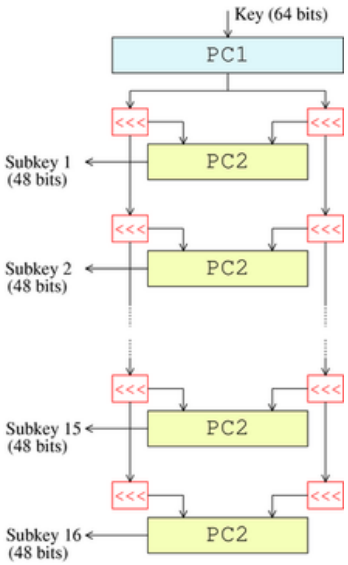


Figura 29-2 Generación de claves DES
Fuente: http://es.wikipedia.org/wiki/Data_Encryption_Standard

2.2.2.2.2.3. TRIPLE-DES (Triple Data Encryption Standard)

Algoritmo desarrollado en el año 2000, con un tipo de cifrado en bloque simétrico con 128 bits, 192 bits y 256 bits, su seguridad es media y utiliza un tipo de clave simple (dividida en 3 partes: k1, k2 y k3). Se basa en aplicar tres veces el algoritmo DES, la clave tiene una longitud de 168 bits (k1, k2 y k3) o de 112 bits (k1 = k2), se requieren 48 rondas. Vulnerable a criptoanálisis diferencial. El tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo), para claves de 112 bits es de 800 días.

El algoritmo criptográfico 3DES parte de una clave de 128 bits, que es divida en dos claves, A y B. Al recibir los datos, se utiliza el algoritmo DES con la clave A, luego se realiza el proceso con la clave B y finalmente se repite el proceso con la clave A, esto permite que la seguridad del algoritmo se incremente, sin embargo se requiere de una mayor cantidad de recursos del computador (**DE LUZ**, 2010 , http://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/)

Las partes principales del algoritmo DES son las siguientes: (**ROJAS & HERNÁNDEZ**, 2014 , http://seguridad-en-redes-mimi.blogspot.com/2012/03/algoritmo-3des.html)

- Particionamiento del texto en bloques de 64 bits (8 bytes).
- Permutación inicial de los bloques.
- Partición de los bloques en dos partes: izquierda (I) y derecha (D).
- Fases de permutación y de sustitución repetidas 16 veces (rondas),
- Reconexión de las partes izquierda y derecha, seguida de la permutación inicial inversa.

El funcionamiento de la variante más simple del 3DES se muestra en la Figura 30-2.

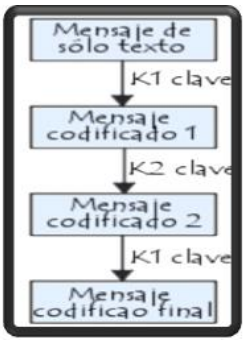


Figura 30-2 Proceso 3DES
Fuente: http://seguridad-en-redes-mimi.blogspot.com/2012/03/algoritmo-3des.html

En el que el mensaje a cifrar es procesado con K1 y K2 y K3 que son las respectivas claves DES. En la variante 3DES las tres claves son diferentes; en la variante 2DES, la primera y tercera clave son iguales. El 3DES permite aumentar de manera significativa la seguridad del DES, sin embargo, requiere de más recursos para el cifrado y descifrado.

2.2.3. Determinación del algoritmo criptográfico base

Luego de realizar la búsqueda de información de estudios primarios acerca de los algoritmos criptográficos simétricos más utilizados y detallarlos en el punto 2.2.2, se procede a seleccionar los algoritmos simétricos o de clave privada debido a sus características.

Los algoritmos criptográficos simétricos seleccionados son:

- AES (Advanced Encryption Standard)
- DES (Data Encryption Standard)
- 3DES (Triple Data Encryption Standard)

Posteriormente, se procede con la síntesis para determinar el algoritmo criptográfico que será utilizado como base, para lo cual se realiza la comparación entre los algoritmos criptográficos simétricos utilizando los siguientes factores de seguridad y rendimiento, tomando como base con la información recopilada y la del artículo científico *"Comparison between DES, 3DES, RC2, RC6, Blowfish and DES"* (**MATHUR & KESARWANI**, 2013 , p.146):

- Tipo de cifrado
- Tipo de clave
- Longitud de la clave

- Tamaño de bloque
- No. Rondas
- Seguridad
- Criptoanálisis
- Tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo)

En la Tabla 8-2 se sistematiza la comparación de los algoritmos criptográficos considerados.

Tabla 8-2 Tabla comparativa entre algoritmos de criptografía simétrica

No.	FACTORES	AES	DES	3DES
1	Tipo de cifrado	Cifrado en bloque simétrico	Cifrado en bloque simétrico	Cifrado en bloque simétrico
2	Tipo de clave	Simple	Simple	Simple (dividida en 3 partes)
3	Longitud de la clave	• 128 bits • 192 bits • 256 bits	• 56 bits	• (k1, k2, k3) 168 bits • (k1 y k2 son la misma) 112 bits
4	Tamaño del bloque	• 128 bits • 192 bits • 256 bits	• 64 bits	• 128 bits • 192 bits • 256 bits
5	No. Rondas	• 10 (128-bits) • 12 (192 bits) • 14 (256 bits)	• 16	• 48
6	Seguridad	Considera seguro	Media	Debilidad en la salida en DES
7	Criptoanálisis	Fuerte contra: • Criptoanálisis diferencial • Criptoanálisis truncado diferencial • Criptoanálisis lineal	Vulnerable a: • Criptoanálisis diferencial • Criptoanálisis lineal	Vulnerable a: • Criptoanálisis diferencial
8	Tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo)	Para clave de 128 bits: • Aproximadamente $5x10^{21}$ años	Para clave de 56 bits: • Aproximadamente 400 días	Para clave de 112 bits: • Aproximadamente 800 días

Realizado por: Méndez Pablo, 2015
Fuente: Mathur & Kesarwani, 2013

De los resultados de la comparación realizada, se puede determinar que el algoritmo criptográfico simétrico AES es el más adecuado debido a sus ventajas en relación a los otros algoritmos ya que permite utilizar claves de 128 bits, 192 bits y 256 bits, entre las principales ventajas se mencionan:

- El tamaño de bloque variable.
- El número de rondas depende de la clave que se utilice.
- Es resistente al criptoanálisis diferencial, truncado diferencial, lineal, por lo que es una de las principales ventajas en comparación con los otros.
- El tiempo requerido para determinar todas las posibles claves (con 50 billones de claves por segundo) es mucho mayor, por lo que lo hace más seguro y resistente en comparación con los otros.

Por lo que será utilizado como base para la elaboración e implementación del nuevo algoritmo criptográfico, a los cuales se incorporará la esteganografía en imágenes para demostrar la hipótesis.

2.3. Esteganografía

2.3.1. Antecedentes

El término **esteganografía** proviene del griego *steganos (*cubierto), *graphos (*escritura), lo que etimológicamente significa "*escritura cubierta"*, es la ciencia que estudia los métodos para ocultar la existencia de un mensaje dentro de otro archivo, de forma que el atacante no note la existencia de dicha información que se encuentra oculta.

La esteganografía no trata de sustituir al cifrado convencional sino de complementarlo, ya que ocultar un mensaje reduce las posibilidades de que sea descubierto durante el intercambio de información a través de medios inseguros (**GUO & LE**, 2010 , p. 879); sin embargo, ya que el mensaje está cifrado, agrega un nivel adicional de seguridad.

A lo largo de la historia han existido varias técnicas para ocultar información, los más conocidos han sido: la tinta invisible, utilizada durante la Segunda Guerra Mundial, las marcas de cualquier tipo sobre ciertos caracteres (desde pequeños pinchazos de alfiler hasta trazos a lápiz que marcan un mensaje oculto en un texto), entre otros mecanismos utilizados. (**RED ACADÉMICA DE INVESTIGACIÓN ESPAÑOLA**, 2013 , http://www.rediris.es/cert/doc/unixsec/node29.html)

Con la evolución de la tecnología, el mecanismo esteganográfico más extendido está basado en las imágenes digitales debido a su excelente capacidad para ocultar información y dado que casi todos los estándares gráficos tienen una graduación de colores mayor de lo que el ojo humano puede apreciar, la imagen no cambia su apariencia de forma notable. (**RED ACADÉMICA DE INVESTIGACIÓN ESPAÑOLA**, 2013 , http://www.rediris.es/cert/doc/unixsec/node29.html)

Las técnicas más utilizadas según el tipo de medio son las siguientes: **(COMUNIDAD GITS INFORMÁTICA**, 2003), http://www.gitsinformatica.com/descargas/Esteganografia.doc)

- **En documentos**: agregando espacios en blanco y tabs porque son más difíciles de identificar para el ojo humano en la mayoría de los editores de texto.
- **En imágenes**: la técnica más utilizada es el LSB (Least Significant Bit), ya que en el computador un archivo de imagen es representado por colores e intensidades de luz en diferentes áreas (pixeles), por lo que los datos del mensaje pueden ser embebidos en la imagen.
- **En audio**: la técnica más utilizada es LBE (Low Bit Encoding) que oculta la información en archivos de audio. Con la técnica Spread Spectrum se añade ruidos al azar a la señal de que la información se oculta dentro de la onda y la propagación en todo el espectro de frecuencias.
- **En vídeo:** suele utilizarse la técnica DCT (Discrete Cosine Transform) que cambia ligeramente cada una de las imágenes en el vídeo, sólo de manera que no sea perceptible por el ojo humano, altera los valores de ciertas partes de las imágenes.
- **En archivos de cualquier tipo:** se utiliza el método de inyección o agregado que consiste en agregar al final de un archivo (de cualquier tipo), otro archivo que será el contenedor del "mensaje a ocultar" (de cualquier tipo). (**CASTILLO, CASTILLO, & NÚÑEZ**, 2013 , http://ccns.jimdo.com/encriptaci%C3%B3n-de-datos/)

La criptografía y la esteganografía pueden complementarse, dando un nivel de seguridad adicional a la información, ya que el mensaje a embeber está previamente cifrado, de tal modo que en caso de que exista un eventual intruso no sólo le costará advertir la presencia misma del mensaje oculto, sino que, si la llegara a extraerlo, lo encontraría cifrado. **(COMUNIDAD GITS INFORMÁTICA**, 2003 , http://www.gitsinformatica.com/descargas/Esteganografia.doc)

2.3.2. Técnicas esteganográficas en imágenes

Entre las principales técnicas criptográficas en imágenes (**LÓPEZ**, 2012 , http://www.unocero.com/2012/11/28/esteganografia-para-cifrar-mensajes-en-imagenes/), se mencionan:

- Enmascaramiento y filtrado
- Inserción de bits en el objeto contenedor
- Creación de un fichero contenedor propio partiendo de la información a ocultar
- Bit Menos Significativo (Least Significant Bit - LSB)

2.3.2.1. Enmascaramiento y filtrado

La técnica de enmascaramiento y filtrado se basa en ocultar marcas de agua que incluyen información dentro de una imagen digital, como el derecho de autor, la propiedad o licencias, entre otros.

La marca de agua digital consiste en insertar un mensaje (un grupo de bits que contiene información sobre el autor por propietario intelectual) en el interior de un objeto digital, con el objetivo de mostrar que el uso del servicio digital por parte de un usuario no autorizado no es legal. (**MORENO**, 2010 , http://www.securityartwork.es/2010/05/03/introduccion-a-la-esteganografia-ii/)

2.3.2.2. Inserción de bits en el objeto contenedor

La técnica de inserción de bits en el objeto contenedor, se basa en añadir los bits de información a partir de una determinada marca estructural del fichero: fin de fichero, espacios de alineamiento, entre otros. El problema de esta técnica es que se incrementa el tamaño del fichero contenedor. (**COMUNIDAD EXPRESIÓN BINARIA**, 2012 , http://www.expresionbinaria.com/el-arte-de-ocultar-informacion-esteganografia/)

2.3.2.3. Creación de un fichero contenedor propio partiendo de la información a ocultar

La técnica de creación de un fichero propio partiendo de la información ocular consiste en crear un fichero contenedor con la propia información que se quiere ocultar. Por ejemplo, dado un algoritmo específico de reordenamiento de los bytes de los datos a ocultar se puede generar una secuencia de pixeles de un archivo BMP que tenga cierto significado visual (**COMUNIDAD EXPRESIÓN BINARIA**, 2012 , http://www.expresionbinaria.com/el-arte-de-ocultar-informacion-esteganografia/).

2.3.2.4. Bit Menos Significativo (Least Significant Bit - LSB)

El algoritmo esteganográfico más utilizado en imágenes es el LSB (Least Significant Bit) que es una estrategia sencilla de implementar en la esteganografía, incrusta los datos en la cubierta de modo que no puede ser detectado por un observador casual. La técnica funciona mediante la

sustitución de parte de la información en un píxel dado con la información de los datos de la imagen. (**MILLER**, 2012 , http://www.aaronmiller.in/thesis/)

Para un computador un archivo de imagen es un archivo que muestra diferentes colores e intensidades de luz en diferentes áreas (pixeles). El formato de imagen más apropiado para ocultar información es el BMP color de 24 bit Bitmap), debido a que es el de mayor proporción (imagen no comprimida) y normalmente es de la más alta calidad. Eventualmente se prefiere optar por formatos BMP de 8 bits o bien otros tales como el GIF, por ser de menor tamaño. Cuando una imagen es de alta calidad y resolución, es más fácil y eficiente ocultar y enmascarar la información dentro de ella. (**COMUNIDAD GITS INFORMÁTICA**, 2003 , http://www.gitsinformatica.com/descargas/Esteganografia.doc)

La incrustación LSB se realiza en el bit menos significativo, esto minimiza la variación en los colores que crea la incrustación, De esta forma la distorsión de la imagen en general se mantiene al mínimo (la perceptibilidad es prácticamente nula), mientras que el mensaje es esparcido a lo largo de sus píxeles. (**CASTILLO, CASTILLO, & NÚÑEZ**, 2013 , http://ccns.jimdo.com/encriptaci%C3%B3n-de-datos/)

Esta técnica funciona mejor cuando el archivo de imagen es grande, posee fuertes variaciones de color (imagen ruidosa) y también aventaja cuanto mayor sea la profundidad de color. De igual forma, esta técnica puede utilizarse eficazmente en imágenes a escala de gris, pero no es apropiada para aquellas en color de 8 bit paletizadas (misma estructura que las de escalas de gris, pero con paleta en color). En general, los mejores resultados se obtienen en imágenes con formato de color RGB (tres bytes, componentes de color Rojo-Verde-Azul, por píxel). (**YOUTUBE**, 2015 , https://www.youtube.com/watch?v=qwIvN0fz5WE)

Además, este método no altera en absoluto el tamaño del archivo portador o cubierta (por eso es "una técnica de sustitución"). Posee la desventaja de que el tamaño del archivo portador debe ser mayor cuanto más grande sea el mensaje a embeber; se necesitan 8 bytes de imagen por cada byte de mensaje a ocultar. Si se desea utilizar una mayor porción de bits de la imagen (por ejemplo, no sólo el último, sino los dos últimos), puede comenzar a ser notorio al ojo humano la alteración general provocada. (**COMUNIDAD GITS INFORMÁTICA**, 2003 , http://www.gitsinformatica.com/descargas/Esteganografia.doc)

Proceso de la técnica LSB

El proceso de la técnica LSB se puede representar de forma genérica de la siguiente forma: en una imagen "A" en la que se desea ocultar un documento de texto "B", se crea la imagen esteganográfica "A + B", como se muestra en la Figura 31-2, se muestra la técnica LSB, en la que se indica como los bits menos significativos de la imagen "A + B" que contendrá todos los bits del documento B. (**PÉREZ**, 2012 , http://www.securityartwork.es/2012/02/24/ocultando-archivos-en-otros-lsb/)

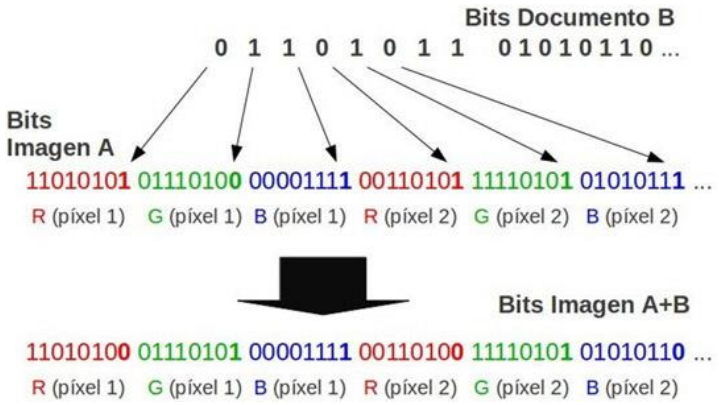


Figura 31-2 Proceso de la técnica LSB
Fuente: http://www.securityartwork.es/2012/02/24/ocultando-archivos-en-otros-lsb/

2.3.3. Determinación de la técnica esteganográfica en imágenes

Luego de realizar la búsqueda de información de estudios primarios acerca de los algoritmos esteganográficos más utilizados y detallarlos en el punto 2.3.2, se procede a seleccionar la técnica Least Significant Bit (LSB) debido a sus características:

- Sencillo de implementar
- Rápido
- Utiliza menos recursos
- Minimiza la variación en los colores.
- Distorsión de la imagen se mantienen al mínimo
- No varía el tamaño de la imagen
- Puede utilizarse en imágenes a color y escala de grises

2.4. Análisis y diseño de algoritmos

El análisis de algoritmos es una herramienta para hacer la evaluación del diseño de un algoritmo, permite establecer la calidad de un programa y compararlo con otros que puedan resolver el

mismo problema. El análisis de algoritmos estudia, desde un punto de vista teórico, los recursos computacionales que requiere la ejecución de un programa, es decir su eficiencia (tiempo de CPU, uso de memoria, ancho de banda, etc.). Además de la eficiencia en el desarrollo de software existen otros factores igualmente relevantes: funcionalidad, corrección, robustez, usabilidad, modularidad, mantenibilidad, fiabilidad, simplicidad y aún el propio costo de programación. (**DÍAZ**, 2004 , http://artemisa.unicauca.edu.co/~nediaz/EDDI/cap01.htm)

Determinar la eficiencia de un algoritmo permite establecer lo que es factible en la implementación de una solución de lo que es imposible.

Complejidad

La complejidad de tiempo de un algoritmo es igual para todas las instancias de tamaño n del problema. En otros casos, la complejidad de un algoritmo de tamaño n es distinta dependiendo de las instancias de tamaño n del problema que resuelve. Esto lleva a estudiar la complejidad del peor caso, caso promedio y mejor caso.

Para un tamaño dado (n):

- La complejidad del algoritmo en el **peor caso** resulta de tomar el máximo tiempo (complejidad máxima) en que se ejecuta el algoritmo, entre todas las instancias del problema (que resuelve el algoritmo) de tamaño n.
- La complejidad en el **caso promedio** es la esperanza matemática del tiempo de ejecución del algoritmo para entradas de tamaño n.
- La complejidad **mejor caso** es el menor tiempo en que se ejecuta el algoritmo para entradas de tamaño n. Por defecto se toma la complejidad del peor caso como medida de complejidad T(n) del algoritmo.

Dado que se realiza un estudio teórico de la complejidad, ignorando aspectos como las características de la máquina y el compilador, se tiene en cuenta que las diferencias en eficiencia se hacen más significativos para tamaños grandes de los datos de entrada se analiza la complejidad en términos de su comportamiento asintótico, dejando de lado la forma exacta de la función de complejidad.

Ordenes de complejidad

Se dice que O(f(n)) define un "orden de complejidad". Las funciones de complejidad algorítmica más habituales en las cuales el único factor del que dependen es el tamaño de la muestra de entrada n, las principales son: (**COMUNIDAD MONOGRAFÍAS**, 2014), http://www.monografias.com/trabajos27/complejidad-algoritmica/complejidad-algoritmica.shtml#ixzz3hc0D4wQF)

- O(1) orden constante
- O(log n) orden logarítmico
- O(n) orden lineal
- O(*n log n*) orden cuasi-lineal
- $O(n^2)$ orden cuadrático
- $O(n^a)$ orden polinomial (a > 2)
- $O(a^n)$ orden exponencial (a > 2)
- O(n!) orden factorial

En la Figura 32-2 se muestra la comparación de acuerdo a orden de complejidad. (**ALAÍZ, FERNANDEZ, & RODRÍGUEZ**, 2011 , http://arantxa.ii.uam.es/~aa/practicas/recursos/ordenesComplejidad.html)

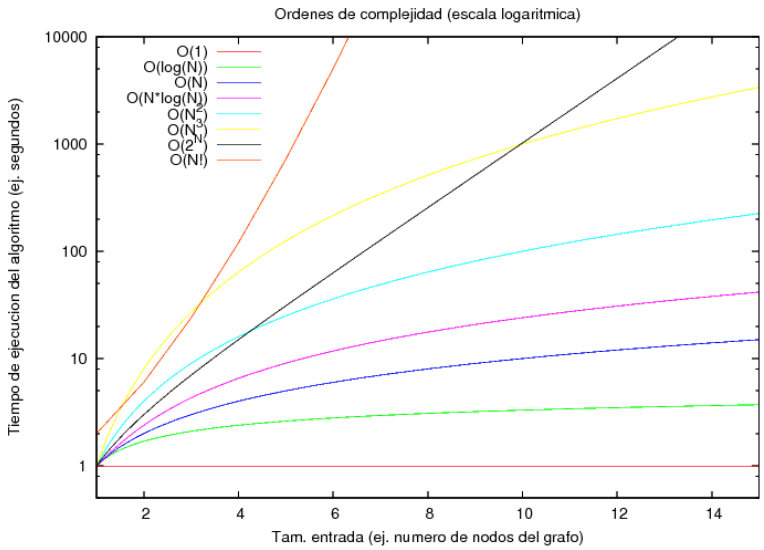


Figura 32-2 Comparativa de acuerdo el orden de complejidad
Fuente: http://arantxa.ii.uam.es/~aa/practicas/recursos/ordenesComplejidad.html

Complejidad y tiempo de procesamiento

El grado de complejidad de algunos algoritmos criptográficos en notación asintótica se muestra a continuación: (**BLANCO**, 2010 , http://itzamna.bnct.ipn.mx:8080/dspace/bitstream/123456789/6239/1/IF2.45.pdf)

- DES $O(n^2)$
- AES O(n)
- RSA O(n)
- RC4 O(n)

La complejidad está dada con respecto a cuantas operaciones se realizan sobre los bits del mensaje o clave, este criterio permite comparar y determinar su complejidad.

CAPÍTULO III

DISEÑO DE INVESTIGACIÓN

En este Capítulo, se detalla el tipo de investigación, diseño, métodos, técnicas, instrumentos con su respectiva validación, adicionalmente se crea e implementa el algoritmo criptográfico base, el nuevo algoritmo criptográfico, la técnica esteganográfica, los prototipos (integran los algoritmos criptográficos y la técnica esteganográfica) y se definen los escenarios de pruebas.

3.1. Tipo de investigación

La presente investigación puede clasificarse de dos tipos: aplicativa y experimental.

- **Aplicativa:** ya que se basa en conocimientos existentes, derivados de investigaciones previas, dirigida al desarrollo tecnológico para establecer nuevos procesos para mejorar los existentes.

- **Experimental:** ya que se basa en pruebas realizadas en escenarios de laboratorio, en las que se observa los elementos más importantes del objeto de estudio que se investiga para obtener una captación de los fenómenos a primera vista.

3.2. Diseño de la investigación

El diseño de la presente investigación es del tipo cuasi experimental ya que se escoge el algoritmo criptográfico que será utilizado como base para la creación del nuevo algoritmo criptográfico, al cual se incorporará la esteganografía en imágenes para mejorar la seguridad, además los datos de prueba son generados por el autor de esta investigación.

3.3. Métodos y técnicas

Los métodos y técnicas utilizados para la investigación son:

3.3.1. Métodos

La presente investigación utiliza el método científico que consta de varias etapas para obtener un conocimiento válido desde el punto de vista científico, utilizando para esto instrumentos que resulten fiables, consta de las siguientes etapas:

- Planteamiento del problema
- Formulación de la hipótesis
- Levantamiento de la información
- Análisis e interpretación de resultados
- Comprobación de la hipótesis
- Difusión de resultados

3.3.2. Técnicas

Las técnicas que serán utilizadas en la presente investigación son:

- **Búsqueda de información:** permite obtener la información necesaria acerca del objeto de estudio de la investigación para su desarrollo, utilizando las fuentes secundarias disponibles.
- **Pruebas:** permite realizar experimentos en escenarios de laboratorio.
- **Observación:** permite determinar resultados de las pruebas realizadas en los escenarios de laboratorio.
- **Análisis:** permite determinar los resultados de la investigación.

3.4. Instrumentos

Los instrumentos para recopilar los datos de los indicadores son los siguientes:

- **Netbeans:** entorno de Desarrollo Integrado (IDE) de código abierto. Permite el desarrollo aplicación Java como por ejemplo: J2SE, web, EJB y aplicaciones móviles (**COMUNIDAD NETBEANS**, 2015 , https://www.netbeans.org)
- **FlexHEX:** editor hexadecimal, herramienta diseñada para editar archivos binarios, archivos compuestos OLE, dispositivos lógicos y unidades físicas. Con FlexHEX se puede inspeccionar, modificar, insertar, buscar, o reemplazar datos binarios, ASCII o UNICODE (**INV SOFTWORKS LLC**, 2015 , http://www.flexhex.com/)

- **Guiffy Image Diff:** herramienta que permite comparar las diferencias entre imágenes pixel a pixel. (**COMUNIDAD GUIFFY SOFTWARE**, 2014 , http://www.guiffy.com/Image-Diff-Tool.html)
- **Cryptool:** herramienta gratuita de e-learning sobre criptografía y criptoanálisis, tanto clásico como moderno. (**DEUTSCHE BANK & COLABORADORES**, 2015 , https://www.cryptool.org/en/cryptool1-en)

3.5. Validación de instrumentos

Los instrumentos software que han sido utilizados en la investigación fueron seleccionados debido a sus características y ventajas que se mencionan a continuación:

FlexHEX

Figura 1-3 Logo de FlexHEX
Fuente: http://www.flexhex.com/

Se ha escogido el instrumento FlexHEX debido a sus características y ventajas, entre las que se mencionan: (**INV SOFTWORKS LLC**, 2015 , http://www.flexhex.com/)

- Gestor de archivos binarios, hexadecimales, archivos compuestos OLE, dispositivos lógicos, unidades físicas.
- Permite inspeccionar, modificar, insertar, buscar o reemplazar datos binarios, hexadecimal, ASCII o UNICODE.
- Posee funciones de navegación y seguimiento de gran alcance que la navegación de datos binarios fácil y conveniente.
- Permite una navegación de archivos sencilla
- Proporciona soporte completo para las características de NTFS avanzadas, tales como secuencias alternativas o archivos dispersos.

En la investigación se lo utiliza para comparar de forma visual las diferencias en el código hexadecimal de las imágenes esteganografiadas que ocultan los mensajes cifrados con los 2 Prototipos desarrollados.

Guiffy Image Diff

Figura 2-3 Logo de Guiffy Image Diff
Fuente: http://www.guiffy.com/Image-Diff-Tool.html

Se ha escogido el instrumento Guiffy Image Diff debido a sus características y ventajas, entre las que se mencionan: (**COMUNIDAD GUIFFY SOFTWARE**, 2014 , http://www.guiffy.com/Image-Diff-Tool.html)

- Compara archivos de imágenes en formatos BMP, GIF, JPEG, JPG, PNG, y WBMP.
- Posee 3 opciones de filtro: B & W, Sombras, Calor
- Destaca las diferencias según filtro de las zonas Grises opcionalmente Overylay.
- Control de Umbral - especifica ciento de diff por píxel
- Cambio de tamaño de imagen (zoom, mejor ajuste, tamaño real)
- Compara métricas (Píxeles DIF> Umbral%, diff color%)
- Información de archivo de imagen del panel: tipo de archivo, tamaño en bytes, formato, profundidad en bits, ancho en píxeles, la altura en píxeles, resolución horizontal en dpi y resolución vertical en dpi
- Interfaz de línea de comandos

En la investigación se lo utiliza para comparar de forma visual las diferencias pixel a pixel de las imágenes esteganografiadas que ocultan los mensajes cifrados con los 2 Prototipos desarrollados.

Cryptool

Figura 3-3 Logo de Cryptool
Fuente: https://www.cryptool.org/en/cryptool1-en

Se ha escogido el instrumento Cryptool 1 (CT1) debido a sus características y ventajas, entre las que se mencionan: (**DEUTSCHE BANK & COLABORADORES**, 2015 , https://www.cryptool.org/en/cryptool1-en)

- Programa de código abierto gratuito de Windows para la criptografía y el criptoanálisis.
- Software de e-learning, compatible tanto con los métodos actuales de enseñanza en escuelas y universidades.

- Numerosos algoritmos criptográficos clásicos y modernos (cifrado y descifrado, generación de clave, contraseñas seguras, autentificación, protocolos seguros, etc.)
- Visualización de varios algoritmos (César, enigma, RSA, Diffie-Hellman, firmas digitales, AES, etc.)
- Criptoanálisis de varios algoritmos (vigenére RSA, AES, etc.)
- Métodos de medición de criptoanálisis (entropía, n-gramas, autocorrelación, etc.)
- Métodos auxiliares relacionados (pruebas de primalidad, factorización, codificación base64, etc.)

En la investigación se lo utiliza para realizar pruebas de criptoanálisis a los mensajes cifrados ocultos en las imágenes esteganografiadas con los 2 Prototipos desarrollados para compararlos.

3.6. Implementación del algoritmo criptográfico base

3.6.1. Desarrollo de la aplicación

Luego de la descripción del algoritmo criptográfico AES que ha sido considerado como base en el Capítulo II, se procede con el desarrollo de la aplicación, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

El la Figura 4-3 se muestran los paquetes de la aplicación

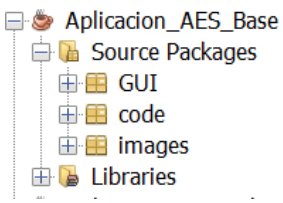


Figura 4-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** paquete de clases desarrolladas necesarias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 5-3 se muestran las clases de cada paquete.

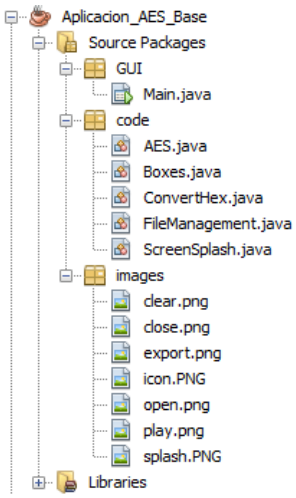


Figura 5-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **GUI** contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 6-3.

Figura 6-3 Interfaz de usuario de la aplicación Algoritmo AES base
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **AES.java (class):** clase que contiene atributos y métodos del algoritmo AES, entre las principales se encuentran**:**

  - Expansión de clave
  - Subword
  - Rotword
  - Cifrado
  - AddRoundKey
  - SubByte
  - ShiftRow
  - MixColumn
  - Descifrado
  - InvShiftRow
  - InvSubByte
  - InvMixColumn
  - InvSubWord

- **Boxes.java (class):** clase que contiene métodos para obtener las Cajas que utiliza el algoritmo AES, entre las principales se encuentran**:**

  - S-Box
  - InvS-Box
  - Rcon

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - Hexadecimal a ASCII
  - ASCII a hexadecimal
  - Hexadecimal a byte
  - Byte a hexadecimal

- **FileManagement.java (class):** clase que contiene atributos y métodos para manejar archivos que utiliza la aplicación, entre las principales se encuentran**:**

- Abrir archivo
- Leer archivo
- Guardar archivo

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 7-3.

Figura 7-3 Splash Aplicación Algoritmo AES base
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizada en el splash y los íconos para la aplicación.

En el Anexo A se detalla el código fuente principal para la implementación del algoritmo criptográfico base.

A continuación, se detallan los procesos de cifrado y de descifrado del algoritmo criptográfico:

3.6.2. Proceso de cifrado

El proceso de cifrado del algoritmo AES base, se muestra en la Figura 8-3.

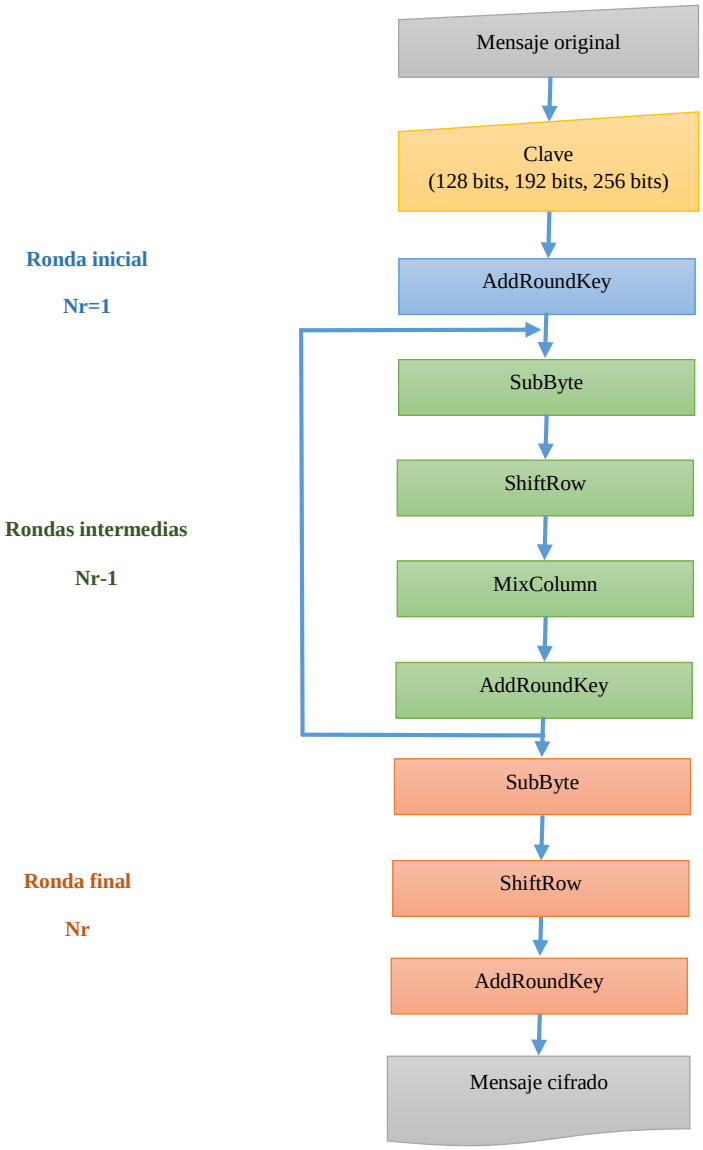


Figura 8-3 Proceso de cifrado del algoritmo AES base
Realizado por: Méndez Pablo, 2015

Ejecución de cifrado con la aplicación AES base

Para probar la aplicación AES base en la que se han creado las funciones requeridas por el algoritmo criptográfico AES base en el proceso de cifrado, se utilizan los datos de la Tabla 1-3.

Tabla 1-3 Datos para ejecución de la aplicación desarrollada para cifrado

Archivo de entrada:	Origen.txt
Clave (128 bits):	}yl%OckF x){gI~o
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el algoritmo criptográfico AES en el proceso de cifrado, se obtiene el siguiente resultado:

Abrir el archivo de texto que contiene el mensaje original denominado “Origen.txt” que se muestra en la Figura 9-3.

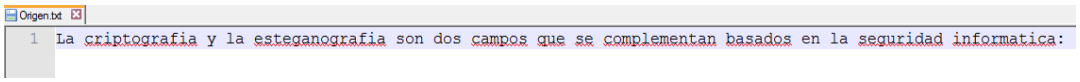


Figura 9-3 Captura del archivo de texto “Origen.txt”
Realizado por: Méndez Pablo, 2015

Se ejecuta el proceso de cifrado AES con el mensaje que se desea cifrar, como se muestra en la Figura 10-3.

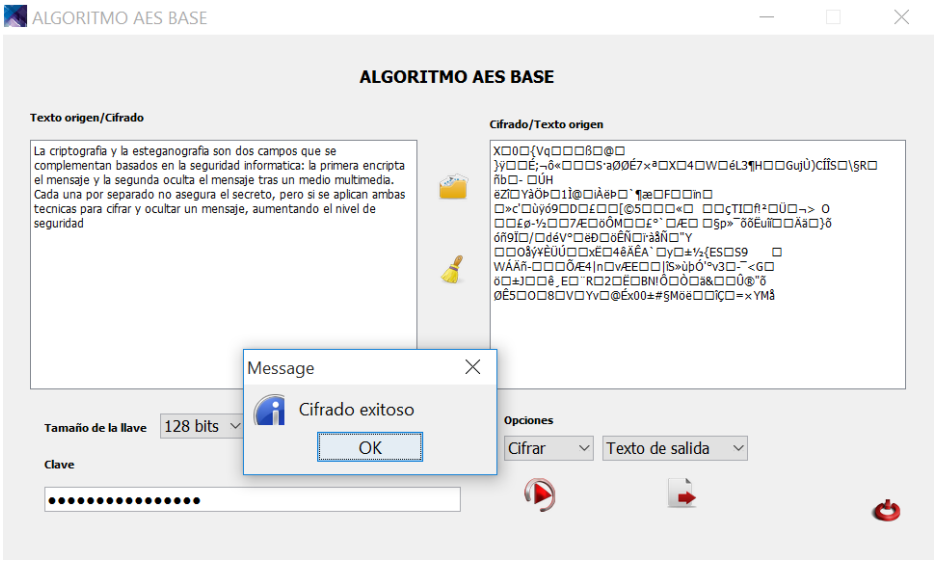


Figura 10-3 Captura del resultado de cifrado de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se exporta el mensaje cifrado a un archivo denominado “Cifrado.txt”, como se muestra en la Figura 11-3.

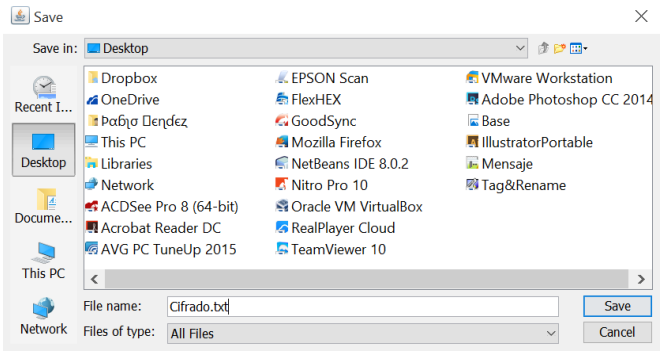


Figura 11-3 Captura de interfaz para exportación del mensaje cifrado
Realizado por: Méndez Pablo, 2015

Se obtiene el archivo con el mensaje cifrado, como se muestra en la Figura 12-3.

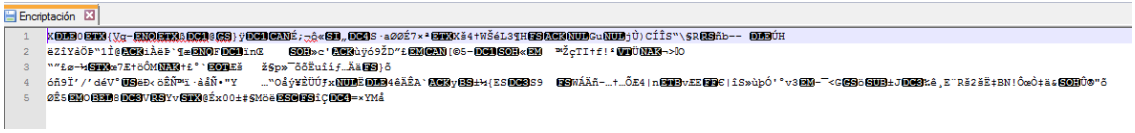


Figura 12-3 Captura del archivo exportado con el mensaje cifrado
Realizado por: Méndez Pablo, 2015

3.6.3. Proceso de descifrado

El proceso de descifrado del algoritmo AES base, se muestra en la Figura 13-3.

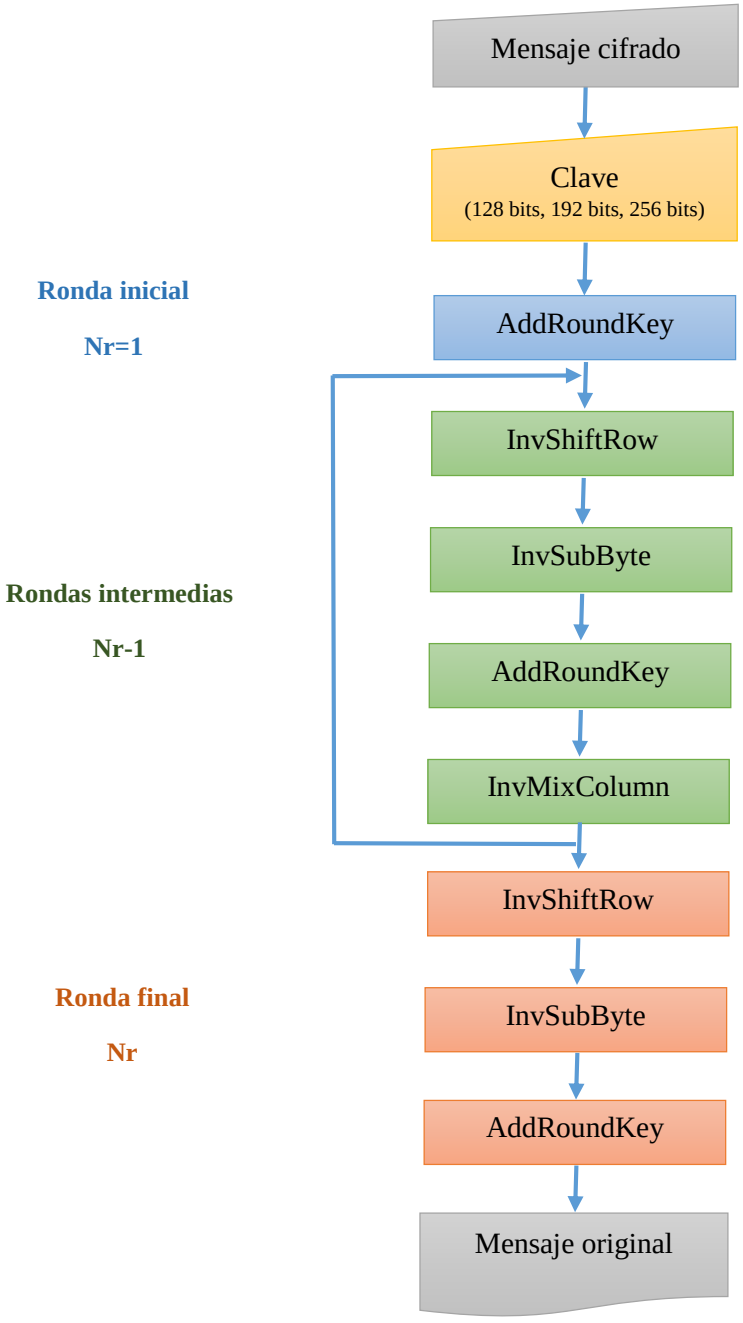


Figura 13-3 Proceso de descifrado del algoritmo AES base
Realizado por: Méndez Pablo, 2015

Ejecución de descifrado con la aplicación AES base

Para probar la aplicación desarrollada creando las funciones requeridas por el algoritmo criptográfico AES en el proceso de descifrado, se utilizan los datos de la Tabla 2-3.

Tabla 2-3 Datos para ejecución de la aplicación desarrollada para descifrado

Archivo de entrada:	Encriptación.txt
Clave (128 bits):	}yl%OckF x){gI~o
Mensaje cifrado:	X□0□{Vq□□□□ß□@□ }ý□□É;¬õ«□□□S·aØØÉ7×ª□X□4□W□éL3¶H□□GujÙ)CÍÎS□\§R□ ñb□- □ÚH ëZî□YàÖÞ□1Î@□íÀëÞ□`¶æ□F□□in□ □»c'□ùý69□D□£□□[©5□□□«□ □□çTI□fî²□Ú□¬> O □□£ø-½□□7Æ□óÔM□□£°`□Æ□ □§p»¯õõÉuíî□□Äã□}õ óñ9Î□/□déV°□ëÐ□óÊÑ□ïàåÑ□"Y □□Oåÿ¥ÈÚÚ□□xÉ□4ëÄÊA`□y□±½{ES□S9 □ WÁÄñ-□□□ÕÆ4|n□vÆE□□|Ss»ùþÓ'°v3□-¯<G□ ó□±J□□ê¸E□`R□2□É□BN!Ô□Ô□ä&□□Û®"õ ØÊ5□O□8□V□Yv□@Êx00±#§Möë□□Ïç□=×YMå

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el algoritmo criptográfico AES en el proceso de descifrado, se obtiene el siguiente resultado:

Abrir el archivo de texto que contiene el mensaje original denominado “Cifrado.txt” que se muestra en la Figura 14-3.

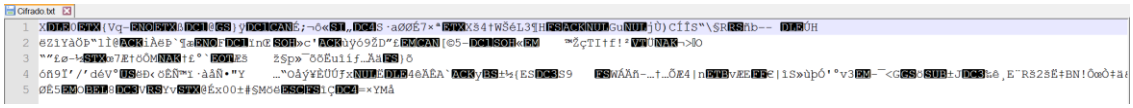


Figura 14-3 Captura del archivo de texto “Cifrado.txt”

Realizado por: Méndez Pablo, 2015

Se ejecuta el proceso de descifrado AES con el mensaje cifrado para obtener el mensaje original, como se muestra en la Figura 15-3.

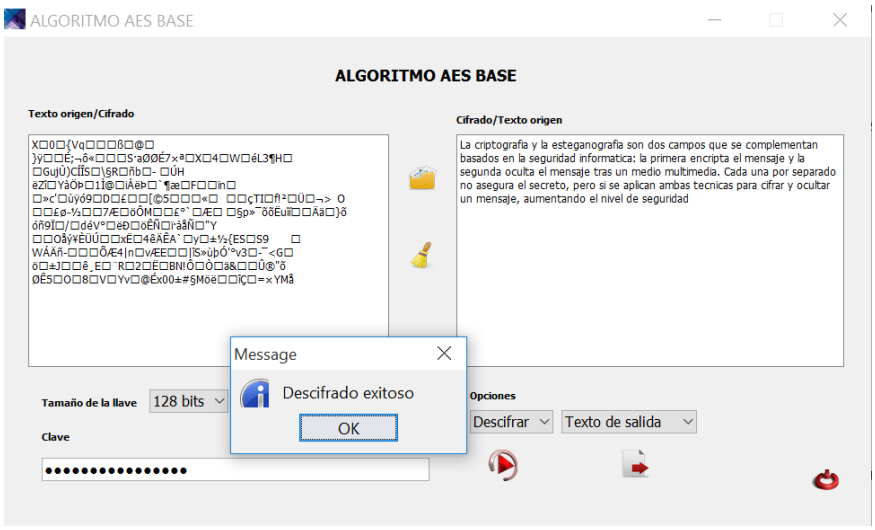


Figura 15-3 Captura del resultado de descifrado de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se exporta el mensaje descifrado a un archivo denominado “Descifrado.txt”, como se muestra en la Figura 16-3.

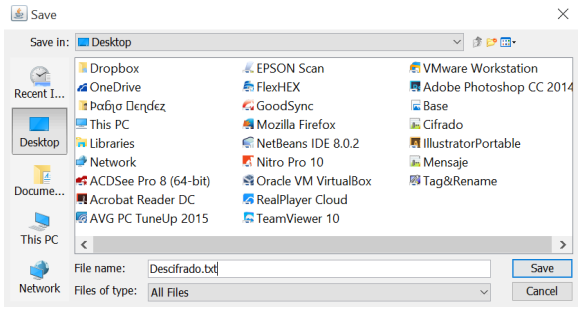


Figura 16-3 Captura de interfaz para exportación del mensaje descifrado
Realizado por: Méndez Pablo, 2015

Se obtiene el archivo con el mensaje descifrado, como se muestra en la Figura 17-3.

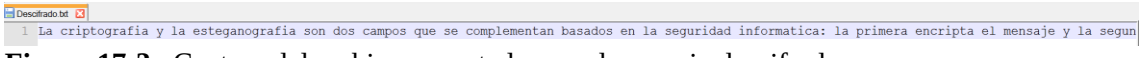


Figura 17-3 Captura del archivo exportado con el mensaje descifrado
Realizado por: Méndez Pablo, 2015

3.6.4. Validación del algoritmo

Para verificar que la aplicación desarrollada cifra/descifra el mensaje original comprobamos los resultados de la aplicación desarrollada en la que se han implementado todos los métodos necesarios por el algoritmo AES con una aplicación que utiliza las siguientes librerías para el cifrado del algoritmo AES:

- javax.crypto.Cipher
- javax.crypto.SecretKey
- javax.crypto.spec.SecretKeySpec

3.6.4.1. Desarrollo de la aplicación

Se procede con el desarrollo de la aplicación con librerías establecidas, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

El la Figura 18-3 se muestran los paquetes de la aplicación

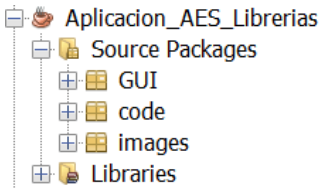


Figura 18-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** paquete de clases desarrolladas necesarias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 19-3 se muestran las clases de cada paquete.

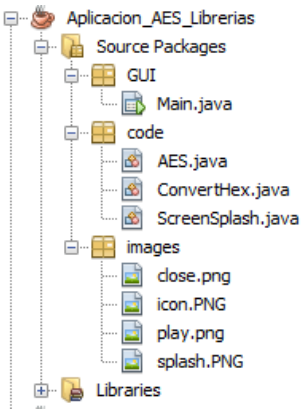


Figura 19-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **GUI** contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 20-3.

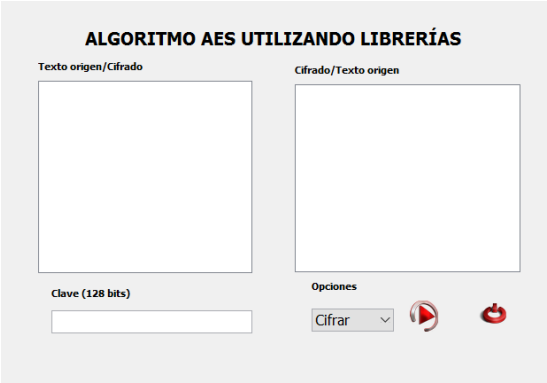


Figura 20-3 Interfaz de usuario de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **AES.java (class):** clase que contiene atributos y métodos del algoritmo AES, entre las principales se encuentran**:**

  - Cifrado
  - Descifrado

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - Hexadecimal a ASCII

- ASCII a hexadecimal

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 21-3.

Figura 21-3 Splash Aplicación Algoritmo con librerías
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizada en el splash y los íconos para la aplicación.

Aplicación con librerías definidas

Cifrado

Para probar la aplicación con librerías definidas y la aplicación desarrollada para cifrado AES para la verificación, se utilizan los datos de la Tabla 3-3.

Tabla 3-3 Datos para ejecución de la aplicación con librerías definidas para cifrado

Clave (128 bits):	}yl%OckF x){gI~o
Mensaje original:	AV7l5E85Z3lJ2VHK
Mensaje cifrado	èß□é1□□ì□ÙâE]1Y?

Realizado por: Méndez Pablo, 2015

Resultados

Aplicación con librerías

Luego de ejecutar el proceso de cifrado con la aplicación con librerías definidas se obtiene el resultado mostrado en la Figura 22-3.

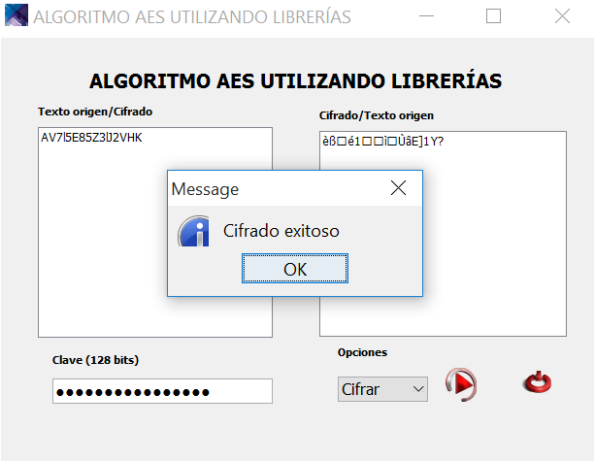


Figura 22-3 Resultado de cifrado con la aplicación con librerías
Realizado por: Méndez Pablo, 2015

Aplicación AES base

Luego de ejecutar el proceso de cifrado con la aplicación AES base desarrollado, se obtiene el resultado mostrado en la Figura 23-3.

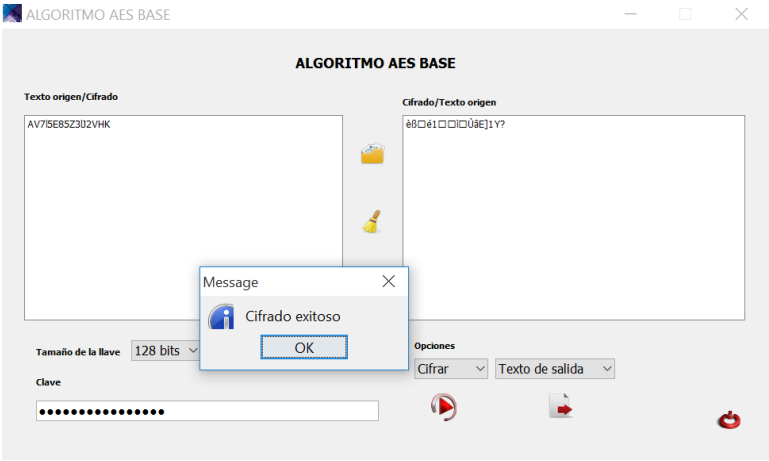


Figura 23-3 Resultado de cifrado con la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Descifrado

Para probar la aplicación con librerías definidas para descifrado AES para la verificación, se utilizan los datos de la Tabla 4-3.

Tabla 4-3 Datos para ejecución de la aplicación con librerías definidas para descifrado

Clave (128 bits):	}yl%OckF x){gI~o
Mensaje cifrado:	èß□é1□□i□ÙâE]1Y?
Mensaje original	AV7l5E85Z3lJ2VHK

Realizado por: Méndez Pablo, 2015

Resultados

Aplicación con librerías

Luego de ejecutar el proceso de descifrado la aplicación con librerías definidas se obtiene el resultado mostrado en la Figura 24-3.

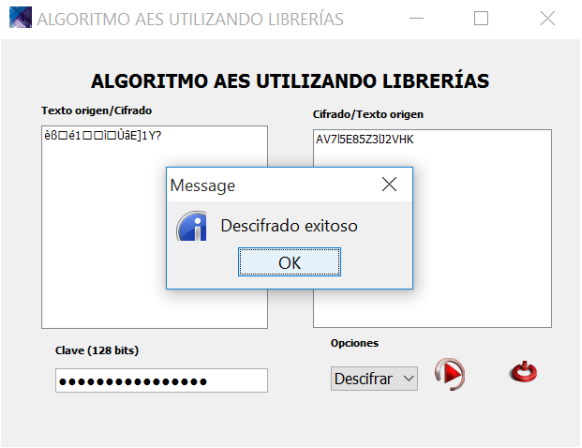


Figura 24-3 Resultado de descifrado con la aplicación con librerías
Realizado por: Méndez Pablo, 2015

Aplicación AES base

Luego de ejecutar el proceso de descifrado la aplicación AES base desarrollado, se obtiene el resultado mostrado en la Figura 25-3.

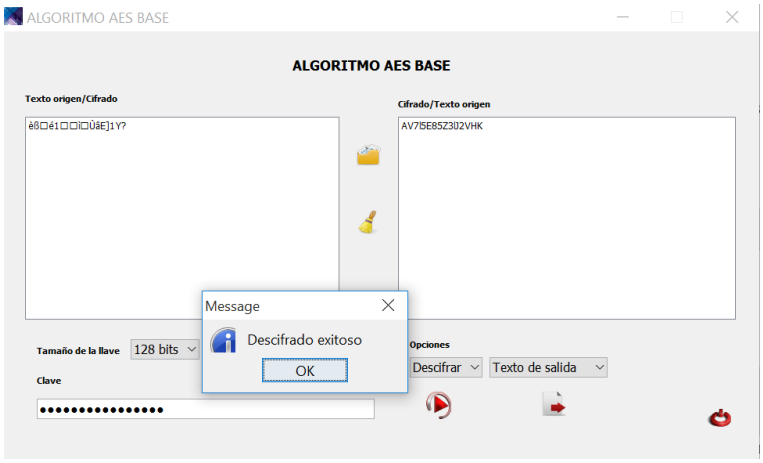


Figura 25-3 Resultado de descifrado con la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se comprueba que la aplicación desarrollada cifra/descifra el mensaje de forma correcta con algoritmo AES.

3.7. Creación e implementación del nuevo algoritmo criptográfico

3.7.1. Propuesta de mejora

Para la creación del nuevo algoritmo criptográfico se ha considerado el algoritmo AES como base y se lo denominará 2NAES (2 New Advanced Encryption Standard).

Para mejorar la seguridad e incrementar la difusión del mensaje, se proponen las siguientes mejoras en el algoritmo criptográfico:

- Utilizar una **nueva función** que se ejecute en todas las rondas (ronda inicial, dentro de las rondas parciales y en la ronda final) denominado **SHIFTCOLUMN**, en la cual se procede a realizar un desplazamiento hacia arriba cíclicamente de las columnas que conforman la matriz de estado actual. Cada columna se desplaza un número de posiciones diferentes. Como se muestra en la Figura 26-3, los bytes en cada columna del Estado son rotados de manera cíclica hacia arriba.

  El número de lugares que cada byte es rotado difiere para cada columna:
  - La primera columna no sufre cambio
  - La segunda columna rota hacia arriba una posición
  - La tercera columna rota hacia arriba dos posiciones
  - La cuarta columna rota hacia arriba tres posiciones

En la Figura 26-3 se muestra la función ShiftColumn propuesta.

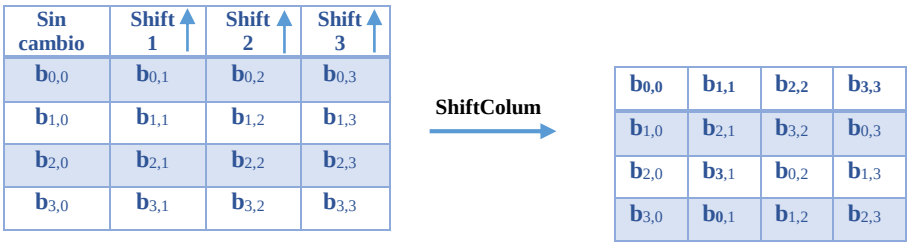


Figura 26-3 Función ShiftColumn
Realizado por: Méndez Pablo, 2015

Ejemplo

En la Figura 27-3 se muestra el ejemplo de la función ShiftColumn propuesta.

Sin cambio	Shift 1 ↑	Shift 2 ↑	Shift 3 ↑
32	88	31	e0
43	5a	32	37
f6	30	98	07
a8	8d	a2	34

ShiftColum →

32	5a	98	34
43	30	a2	e0
f6	8d	31	37
a8	88	32	07

Figura 27-3 Ejemplo de función ShiftColumn
Realizado por: Méndez Pablo, 2015

- Ejecutar 2 veces el algoritmo, para la primera ejecución se utiliza la clave ingresada por el usuario (clave A) y para la segunda ejecución se utiliza una segunda clave basada en la primera (clave B). Para obtener la clave B se realiza el siguiente proceso:

  - Se obtiene la clave ingresada por el usuario (clave A).
  - Se invierte el orden de elementos de la clave A.
  - Se realiza un ShiftRow de 3 posiciones hacia la izquierda de la clave A.
  - Cada valor decimal de la clave A, se suma la posición en la que se encuentre.

Ejemplo:

- El usuario ingresa la clave A de 128 bits, que es }yl%OckF x){gI~o, cuyo código decimal se muestra en la Figura 28-3.

125	121	108	37	79	99	107	70	32	120	41	123	103	73	126	111
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figura 28-3 Ejemplo matriz de código decimal de la clave A
Realizado por: Méndez Pablo, 2015

- Se invierte el orden de elementos de la clave A, como se muestra en la Figura 29-3.

111	126	73	103	123	41	120	32	70	107	99	79	37	108	121	125
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figura 29-3 Ejemplo matriz de código decimal de la clave A
Realizado por: Méndez Pablo, 2015

- Se realiza un ShiftRow de 3 posiciones hacia la izquierda de la clave A, como se muestra en la Figura 30-3.

103	123	41	120	32	70	107	99	79	37	108	121	125	111	126	73
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figura 30-3 Ejemplo matriz invertida de la clave A
Realizado por: Méndez Pablo, 2015

- A cada valor decimal de la clave A se suma la posición en la que se encuentra, obteniendo la clave B de 128 bits, como se muestra en la Figura 31-3.

103	124	43	123	36	75	113	106	87	46	118	132	137	124	140	88
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figura 31-3 Ejemplo matriz de código decimal de la clave B
Realizado por: Méndez Pablo, 2015

De esta forma se crea un algoritmo que se denomina 2NAES con 2 claves, que para este ejemplo son de 128 bits:

- **Clave A:** }yl%OckF x){gI~o
- **Clave B:** g|+{$KqjW.v,,‰|Œ

El proceso resumido de cifrado se muestra en la Figura 32-3 y el proceso resumido de descifrado de muestra en la Figura 33-3.

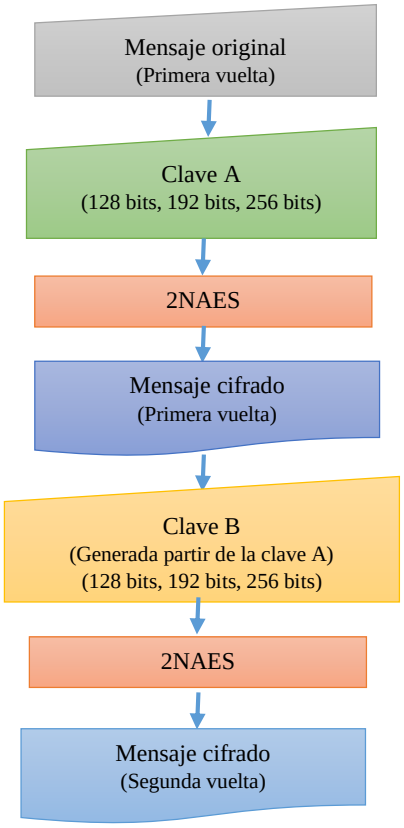


Figura 32-3 Nuevo algoritmo de cifrado 2NAES
Realizado por: Méndez Pablo, 2015

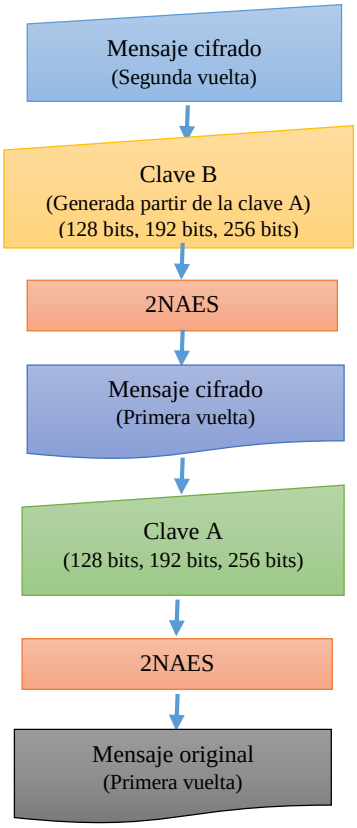


Figura 33-3 Nuevo algoritmo de descifrado 2NAES
Realizado por: Méndez Pablo, 2015

3.7.2. Desarrollo de la aplicación

Luego de la definición de las modificaciones propuestas para el nuevo algoritmo criptográfico 2NAES, se procede con el desarrollo de la aplicación, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

En la Figura 34-3 se muestran los paquetes de la aplicación

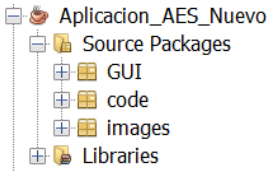


Figura 34-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** paquete de clases desarrolladas necesarias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 35-3 se muestran las clases de cada paquete.

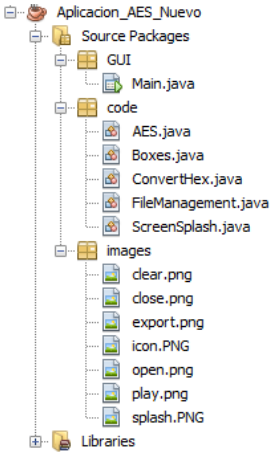


Figura 35-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **GUI** contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 36-3.

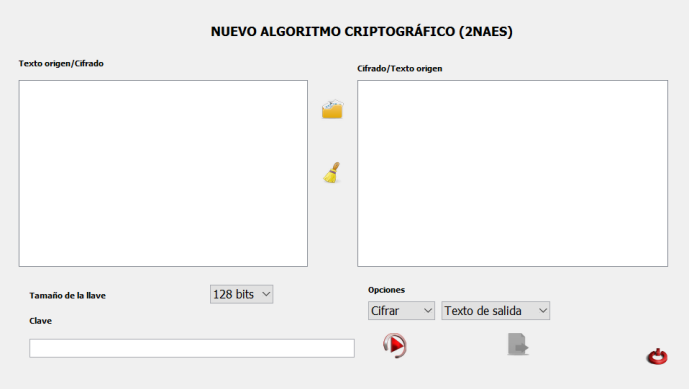


Figura 36-3 Interfaz de usuario de la aplicación Nuevo Algoritmo Criptográfico (2NAES)
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **AES.java (class):** clase que contiene atributos y métodos del algoritmo AES, entre las principales se encuentran**:**

  - Expansión de clave
  - Subword
  - Rotword
  - Cifrado
  - AddRoundKey
  - SubByte
  - ShiftRow
  - **ShiftColumn**
  - MixColumn
  - Descifrado
  - InvShiftRow
  - **InvShiftColumn**
  - InvSubByte
  - InvMixColumn
  - InvSubWord
  - **Generación de clave**

- **Boxes.java (class):** clase que contiene métodos para obtener las Cajas que utiliza el algoritmo AES, entre las principales se encuentran**:**

  - S-Box
  - InvS-Box
  - Rcon

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - Hexadecimal a ASCII
  - ASCII a hexadecimal
  - Hexadecimal a byte
  - Byte a hexadecimal

- **FileManagement.java (class):** clase que contiene atributos y métodos para manejar archivos que utiliza la aplicación, entre las principales se encuentran**:**

  - Abrir archivo
  - Leer archivo
  - Guardar archivo

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 37-3.

Figura 37-3 Splash Aplicación Nuevo algoritmo 2NAES
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizada en el splash y los íconos para la aplicación.

En el Anexo A se detalla el código fuente principal para la creación e implementación del nuevo algoritmo criptográfico.

A continuación, se detallan los procesos de cifrado y de descifrado del nuevo algoritmo criptográfico 2NAES:

3.7.3. Nuevo proceso de cifrado

El nuevo proceso de cifrado del nuevo algoritmo (2NAES), se muestra en la Figura 38-3.

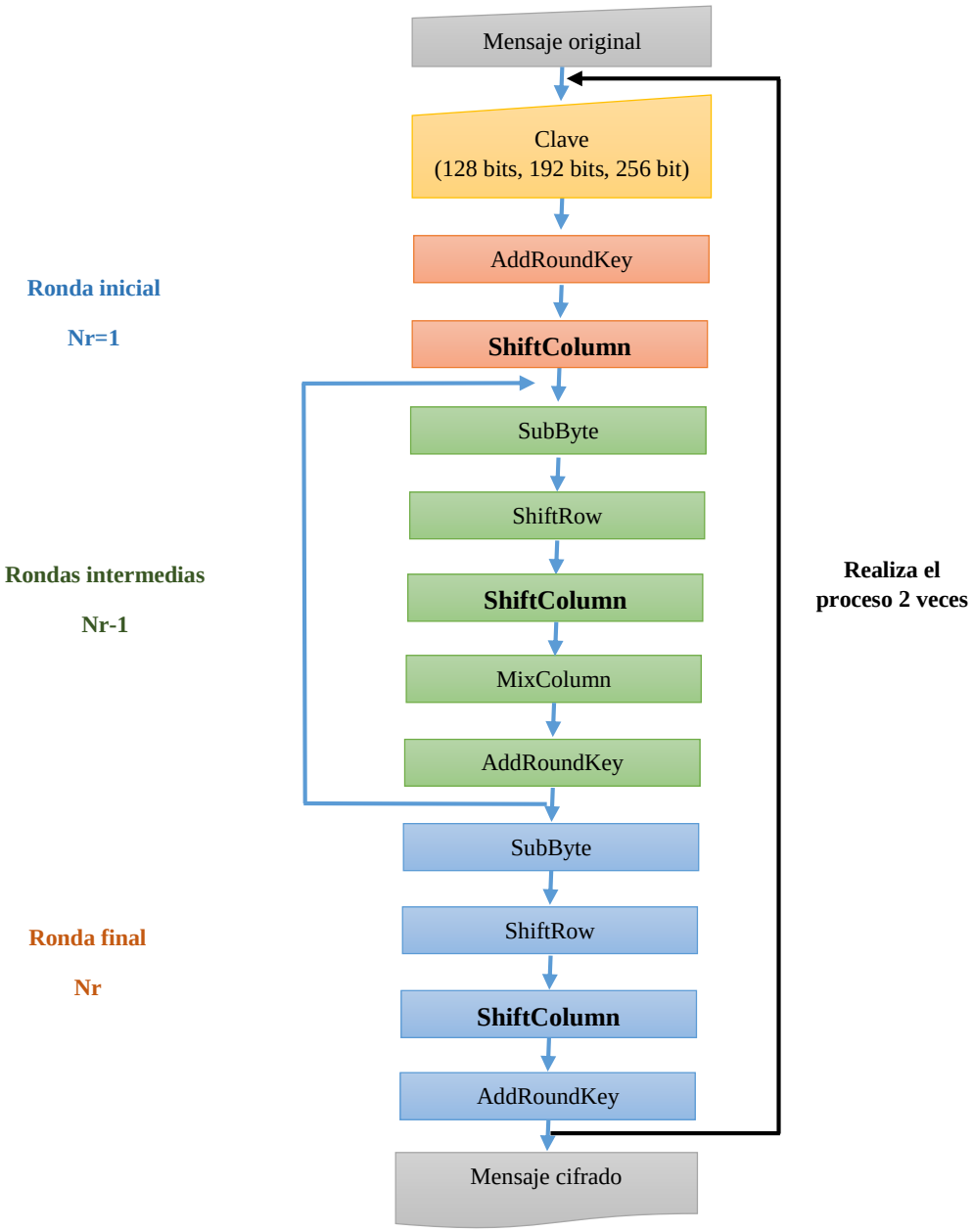


Figura 38-3 Proceso de cifrado del nuevo algoritmo 2NAES
Realizado por: Méndez Pablo, 2015

Ejecución de cifrado con la aplicación 2NAES

Para probar la aplicación 2NAES en la que se han creado las funciones implementadas por el nuevo algoritmo criptográfico para el proceso de cifrado, se utilizan los datos de la Tabla 5-3.

Tabla 5-3 Datos para ejecución de la aplicación desarrollada para cifrado 2NAES

Archivo de entrada:	Origen.txt
Clave (128 bits):	}yl%OckF x){gI~o
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el algoritmo criptográfico 2NAES en el proceso de cifrado, se obtiene el siguiente resultado:

Abrir el archivo de texto que contiene el mensaje original denominado “Origen.txt” que se muestra en la Figura 39-3.

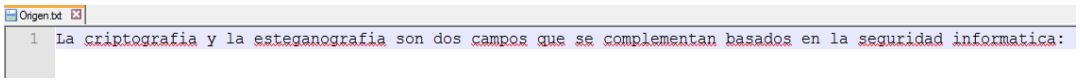


Figura 39-3 Captura del archivo de texto “Origen.txt”
Realizado por: Méndez Pablo, 2015

Se ejecuta el proceso de cifrado 2NAES con el mensaje que se desea cifrar, como se muestra en la Figura 40-3.

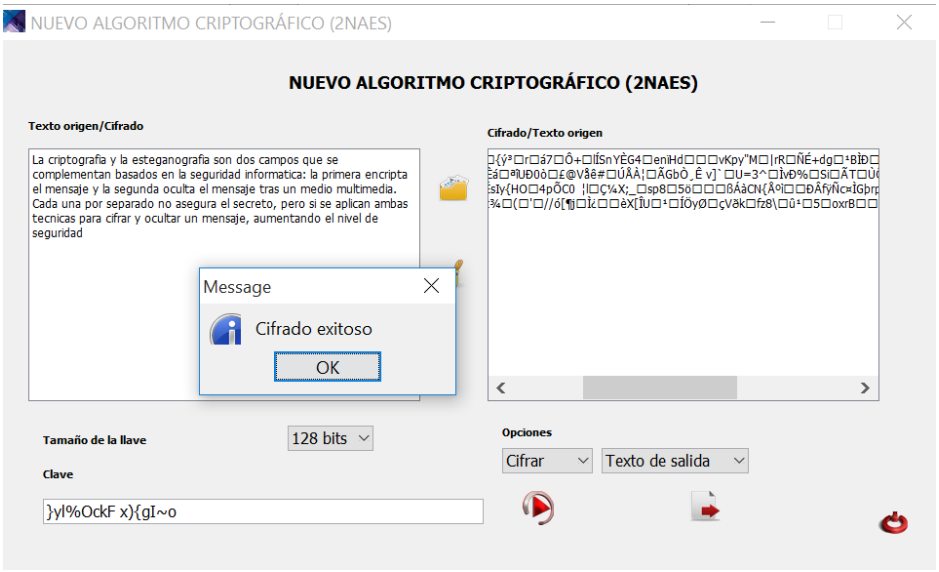


Figura 40-3 Captura del resultado de cifrado de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se exporta el mensaje cifrado a un archivo denominado “Cifrado.txt”, como se muestra en la Figura 41-3.

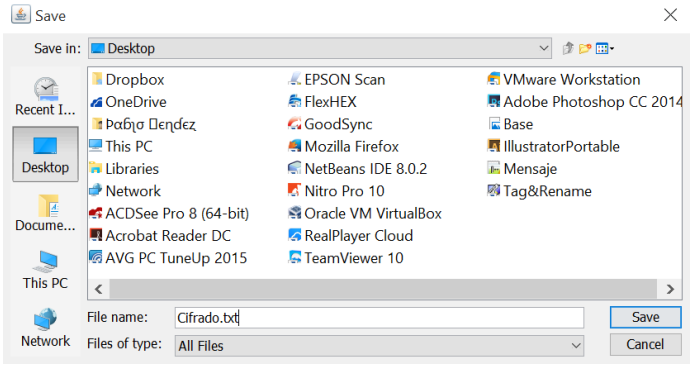


Figura 41-3 Captura de interfaz para exportación del mensaje cifrado
Realizado por: Méndez Pablo, 2015

Se obtiene el archivo con el mensaje cifrado, como se muestra en la Figura 42-3.

Figura 42-3 Captura del archivo exportado con el mensaje cifrado
Realizado por: Méndez Pablo, 2015

3.7.4. Nuevo proceso de descifrado

El nuevo proceso de descifrado del nuevo algoritmo (2NAES), se muestra en la Figura 43-3:

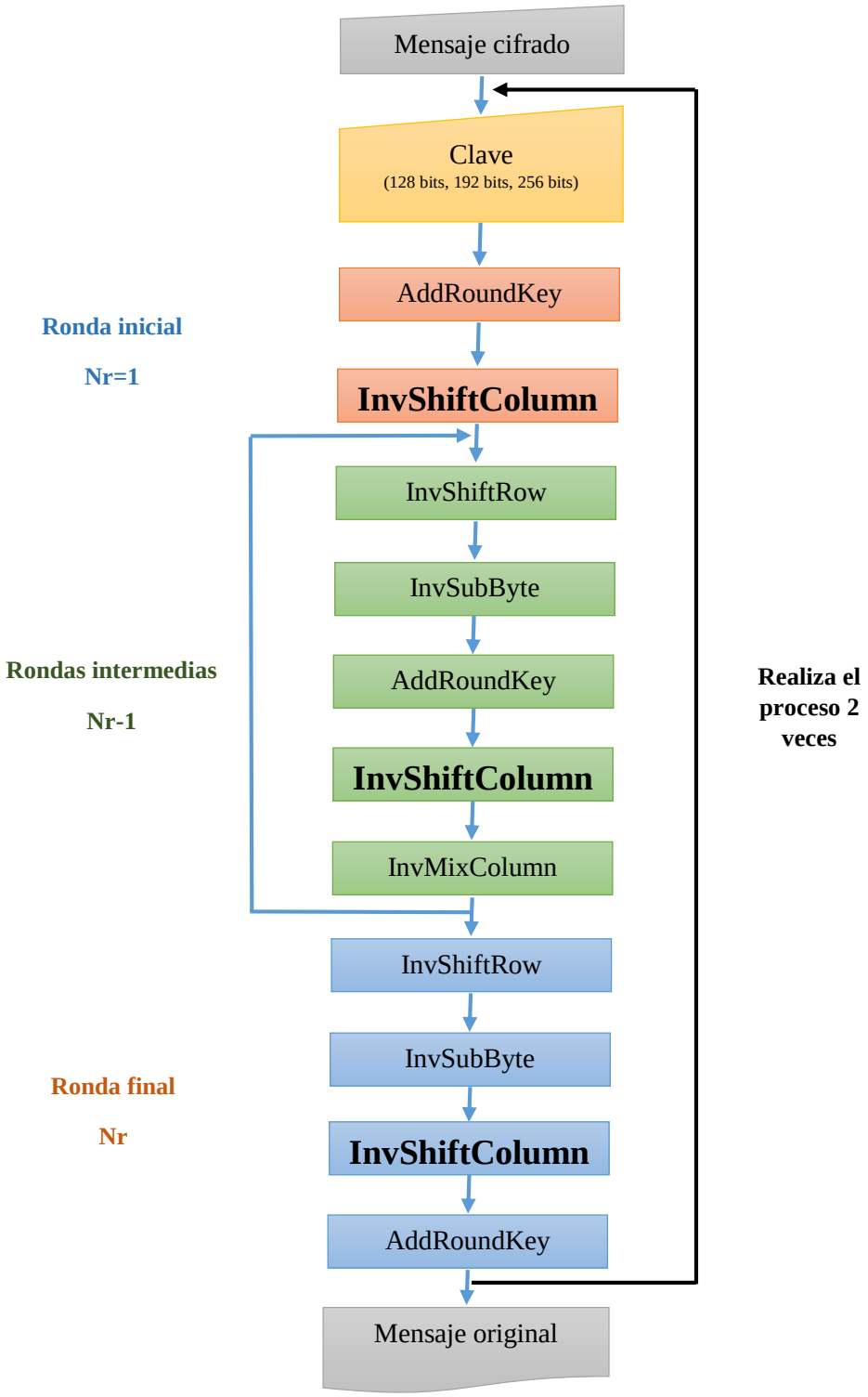


Figura 43-3 Proceso de descifrado del nuevo algoritmo 2NAES
Realizado por: Méndez Pablo, 2015

Ejecución de descifrado con la aplicación 2NAES

Para probar la aplicación desarrollada creando las funciones requeridas por el nuevo algoritmo criptográfico 2NAES en el proceso de descifrado, se utilizan los datos de la Tabla 6-3.

Tabla 6-3 Datos para ejecución de la aplicación desarrollada para descifrado 2NAES

Archivo de entrada:	Cifrado.txt
Clave (128 bits):	}yl%OckF x){gI~o
Mensaje cifrado:	]□C.X□□`ò½□□□P/□o² ¬□□{ý³□r□á7□Ô+□ÍÍSnYÈG4□eniHd □□□vKpy"M□|rR□ÑÉ+dg□¹BÏÐ□®□ü4□ð)□>□3□v□□□[ýK2K □)»p`□Á°F"j;3□öÖÈá□ªÙÐòò□£@Vâê#□ÚÅÀ¦□ÃGbÒ_Ê v]`□U= 3^□ÌvÐ%□Si□ÃT□ÙÕè$□w□TÒFz□1ÿÄRò□ÞÁËèÂ#□Ò_ï¨ì□»|b_ □[□ç7ç@p□1<´«Â□□wÐ□□□□És□Iy{HO□4pÕC0□¦l□Ç¼X;_□ sp8□5ö□□□ßÁàCN{Å°ì□□ÐÂfýÑc¤ÌGþrpJt□□@Çô^Í□jìþMÈèv»□: @H0qã□ÿc¾□(□'□//ó[¶j□Ì¿□□èX{ÎU□¹□ÍÖyØ□çVëk□fz8\□û¹□ 5□oxrB□□

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el algoritmo criptográfico 2NAES en el proceso de descifrado, se obtiene el siguiente resultado:

Abrir el archivo de texto que contiene el mensaje original denominado “Cifrado.txt” que se muestra en la Figura 44-3.

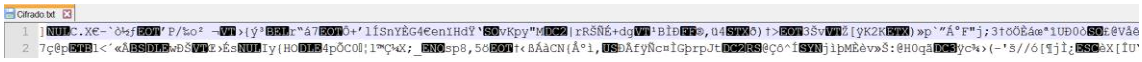


Figura 44-3 Captura del archivo de texto “Cifrado.txt”

Realizado por: Méndez Pablo, 2015

Se ejecuta el proceso de descifrado 2NAES con el mensaje cifrado para obtener el mensaje original, como se muestra en la Figura 45-3.

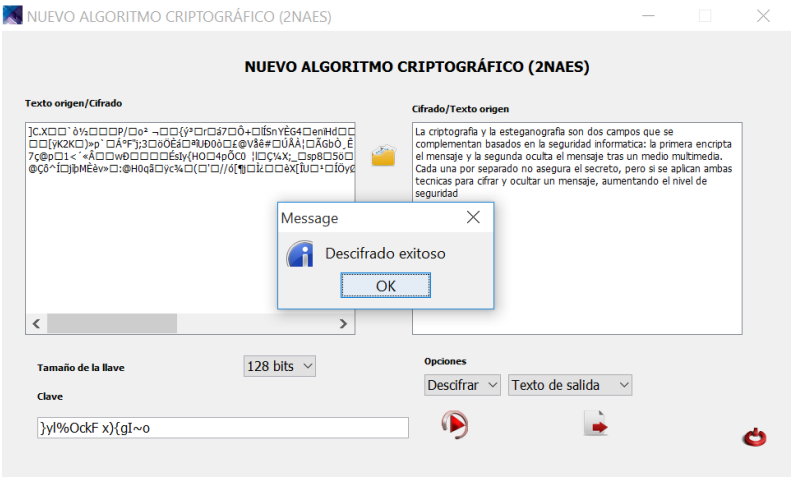


Figura 45-3 Captura del resultado de descifrado de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se exporta el mensaje descifrado a un archivo denominado “Descifrado.txt”, como se muestra en la Figura 46-3.

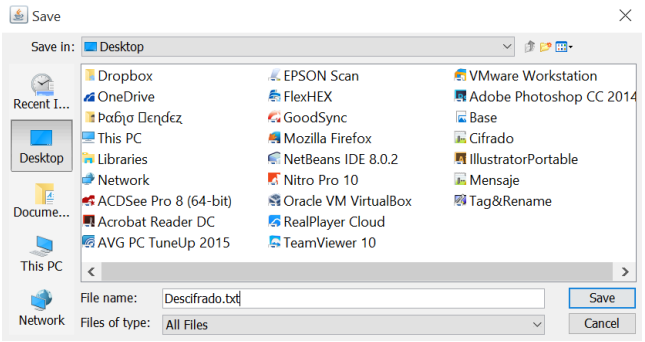


Figura 46-3 Captura de interfaz para exportación del mensaje descifrado
Realizado por: Méndez Pablo, 2015

Se obtiene el archivo con el mensaje descifrado, como se muestra en la Figura 47-3.

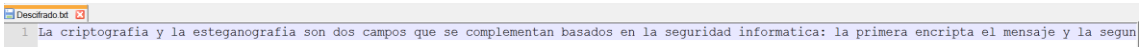


Figura 47-3 Captura del archivo exportado con el mensaje descifrado
Realizado por: Méndez Pablo, 2015

3.7.5. Análisis y diseño de algoritmos

A continuación, se realiza el análisis de complejidad de los algoritmos AES base y 2NAES:

Análisis de complejidad del algoritmo AES base

En la Figura 48-3 se muestra la complejidad del algoritmo AES base, con el valor de complejidad de cada proceso.

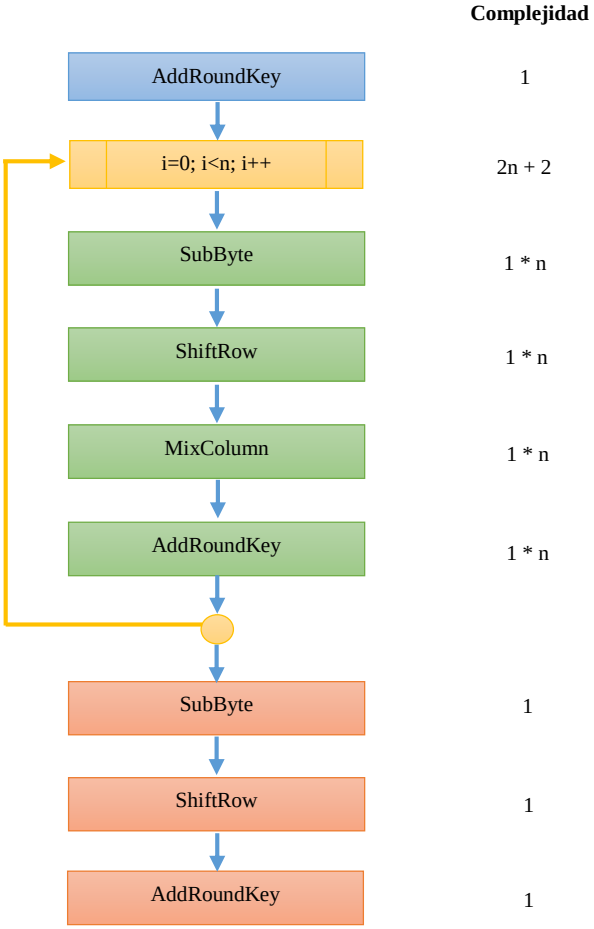


Figura 48-3 Complejidad del algoritmo AES base
Realizado por: Méndez Pablo, 2015

Cálculo de la complejidad del algoritmo AES base

El cálculo dela la complejidad del algoritmo AES base es el siguiente:

$$1 + (2n + 2) + (1 * n) + (1 * n) + (1 * n) + (1 * n) + 1 + 1 + 1$$
$$= 1 + 2n + 2 + n + n + n + n + 1 + 1 + 1$$
$$= 6n + 6$$

La complejidad del algoritmo AES base es 6n+6.

Análisis de complejidad del nuevo algoritmo criptográfico 2NAES

En la Figura 49-3 se muestra la complejidad del nuevo algoritmo criptográfico 2NAES, con el valor de complejidad de cada proceso.

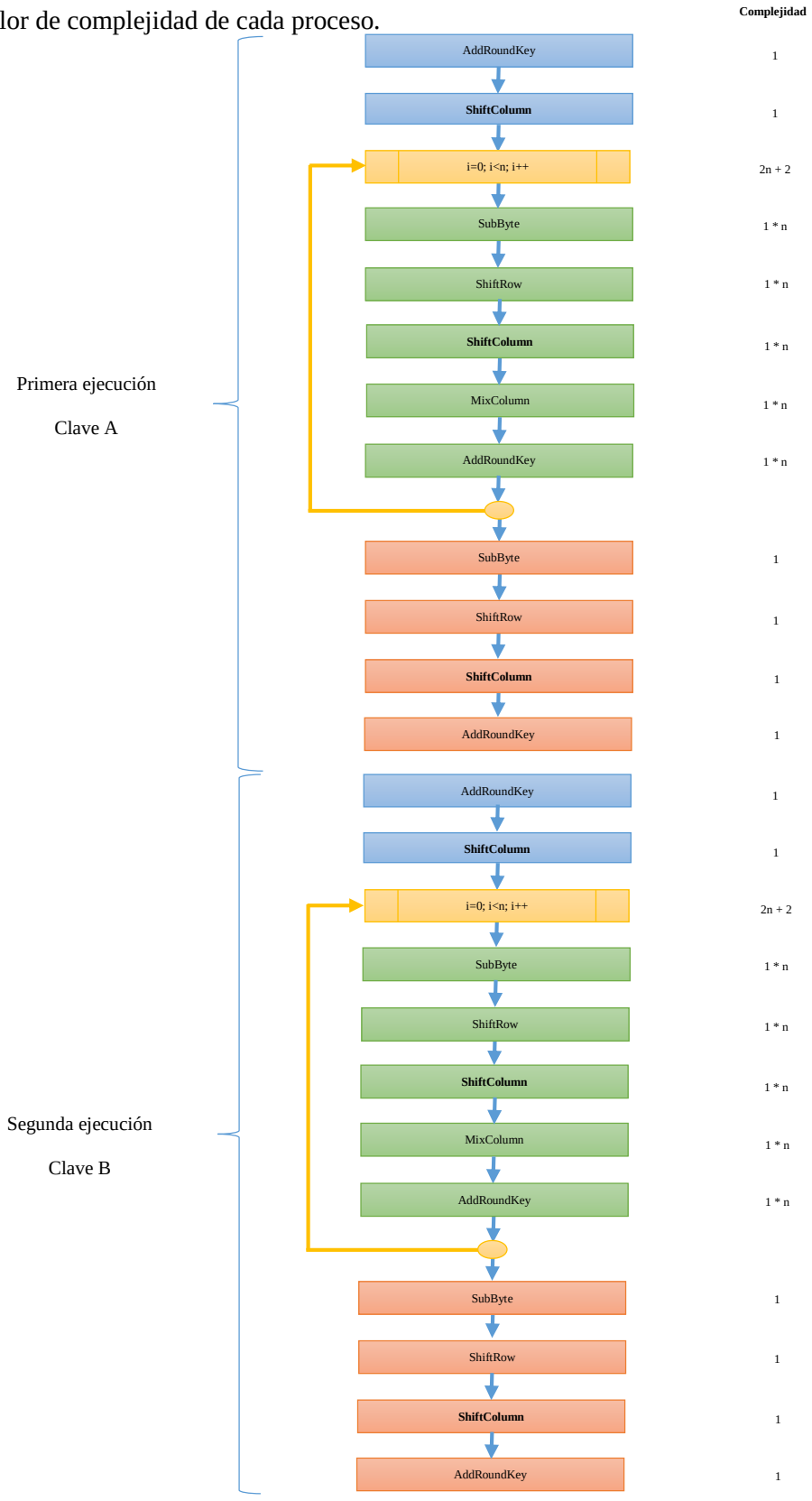


Figura 49-3 Complejidad del nuevo algoritmo 2NAES
Realizado por: Méndez Pablo, 2015

Cálculo de la complejidad del algoritmo AES base

El cálculo dela la complejidad del algoritmo AES base es el siguiente:

$$[2 + (2n + 2) + (1 * n) + (1 * n) + (1 * n) + (1 * n) + (1 * n) + 1 + 1 + 1 + 1] + [2 + (2n + 2) + (1 * n) + (1 * n) + (1 * n) + (1 * n) + (1 * n) + 1 + 1 + 1 + 1]$$

$$= [2 + 2n + 2 + n + n + n + n + n + 1 + 1 + 1 + 1] + [2 + 2n + 2 + n + n + n + n + n + 1 + 1 + 1 + 1]$$

$$= 14n + 16$$

La complejidad del nuevo algoritmo 2NAES es 14n+16.

Al calcular la complejidad del nuevo algoritmo criptográfico (2NAES) y del algoritmo criptográfico AES base se determina que tienen el mismo orden de complejidad lineal O(n).

3.8. Implementación del algoritmo esteganográfico en imágenes

3.8.1. Desarrollo de la aplicación

Luego de la descripción acerca del método esteganográfico LSB en el Capítulo II, se procede con el desarrollo de la aplicación, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

El la Figura 50-3 se muestran los paquetes de la aplicación

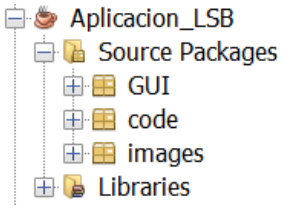


Figura 50-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** paquete de clases desarrolladas necesa0rias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 51-3 se muestran las clases de cada paquete.

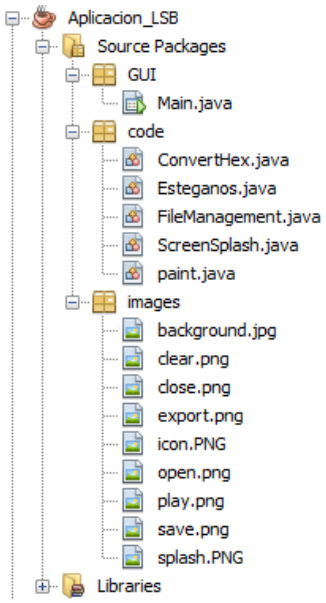


Figura 51-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **GUI** contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 52-3.

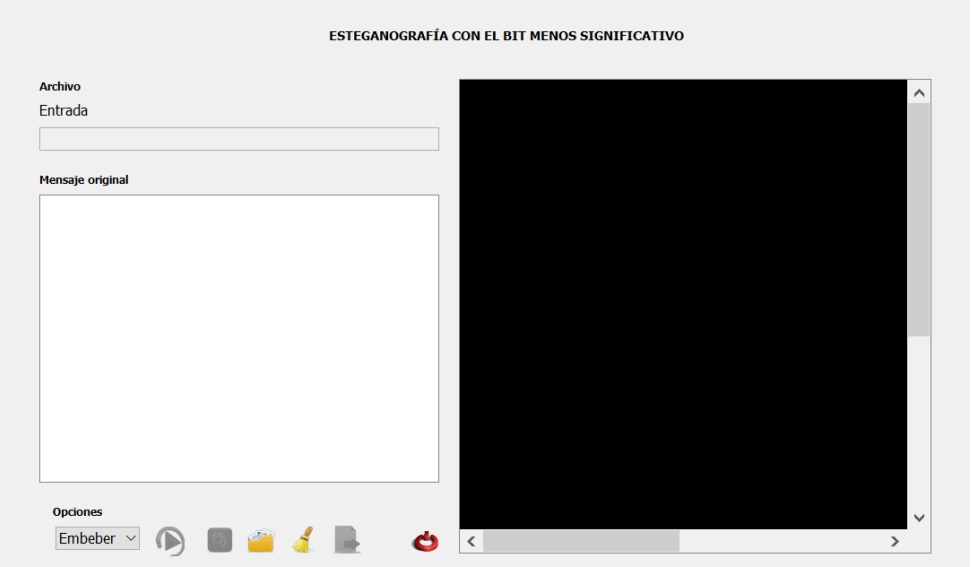


Figura 52-3 Interfaz de usuario de la aplicación Esteganografía LSB
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - ASCII a hexadecimal
  - Hexadecimal a byte

- **Esteganos.java (class):** clase que contiene atributos y métodos necesarios para los procesos de esteganografía, entre los principales se encuentran:

  - Armar mensaje
  - Embeber mensaje
  - Confirmar si existe embebido
  - Tamaño del mensaje
  - Extraer mensaje
  - Reemplazar LSB

- **FileManagement.java (class):** clase que contiene atributos y métodos para manejar archivos que utiliza la aplicación, entre las principales se encuentran**:**

  - Abrir archivo
  - Leer archivo

- Guardar archivo

- **Paint (class):** clase que contiene atributos y métodos para el manejo de las imágenes, entre las principales se encuentran:

- Abrir imagen
- Guardar imagen
- Ubicar imagen

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 53-3.

Figura 53-3 Splash Aplicación Esteganografía LSB
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizadas en el splash, el fondo como base del panel y los íconos para la aplicación.

En el Anexo A se detalla el código fuente principal para la implementación del algoritmo esteganográfico en imágenes.

A continuación, se detallan los procesos de embebido y de extracción del algoritmo esteganográfico:

3.8.2. Proceso de embebido

El proceso de embebido del algoritmo esteganográfico LSB en imágenes, se muestra en la Figura 54-3.

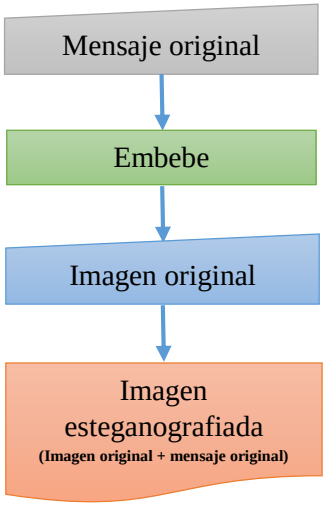


Figura 54-3 Proceso de embebido del algoritmo esteganográfico LSB en imágenes
Realizado por: Méndez Pablo, 2015

Ejecución

Para probar la aplicación desarrollada creando las funciones requeridas por el método LSB en el proceso de embebido, se utilizan los datos de la Tabla 7-3.

Tabla 7-3 Datos para ejecución de la aplicación desarrollada para embebido

Archivo de entrada:	Base.bmp
Archivo de salida:	Base_embebido.bmp
Mensaje cifrado:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el método LSB para el proceso de embebido, se obtiene el resultado mostrado en la Figura 55-3.

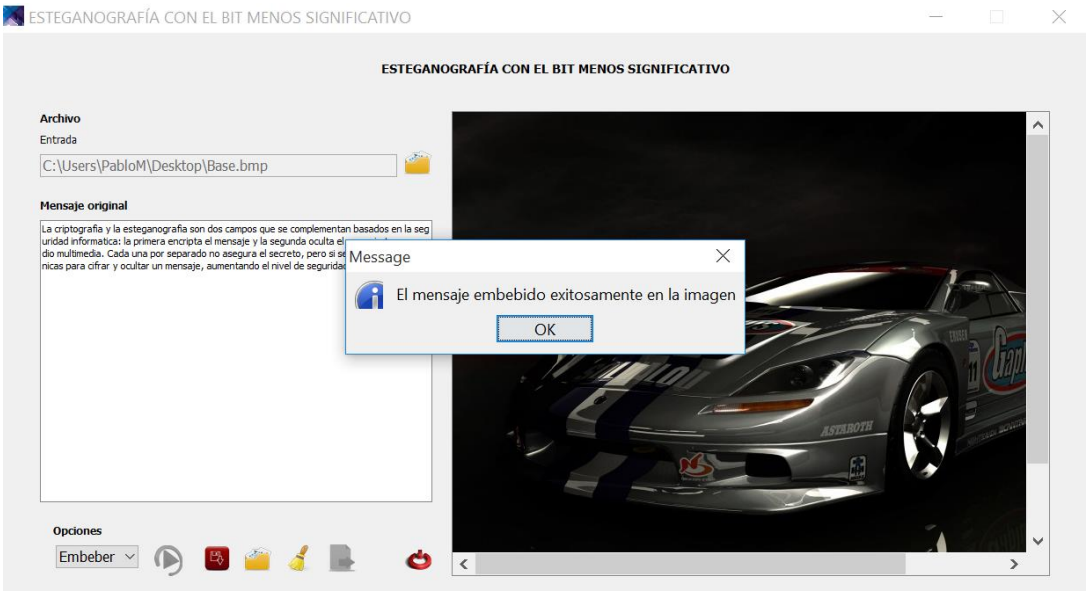


Figura 55-3 Captura del resultado de embebido de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Luego de guardar la imagen embebida, se muestra la Figura 56-3.

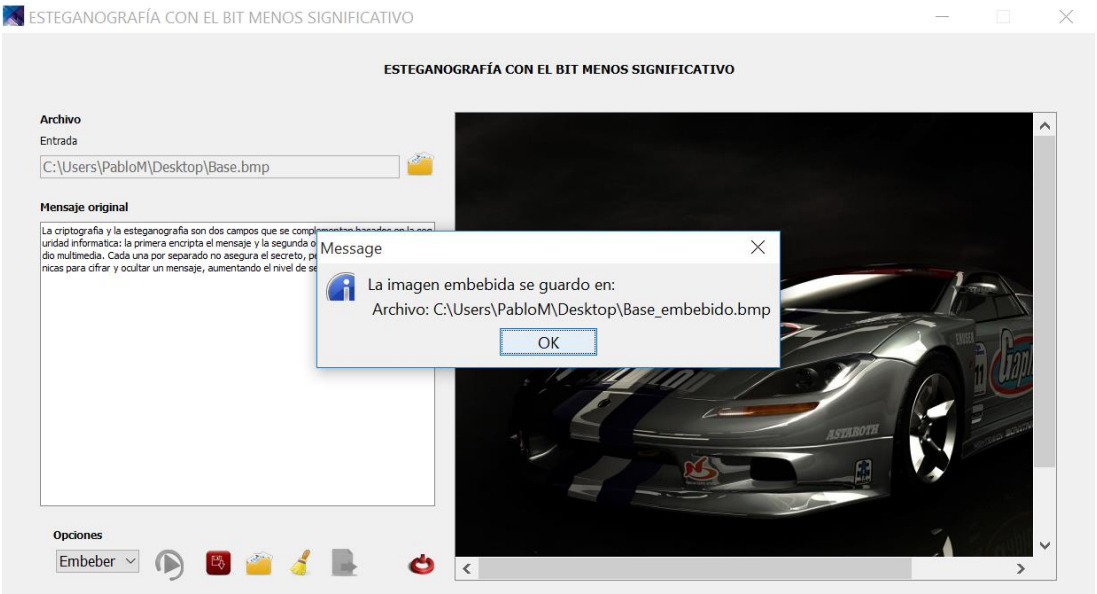


Figura 56-3 Captura del resultado de guardar la imagen embebida de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se comparan las imágenes "Base.bmp" y "Base_embebido.bmp", en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en la Figura 57-3 y Figura 58-3.

Base

Figura 57-3 Imagen "Base.bmp"
Realizado por: Méndez Pablo, 2015

Base_embebido.bmp

Figura 58-3 Imagen "Base_embebido.bmp"
Realizado por: Méndez Pablo, 2015

3.8.3. Proceso de extracción

El proceso de extracción del algoritmo esteganográfico LSB en imágenes se muestra en la Figura 59-3:

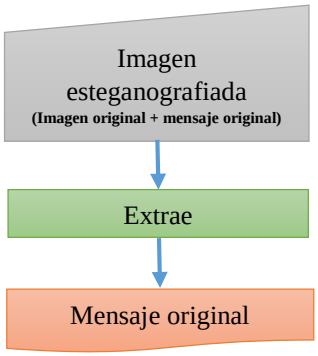


Figura 59-3 Proceso de extracción del algoritmo esteganográfico LSB en imágenes
Realizado por: Méndez Pablo, 2015

Ejecución

Para probar la aplicación desarrollada creando las funciones requeridas por el método LSB en el proceso de extracción, se utilizan los datos de la Tabla 8-3.

Tabla 8-3 Datos para ejecución de la aplicación desarrollada para extracción

Archivo de entrada:	Base_embebido.bmp
Archivo de salida:	Mensaje.txt

Realizado por: Méndez Pablo, 2015

Resultado

Luego de ejecutar la aplicación desarrollada creando las funciones requeridas por el método LSB para el proceso de extracción, se obtiene el resultado mostrado en la Figura 60-3.

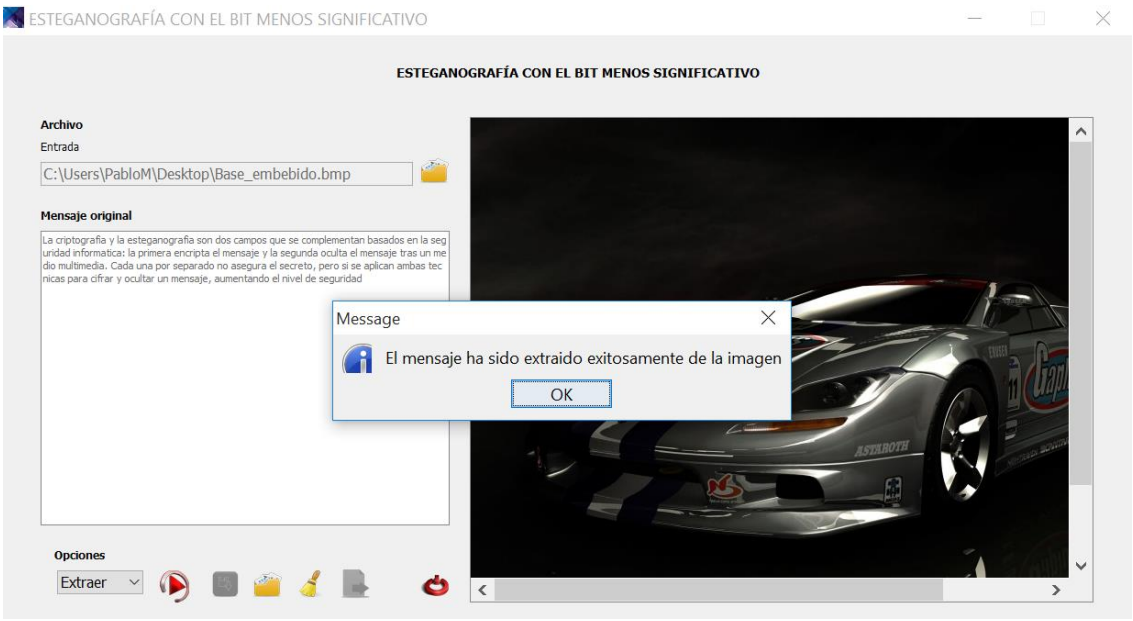


Figura 60-3 Captura del resultado de extracción de la aplicación desarrollada
Realizado por: Méndez Pablo, 2015

Se exporta el mensaje original que ha sido extraído de la imagen esteganografiada a un archivo denominado “Mensaje.txt”, como se muestra en la Figura 61-3.

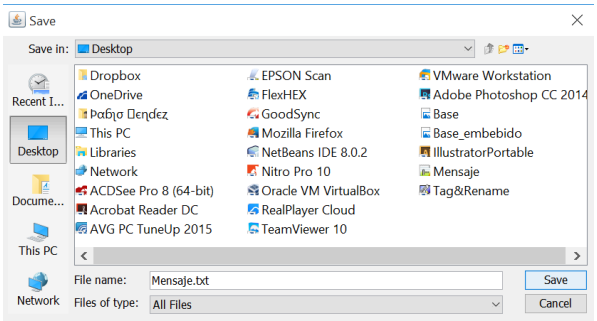


Figura 61-3 Captura de interfaz para exportación del mensaje extraído
Realizado por: Méndez Pablo, 2015

Se muestra el mensaje de confirmación de la extracción como se muestra en la Figura 62-3.

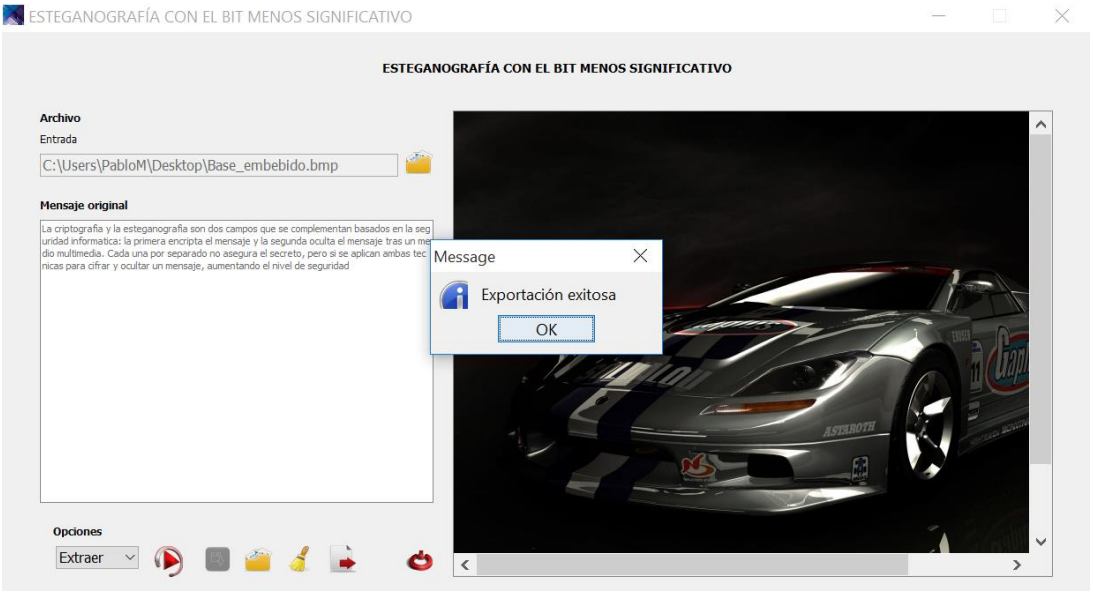


Figura 62-3 Confirmación de exportación del mensaje
Realizado por: Méndez Pablo, 2015

Se obtiene el archivo con el mensaje original que ha sido extraído de la imagen, como se muestra en la Figura 63-3.

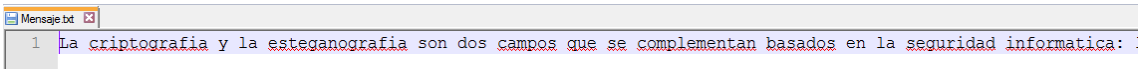


Figura 63-3 Captura del archivo extraído con el mensaje descifrado
Realizado por: Méndez Pablo, 2015

3.9. Integración de los algoritmos criptográficos y esteganográficos

Para realizar las pruebas se desarrollarán dos Prototipos:

Prototipo I

Para el Prototipo I se integra la aplicación con el algoritmo criptográfico que es considerado como base y la integración esteganográfica en imágenes.

Se implementa el algoritmo AES base con sus funciones:

- AddRoundKey
- SubByte
- ShiftRow
- MixColumn

Al algoritmo criptográfico AES base se incorpora esteganografía en imágenes con la técnica de LSB.

Prototipo II

Para el Prototipo II se integra la aplicación con el nuevo algoritmo criptográfico y la integración esteganográfica en imágenes.

Se implementa el nuevo algoritmo 2NAES con sus funciones:

- AddRoundKey
- SubByte
- ShiftRow
- **ShiftColumn (nueva función)**
- MixColumn

Además, el algoritmo 2NAES repite el proceso con una nueva clave generada a partir de la primera. Al algoritmo criptográfico 2NAES se incorpora esteganografía en imágenes con la técnica de LSB.

3.9.1. Prototipo I

3.9.1.1. Desarrollo de la aplicación

Se procede con el desarrollo de la aplicación del Prototipo I, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

El la Figura 64-3 se muestran los paquetes de la aplicación

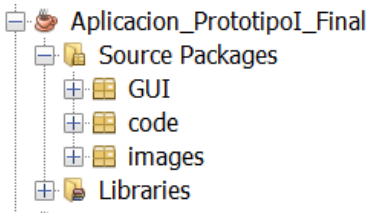


Figura 64-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** paquete de clases desarrolladas necesarias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 65-3 se muestran las clases de cada paquete.

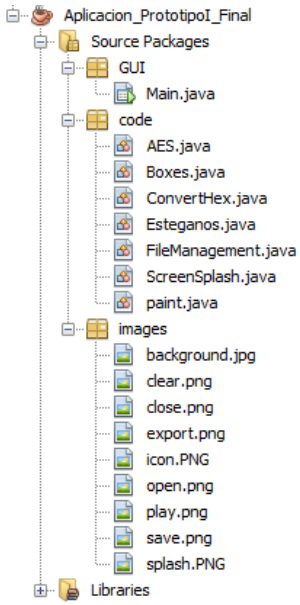


Figura 65-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete **GUI** contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 66-3.

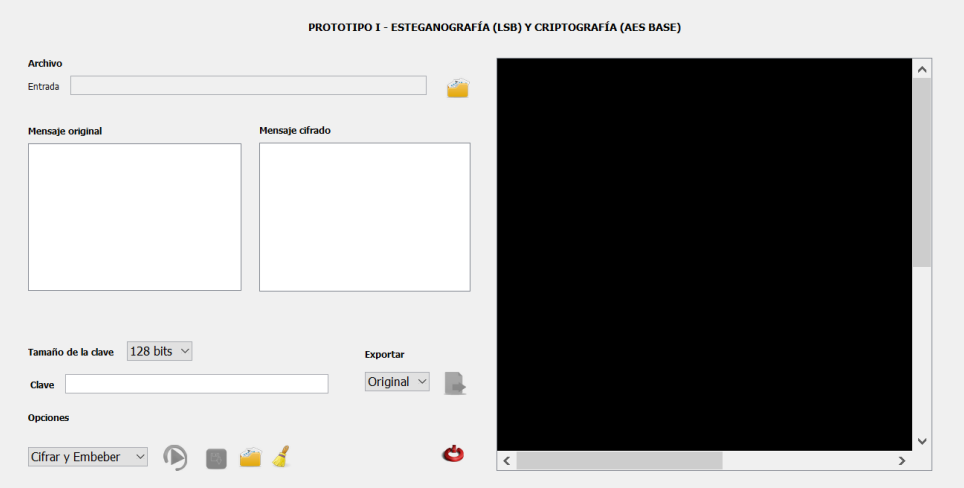


Figura 66-3 Interfaz de usuario de la aplicación Prototipo I
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **AES.java (class):** clase que contiene atributos y métodos del algoritmo AES, entre las principales se encuentran**:**

  - Expansión de clave
  - Subword
  - Rotword
  - Cifrado
  - AddRoundKey
  - SubByte
  - ShiftRow
  - MixColumn
  - Descifrado
  - InvShiftRow
  - InvSubByte
  - InvMixColumn
  - InvSubWord

- **Boxes.java (class):** clase que contiene métodos para obtener las Cajas que utiliza el algoritmo AES, entre las principales se encuentran**:**

  - S-Box
  - InvS-Box
  - Rcon

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - Hexadecimal a ASCII
  - ASCII a hexadecimal
  - Hexadecimal a byte
  - Byte a hexadecimal

- **Esteganos.java (class):** clase que contiene atributos y métodos necesarios para los procesos de esteganografía, entre los principales se encuentran:

  - Armar mensaje
  - Cifrar y embeber mensaje
  - Confirmar si existe embebido
  - Tamaño del mensaje
  - Extraer y descifrar mensaje
  - Reemplazar LSB

- **FileManagement.java (class):** clase que contiene atributos y métodos para manejar archivos que utiliza la aplicación, entre las principales se encuentran**:**

  - Abrir archivo
  - Leer archivo
  - Guardar archivo

- **Paint (class):** clase que contiene atributos y métodos para el manejo de las imágenes, entre las principales se encuentran:

  - Abrir imagen
  - Guardar imagen
  - Ubicar imagen

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 67-3.

Figura 67-3 Splash Aplicación Prototipo I LSB-AES
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizadas en el splash, el fondo como base del panel y los íconos para la aplicación.

En el Anexo A se detalla el código fuente principal para la integración del algoritmo criptográfico y esteganográfico para el Prototipo I.

A continuación, se detallan los procesos del algoritmo criptográfico y esteganográfico integrado en el Prototipo I de:

- Cifrado y embebido
- Extracción y descifrado

3.9.1.2. Cifrado y embebido

El proceso de algoritmo de cifrado (algoritmo AES base) y embebido (algoritmo esteganográfico LSB en imágenes) del Prototipo I se muestra en la Figura 68-3.

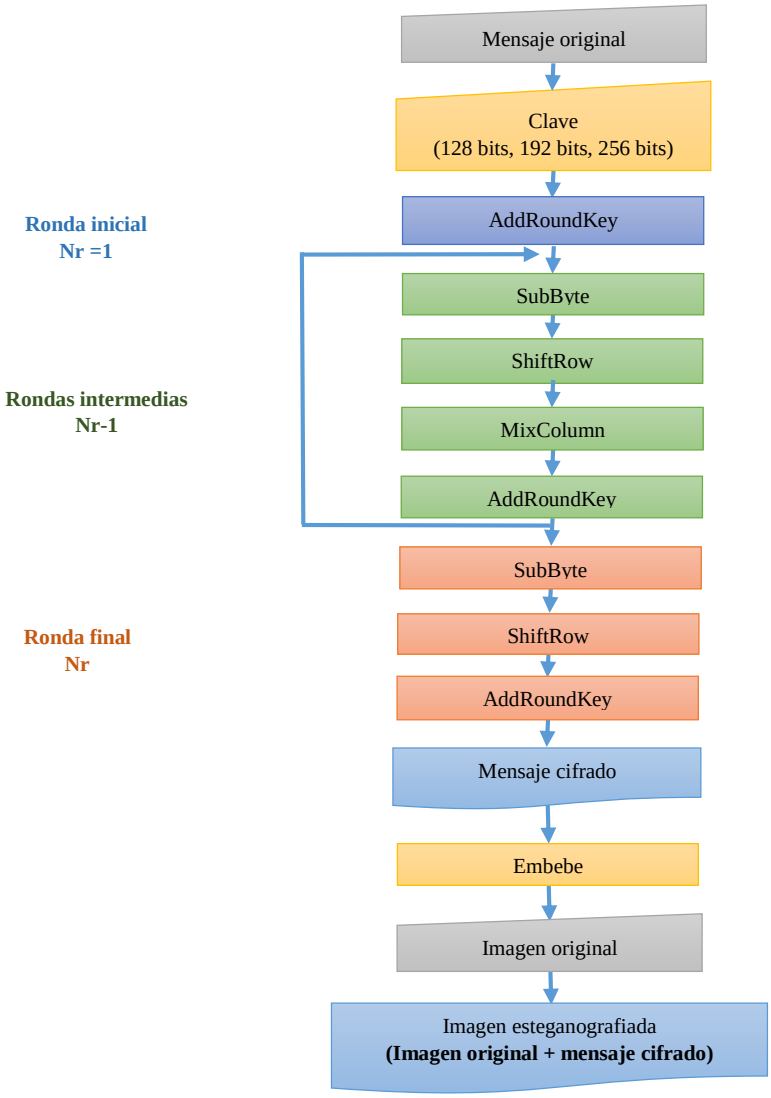


Figura 68-3 Proceso de cifrado y embebido del Prototipo I
Realizado por: Méndez Pablo, 2015

3.9.1.3. Extracción y descifrado

El proceso de algoritmo de extracción (algoritmo esteganográfico LSB en imágenes) y descifrado (algoritmo AES base) del Prototipo I se muestra en la Figura 69-3.

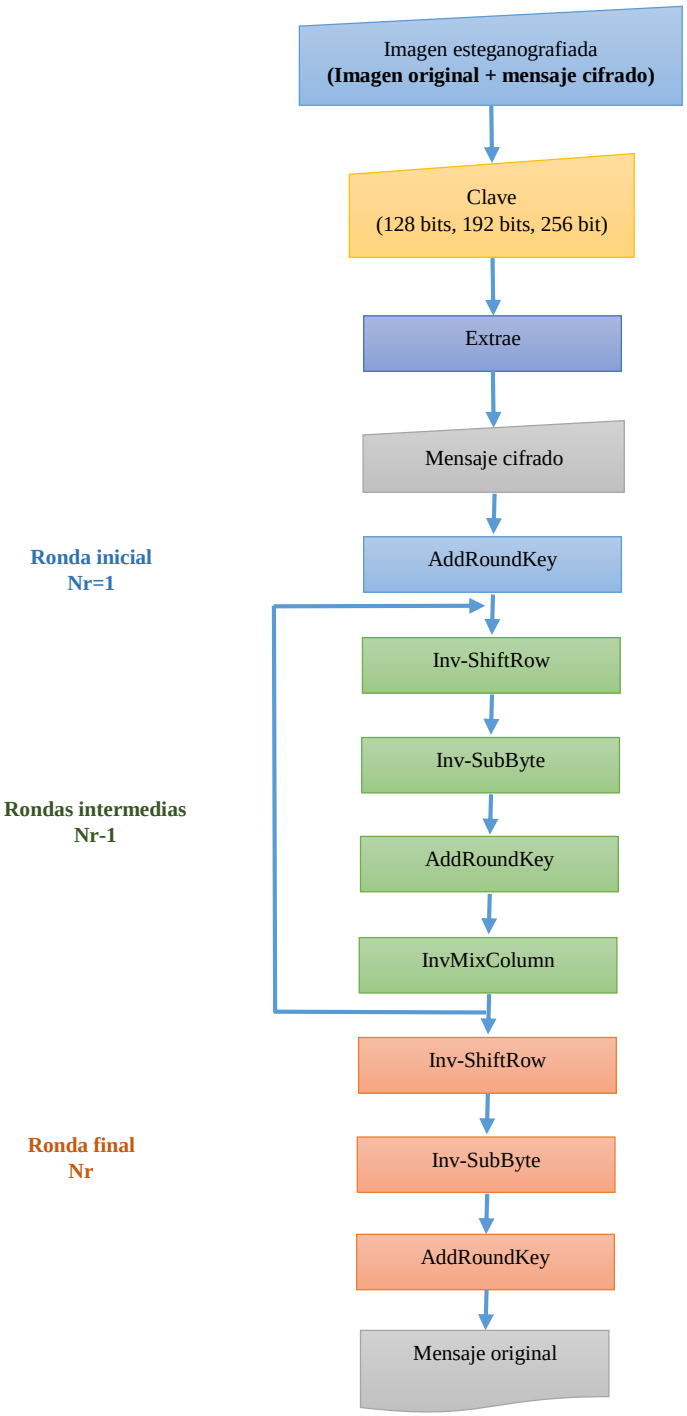


Figura 69-3 Proceso de extracción y descifrado del Prototipo I
Realizado por: Méndez Pablo, 2015

3.9.2. Prototipo II

3.9.2.1. Desarrollo de la aplicación

Se procede con el desarrollo de la aplicación del Prototipo II, utilizando el IDE de desarrollo Netbeans con el lenguaje de programación Java.

Paquetes

El la Figura 70-3 se muestran los paquetes de la aplicación.

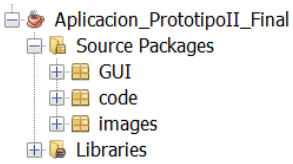


Figura 70-3 Paquetes de la aplicación
Realizado por: Méndez Pablo, 2015

La aplicación consta de 3 paquetes:

- **GUI:** paquete para interfaz de usuario
- **Code:** clases desarrolladas necesarias para la aplicación
- **Images:** paquete de imágenes utilizadas en la aplicación

Clases

Cada paquete posee clases requeridas para la aplicación, en la Figura 71-3 se muestran las clases de cada paquete.

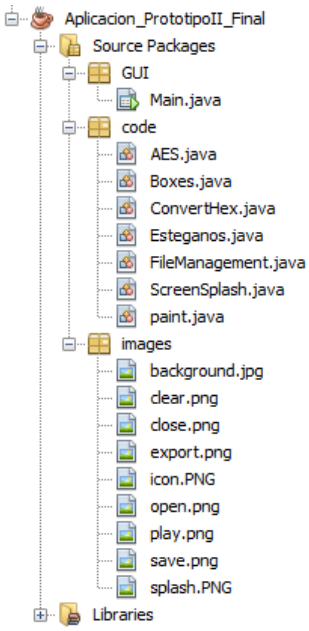


Figura 71-3 Clases de la aplicación
Realizado por: Méndez Pablo, 2015

El paquete GUI contiene:

- **Main.java (JFrame):** utilizado para el diseño de la interfaz de usuario, como se muestra en la Figura 72-3.

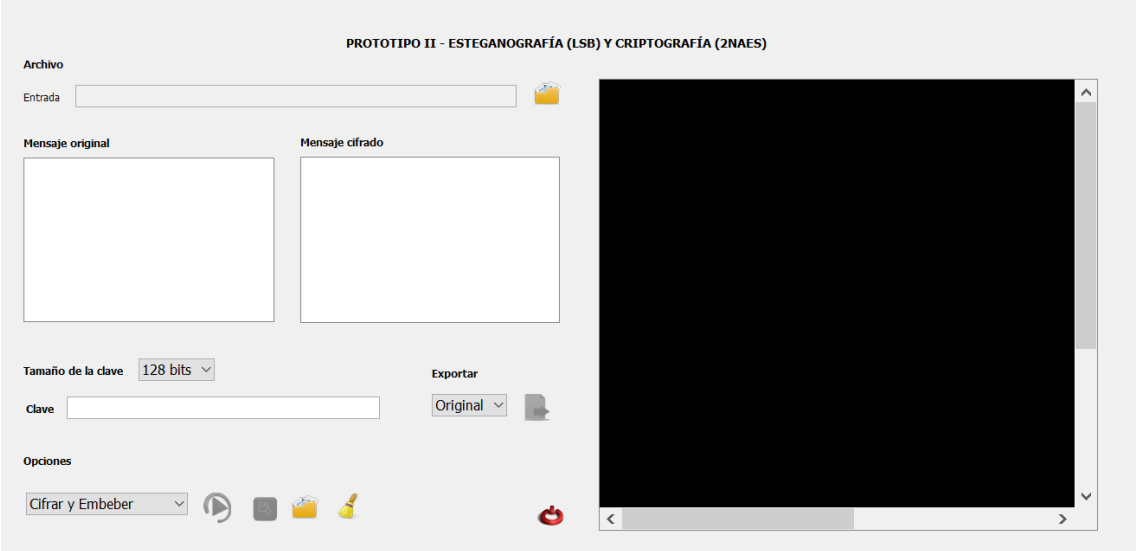


Figura 72-3 Interfaz de usuario de la aplicación Prototipo II
Realizado por: Méndez Pablo, 2015

El paquete **code** contiene:

- **AES.java (class):** clase que contiene atributos y métodos del algoritmo AES, entre las principales se encuentran**:**

- Expansión de clave
- Subword
- Rotword
- Cifrado
- AddRoundKey
- SubByte
- ShiftRow
- **ShiftColumn**
- MixColumn
- Descifrado
- InvShiftRow
- **InvShiftColumn**
- InvSubByte
- InvMixColumn
- InvSubWord
- **Generación de clave**

- **Boxes.java (class):** clase que contiene métodos para obtener las Cajas que utiliza el algoritmo AES, entre las principales se encuentran**:**

  - S-Box
  - InvS-Box
  - Rcon

- **ConvertHex.java (class):** clase que contiene métodos de conversión que utiliza la aplicación, entre las principales se encuentran**:**

  - Hexadecimal a ASCII
  - ASCII a hexadecimal
  - Hexadecimal a byte
  - Byte a hexadecimal

- **Esteganos.java (class):** clase que contiene atributos y métodos necesarios para los procesos de esteganografía, entre los principales se encuentran:

  - Armar mensaje
  - Cifrar y embeber mensaje

- Confirmar si existe embebido
- Tamaño del mensaje
- Extraer y descifrar mensaje
- Reemplazar LSB

- **FileManagement.java (class):** clase que contiene atributos y métodos para manejar archivos que utiliza la aplicación, entre las principales se encuentran**:**

- Abrir archivo
- Leer archivo
- Guardar archivo

- **Paint (class):** clase que contiene atributos y métodos para el manejo de las imágenes, entre las principales se encuentran:

- Abrir imagen
- Guardar imagen
- Ubicar imagen

- **ScreenSplash.java (class):** clase que contiene atributos y métodos para mostrar la pantalla del splash para la aplicación, la cual se muestra en la Figura 73-3.

Figura 73-3 Splash Aplicación Prototipo II LSB-2NAES
Realizado por: Méndez Pablo, 2015

El paquete **images** contiene las imágenes utilizadas en el splash, el fondo como base del panel y los íconos para la aplicación.

En el Anexo A se detalla el código fuente principal para la integración del algoritmo criptográfico y esteganográfico para el Prototipo II.

A continuación, se detallan los procesos del algoritmo criptográfico y esteganográfico integrado en el Prototipo II de:

- Cifrado y embebido
- Extracción y descifrado

3.9.2.2. Cifrado y embebido

El proceso de algoritmo de cifrado (algoritmo 2NAES) y embebido (LSB en imágenes) del Prototipo II se muestra en la Figura 74-3.

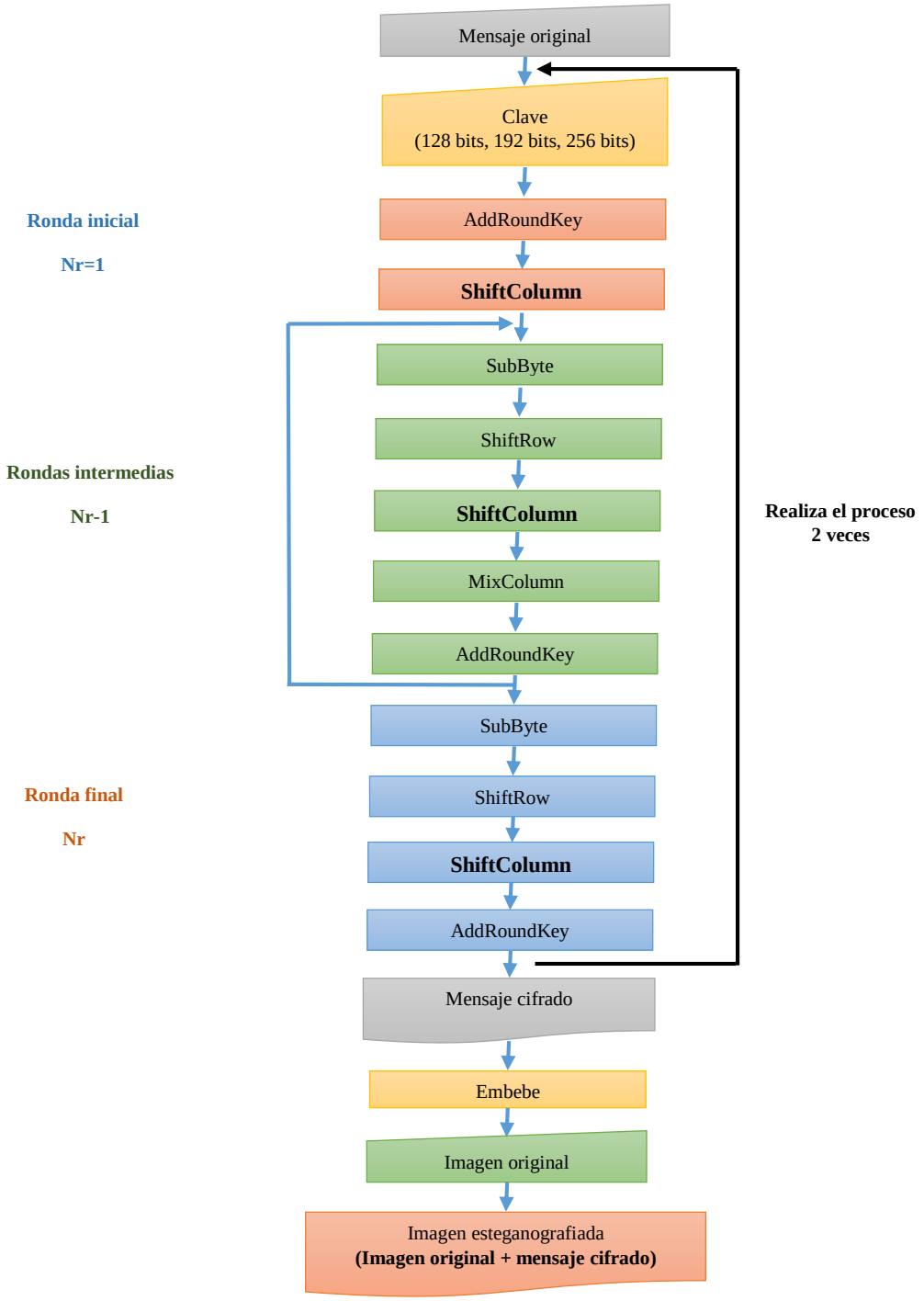


Figura 74-3 Proceso de cifrado y embebido del Prototipo II
Realizado por: Méndez Pablo, 2015

3.9.2.3. Extracción y descifrado

El proceso de algoritmo de extracción (LSB en imágenes) y descifrado (2NAES) del Prototipo II se muestra en la Figura 75-3.

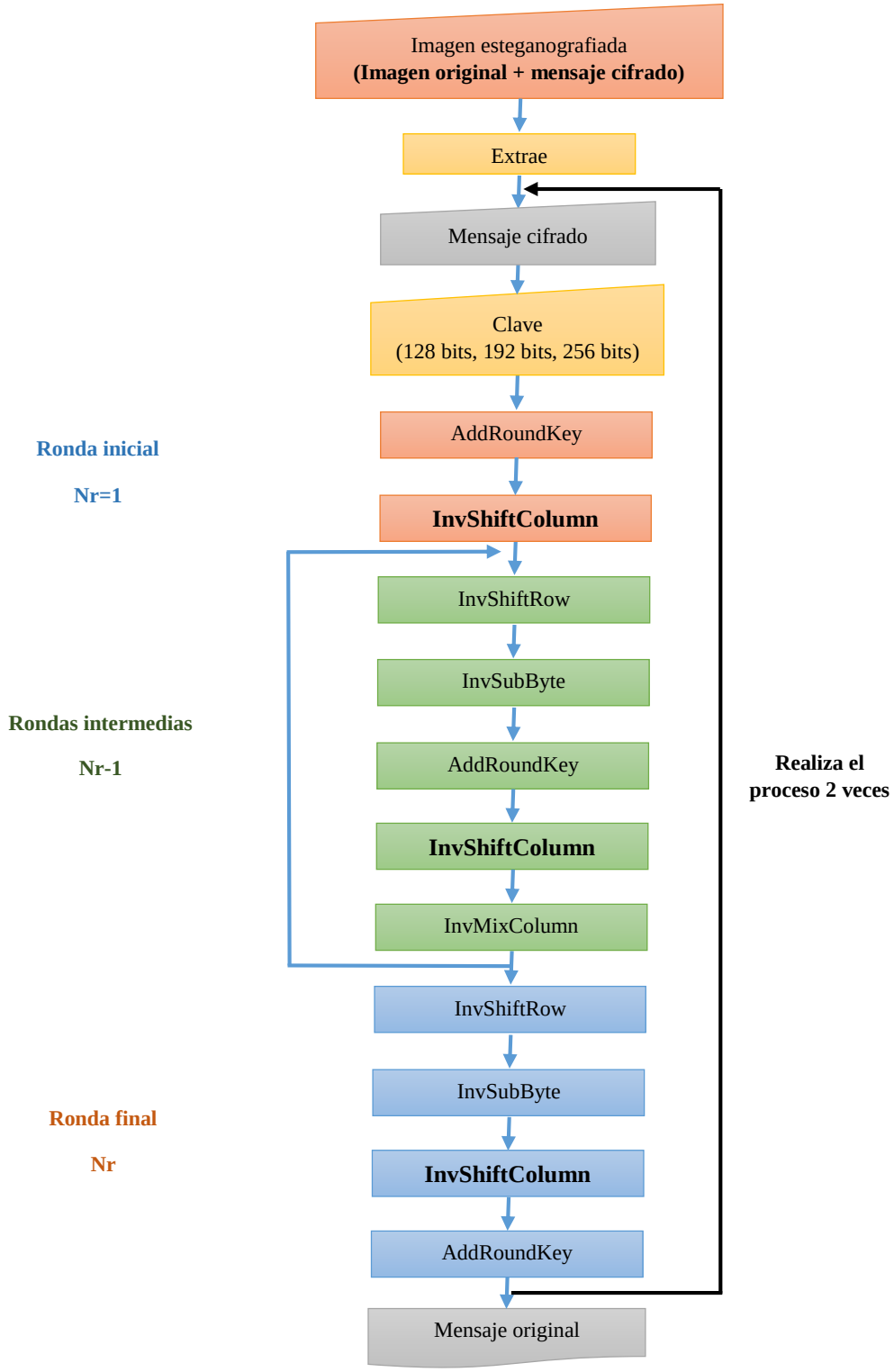


Figura 75-3 Proceso de extracción y descifrado del Prototipo II
Realizado por: Méndez Pablo, 2015

3.10. Definición de los escenarios de pruebas

Ambiente de pruebas

Se establece un ambiente de pruebas común en el que comparten los escenarios para el cifrado/embebido y extracción/descifrado. Las condiciones del ambiente de pruebas para los 2 escenarios son:

- Utilización de tamaños de clave de 128 bits, 192 bits, 256 bits
- Tamaños de bloque de 128 bits
- Utilización de imágenes BMP

Escenarios

En el ambiente de pruebas se definen dos escenarios:

- **Escenario 1**

En el primer escenario se utilizará el Prototipo I que incluye el algoritmo criptográfico AES que ha sido tomado como algoritmo base, con la incorporación de la esteganografía en imágenes utilizando el método LSB.

- **Escenario 2**

En el segundo escenario se utilizará el Prototipo II que incluye el nuevo algoritmo criptográfico desarrollado 2NAES, con la incorporación de la esteganografía en imágenes utilizando el método LSB.

Resultados

Posteriormente se realizan las pruebas en los dos escenarios planteados con la finalidad de demostrar la mejora de la seguridad del nuevo algoritmo implementado (2NAES), para lo cual se compararán los resultados obtenidos entre:

- Prototipo I que incluye el algoritmo AES existente y la incorporación de la esteganografía (LSB)

- Prototipo II que incluye el nuevo algoritmo criptográfico (2NAES) implementado y la incorporación de la esteganografía (LSB)

3.11. Hipótesis

3.11.1. Determinación de variables

De acuerdo a la hipótesis planteada, luego de se determinan las siguientes variables:

- Algoritmo criptográfico con la incorporación de la esteganografía en imágenes
- Seguridad

3.11.2. Operacionalización conceptual

La Tabla 9-3, muestra la operacionalización conceptual de las variables determinadas.

Tabla 9-3 Operacionalización conceptual

VARIABLE	TIPO	CONCEPTO
Algoritmo criptográfico con la incorporación de la esteganografía en imágenes	Independiente	Algoritmo criptográfico para cifrar la información.
Seguridad	Dependiente	Nivel de protección de la información utilizando la criptografía y esteganografía

Realizado por: Méndez Pablo, 2015

3.11.3. Operacionalización metodológica

La Tabla 10-3, muestra la operacionalización metodológica de las variables determinadas.

Tabla 10-3 Operacionalización metodológica

VARIABLE	INDICADOR	TÉCNICA	INSTRUMENTO/ FUENTE
Algoritmo criptográfico con la incorporación de la esteganografía en imágenes	• Complejidad • Líneas de código • Recursos utilizados	• Búsqueda de información • Pruebas • Observación	• Netbeans
Seguridad	• No. claves utilizadas • No. rondas • No. funciones usadas por el algoritmo • No. funciones ejecutadas por algoritmo • Entropía • Histograma • N-grama • Autocorrelación • Análisis de fuerza bruta	• Pruebas • Observación • Análisis	• FlexHEX • Guiffy Image Diff • Cryptool

Realizado por: Méndez Pablo, 2015

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

En este Capítulo, se desarrollan las pruebas en los escenarios establecidos, se analizan, comparan los resultados obtenidos y se comprueba la hipótesis planteada.

4.1. Desarrollo de las pruebas

4.1.1. Prototipo I

El Prototipo I que se ha desarrollado utiliza el algoritmo criptográfico AES base con la incorporación del método esteganográfico LSB (Least Significant Bit).

4.1.1.1. Cifrado y embebido

4.1.1.1.1. Clave de 128 bits

Para probar el proceso de cifrado y embebido con el Prototipo I desarrollado con clave de 128 bits, se utilizan los datos de la Tabla 1-4.

Tabla 1-4 Datos para ejecución de cifrado y embebido con el Prototipo I con clave de 128 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Imagen de entrada:	
Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles

Clave (128 bits):	}yl%OckF x){gI~o
Mensaje:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo I con clave de 128 bits, se obtiene el resultado mostrado en Tabla 2-4 y la Figura 1-4.

Tabla 2-4 Mensaje cifrado y embebido con el Prototipo I con clave de 128 bits

Mensaje cifrado:	X□0□{Vq□□□ß□@□}ý□□É;¬ô«□□□S·aØØÉ7×ª□X□4□W□é L3¶H□□□GuJ□jÙ)CÎÎS□\§R□ñb□- □ÚH èZÍ□YàÔÞ□1Ì@□iÀëÞ□`¶æ□F□□in□ □»c'□ùÿó9□D□£□□[©5 □□□«□ □□çTI□fî²□Ü□¬>□O □□£ø-½□□7Æ□óÔM□□£°`□Æ□ □§p»¯õõËuïÏ□□Âã□}ðóñ 9Ì□/□déV°□ëÐ□öÊÑ□r àåÑ□"Y □□Oåý¥ÉÙÚ□□x□É□4ê ÄÊÀ`□y□±½{ES□S9 □WÁÁñ-□□□ÔÆ4|n□vÆE□□|ÌS»ùþÓ' °v3□-¯<G□ö□±J□□ê¸E□¨R□2□Ë□BN!Ô□Ò□ä&□□Û®"õ ØÊ5□O□8□V□Yv□@Éx00±#§Möë□□Ïç□=×YMå

Realizado por: Méndez Pablo, 2015

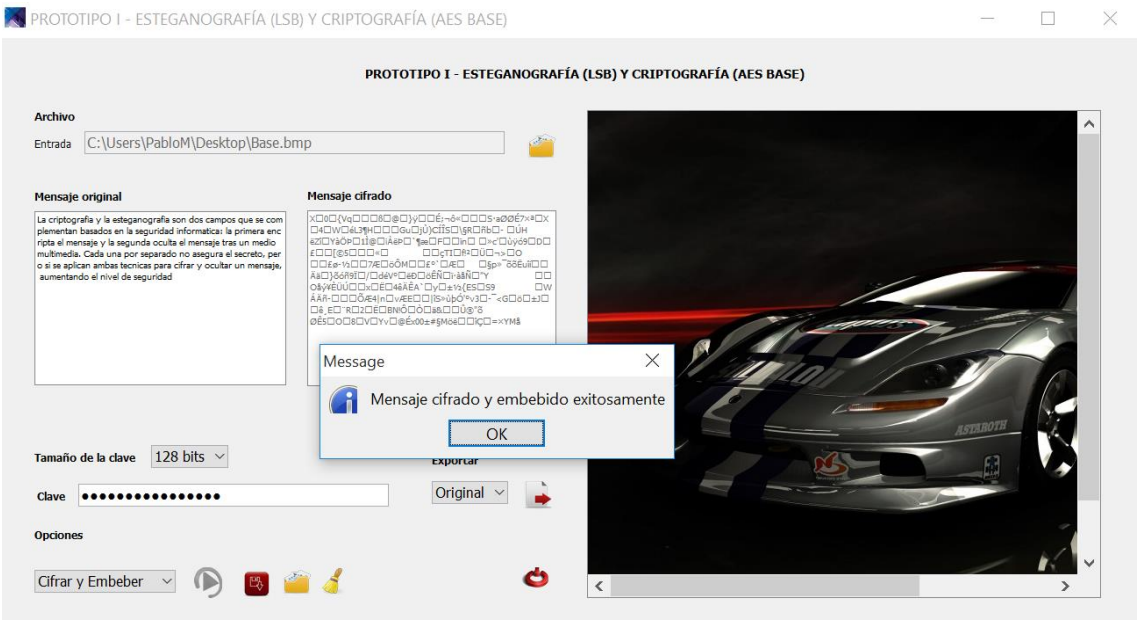


Figura 1-4 Captura del resultado de cifrado y embebido del Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.1.1.1.2. Clave de 192 bits

Para probar el proceso de cifrado y embebido con el Prototipo I desarrollado con clave de 192 bits, se utilizan los datos de la Tabla 3-4.

Tabla 3-4 Datos para ejecución del Prototipo I con clave de 192 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Imagen de entrada:	
Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Clave (192 bits):	<E?<E?<E?<E?<E?<E?<E?<E?
Mensaje:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo I con clave de 192 bits, se obtiene el resultado mostrado en Tabla 4-4 y la Figura 2-4.

Tabla 4-4 Mensaje cifrado y embebido con el Prototipo I con clave de 192 bits

Mensaje cifrado:	¼£ZnÌE®*□R÷{9aý,Ì&üØ¼Ø<c$□ÐQ□»□i□-²ï)å@□T□□ùÈQìå□ »□Û,k(<□□(□Ì-Ô×Ï6á□□ {□\□[□±¸ÌÐ¶9Ú□¿5□¨□AÙ=¸1VfÑî× □³ñ□ä□k¤É□È¨A§â□µ-□□Ê□NE□Û=szE□³JG É^6Xµ-+û□□□□Ô«B\□□I~□´5□□Ñmr□/□□©□ñ□□□g4□§å êª(□/ □□ ıÍ«j:□Ê¤ı+ç□zï□«¨Óê□®%-W'□u½Qþ□□µwÞ□öê=□@ı Ø¦Úó□D/V□□¥ô□´õø□Úï□□ma)ïÇq¨ □□B¶□¶#¢□NÔxXsÎY□□ß.å]»èm@Õõº□□Lâ±½ê□□-TúÎÝ□K□C □ÿhòQw÷<M@u\□□Íý□ZØä®\|d□□□□¹~□□$□3□Tòµjt'©À

Realizado por: Méndez Pablo, 2015

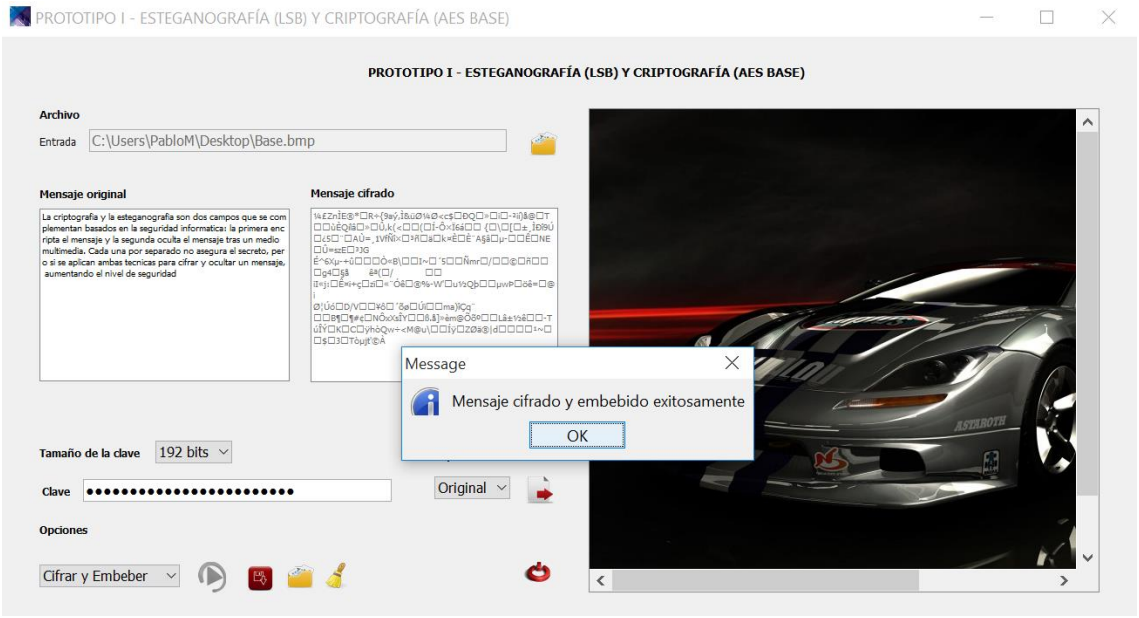


Figura 2-4 Captura del resultado de cifrado y embebido del Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

4.1.1.1.3. Clave de 256 bits

Para probar el proceso de cifrado y embebido con el Prototipo I desarrollado con clave de 256 bits, se utilizan los datos de la Tabla 5-4.

Tabla 5-4 Datos para ejecución del Prototipo I con clave de 256 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Imagen de entrada:	
Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Clave (256 bits):	<E?<E?<E?<E?<E?<E?<E?<E?<E?<E?<E

Mensaje:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo I con clave de 256 bits, se obtiene el resultado mostrado en Tabla 6-4 y la Figura 3-4.

Tabla 6-4 Mensaje cifrado y embebido con el Prototipo I con clave de 256 bits

Mensaje cifrado:	q/*ï7°c□^CQÔYe□±¿`îG[□g□ßI8è¹G7K¾3+®□\□>A1□Bcg^ëÖY AmO□□>□□Ü□j,□□`CT6üJ□w¨>>CÙtª5□Nê'□□□i%Â³,□Z□I 1Aqàâ□®×-µ¸□□*[ì□Ícw□□Ý¸iå£ÎÎX□□T□□¬\¦ -÷#ªÚÉÚ2ë□□ êÍº¡G□□U²¸þ9¢□□ò0¿ÿá□8éÑðŸT÷□p'%X□@□□±□□±¿·ßöaòG Þ±□W□□ □'ß□□?n7¸□□□ÁZR□□Ý□¶□je□□s□ }Üi±u□Çû□céqíü□□õê□□}□□Þ´eÓ□tÆ□□Xåò□óË¨£åBäÔßé□ Z49paEÉ¦¦ðAñüÑÒß·ÛÂ 0£(□#m□¹-□□□_□Q 8úÎby»Sò□Ô□□□ □]□□Aá³¨□P□□Î□¥Ü,®□□□õª□Ùà

Realizado por: Méndez Pablo, 2015

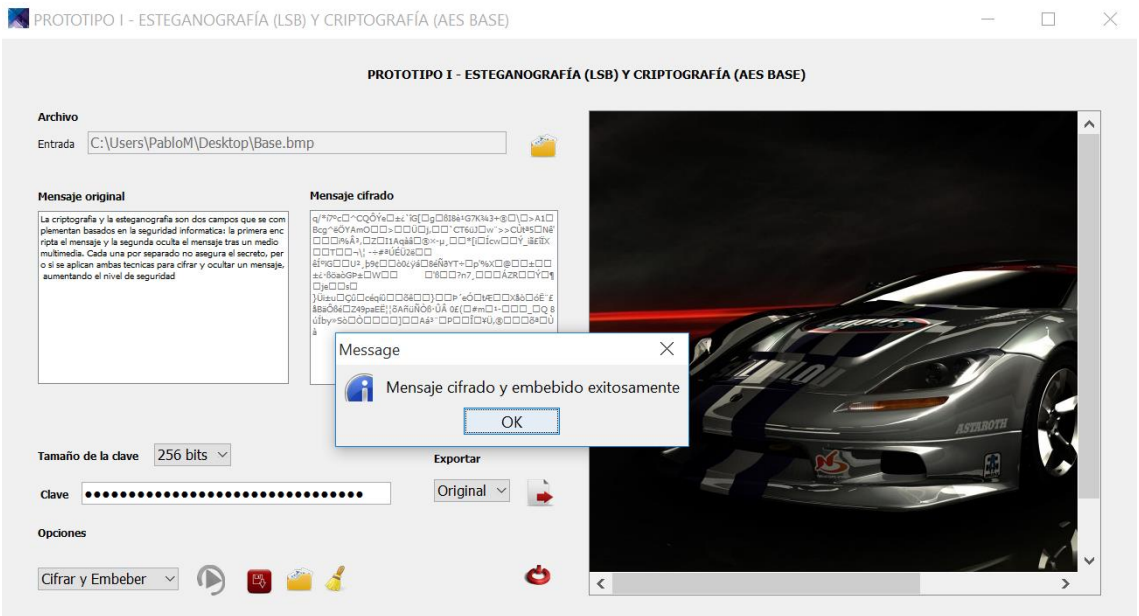


Figura 3-4 Captura del resultado de cifrado y embebido del Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

4.1.1.2. Extracción y descifrado

4.1.1.2.1. Clave de 128 bits

Para probar el proceso de extracción y descifrado con el Prototipo I desarrollado con clave de 128 bits, se utilizan los datos de la Tabla 7-4.

Tabla 7-4 Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 128 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles
Imagen de entrada:	
Clave (128 bits):	}yl%OckF x){gI~o

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo I con clave de 128 bits, se obtiene el resultado mostrado en la Tabla 8-4 y la Figura 4-4.

Tabla 8-4 Mensaje extraído y descifrado con el Prototipo I con clave de 128 bits

Mensaje cifrado:	X□0□{Vq□□□ß□@□}ÿ□□É;¬ô«□□□S·aØØÉ7×ª□X□4□W□é L3¶H□□□Gu□jÙ)CÎÎS□\§R□ñb□- □ÚH ëZÏ□YàÔÞ□1Î@□ÌàëÞ□`¶æ□F□□in□ □»c'□ùÿó9□D□£□□[©5 □□□«□ □□çT]□fl²□Ü□¬>□O □□£ø-½□□7Æ□öÔM□□£°`□Æ□ □§p»¯õõËuï□□Äã□}õóñ 9Ï□/□déV°□ëÐ□öÊÑ□ràåÑ□"Y □□Oåý¥ÈÚÚ□□x□É□4ê ÄÊÀ` □y□±½{ES□S9 □WÁÄñ-□□□ÔÆ4|n□vÆE□□|ïS»ùþÓ' °v3□-¯<G□ö□±J□□ê¸E□¨R□2□Ë□BN!Ô□Ò□ä&□□Û®"õ ØÊ5□O□8□V□Yv□@Éx00±#§Möë□□Ïç□=×YMå
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el m ensaje tras un medio multimedia. Cada una por separa do no asegura el secreto, pero si se aplican ambas tecn icas para cifrar y ocultar un mensaje, aumentando el niv el de seguridad

Realizado por: Méndez Pablo, 2015

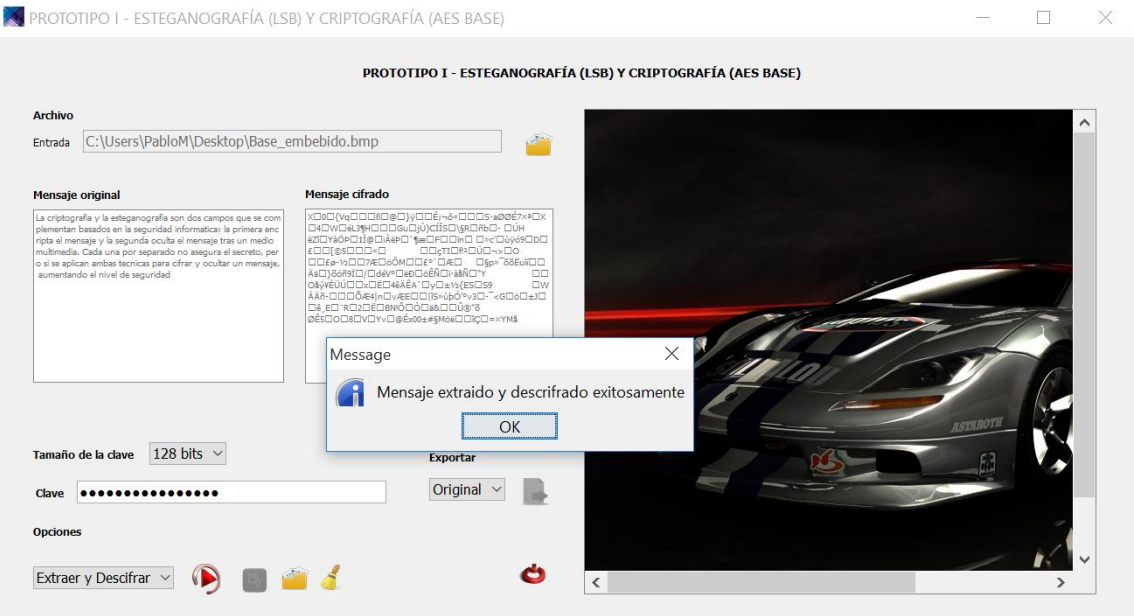


Figura 4-4 Captura del resultado de descifrado y extracción del Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.1.1.2.2. Clave de 192 bits

Para probar el proceso de extracción y descifrado con el Prototipo I desarrollado con clave de 192 bits, se utilizan los datos de la Tabla 9-4.

Tabla 9-4 Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 192 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600pixeles
Imagen de entrada:	
Clave (192 bits):	<E?<E?<E?<E?<E?<E?<E?<E?

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo I con clave de 192 bits, se obtiene el resultado mostrado en la Tabla 10-4 y la Figura 5-4.

Tabla 10-4 Mensaje extraído y descifrado con el Prototipo I con clave de 192 bits

Mensaje cifrado:	¼£ZnÌE®*□R÷{9aý,Ì&üØ¼Ø<c$□ÐQ□»□i□-²i)å@□T□□ùÈQÍã□ »□Û,k(<□□(□Í-Ô×Ìóá□□ {□\□[□±¸ÌÐ9Ú□¿5□¨□AÙ=¸1VfÑÍ× □³ñ□ä□k¤È□È¨A§â□µ-□□Ê□NE□Û=szE□³JG É^6Xµ-+û□□□□Ò«B\□□I~□´5□□Ñmr□/□□©□ñ□□□g4□§å êª(□/ □□ ïÍ«j:□Ê¤i+ç□zï□«¨Ôê□®%-W'□u½Qþ□□µwÞ□öê=□@i Ø¦Úó□D/V□□¥ô□´õø□Úí□□ma)ÏÇq¨ □□B¶□¶#¢□NÔxXsÎY□□ß.å]»èm@Õõº□□Lâ±½ê□□-TúÎÝ□K□C □ÿhòQw÷<M@u\□□Íý□ZØã®\|d□□□□¹~□□$□3□Tòµjt'©À
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

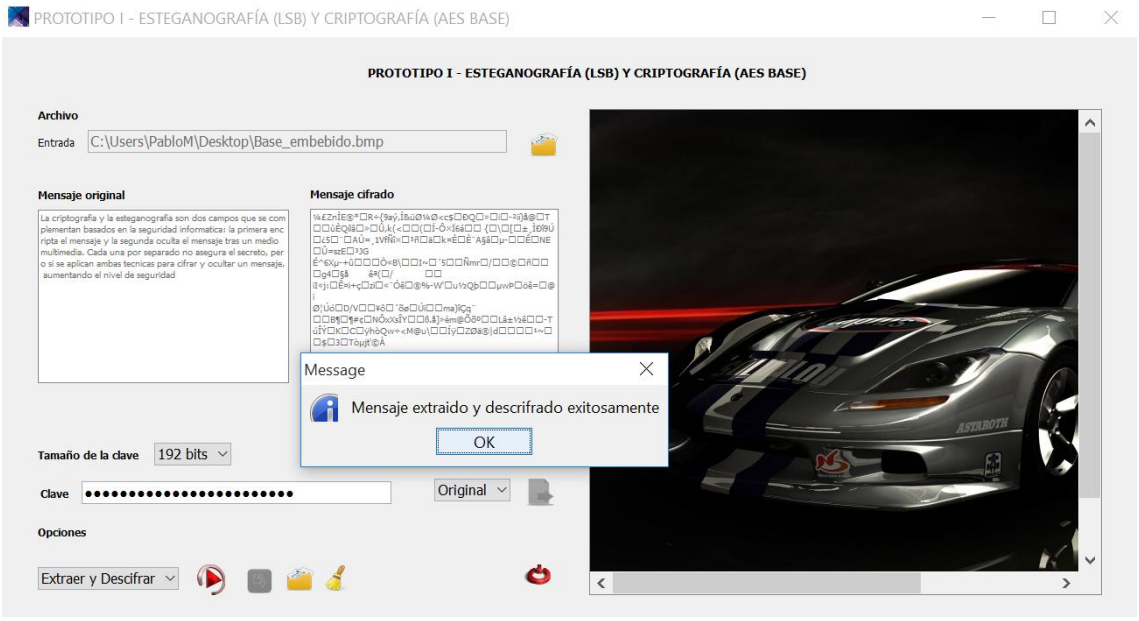


Figura 5-4 Captura del resultado de descifrado y extracción del Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.1.1.2.3. Clave de 256 bits

Para probar el proceso de extracción y descifrado con el Prototipo I desarrollado con clave de 256 bits, se utilizan los datos de la Tabla 11-4.

Tabla 11-4 Datos para ejecución de descifrado y extracción con el Prototipo I con clave de 256 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de salida:	800x600 pixeles

Imagen de entrada:	
Clave (256 bits):	<E?<E?<E?<E?<E?<E?<E?<E?<E?<E?<E

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo I con clave de 256 bits, se obtiene el resultado mostrado en la Tabla 12-4 y la Figura 6-4.

Tabla 12-4 Mensaje extraído y descifrado con el Prototipo I con clave de 256 bits

Mensaje cifrado:	q/*ï7°c□^CQÔÝe□±¿`ÏG[□g□ßÏ8è¹G7K¾3+®□\□>A1□Bcg^ëÖY AmO□□>□□Ü□j,□□`CT6ùJ□w¨>>CÙt⁸5□Nê'□□□i%Â³,□Z□I 1Aqàâ□®×-µ¸□□*[i□Ícw□□Ý_ïã£ÏÏX□□T□□¬\¦ -÷#ªÚÉÜ2è□□ êÍ⁹IG□□U²¸þ9¢□□òò¿ÿá□8éÑðYT÷□p'%X□@□□±□□±¿·ßóaòG Þ±□W□□ □'ß□□?n7¸□□□ÁZR□□Ý□¶□je□□s□ }Üï±u□Çû□céqïü□□õê□□}□□Þ´eÓ□tÆ□□Xåò□óË¨£åBäÔßé□ Z49paEÉ¦¦õAñüÑÒß·ÛÂ 0£(□#m□¹-□□□_□Q 8úÍby»Sò□Ó□□□ □]□□Aá³¨□P□□Î□¥Ü,®□□□õª□Ùà
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

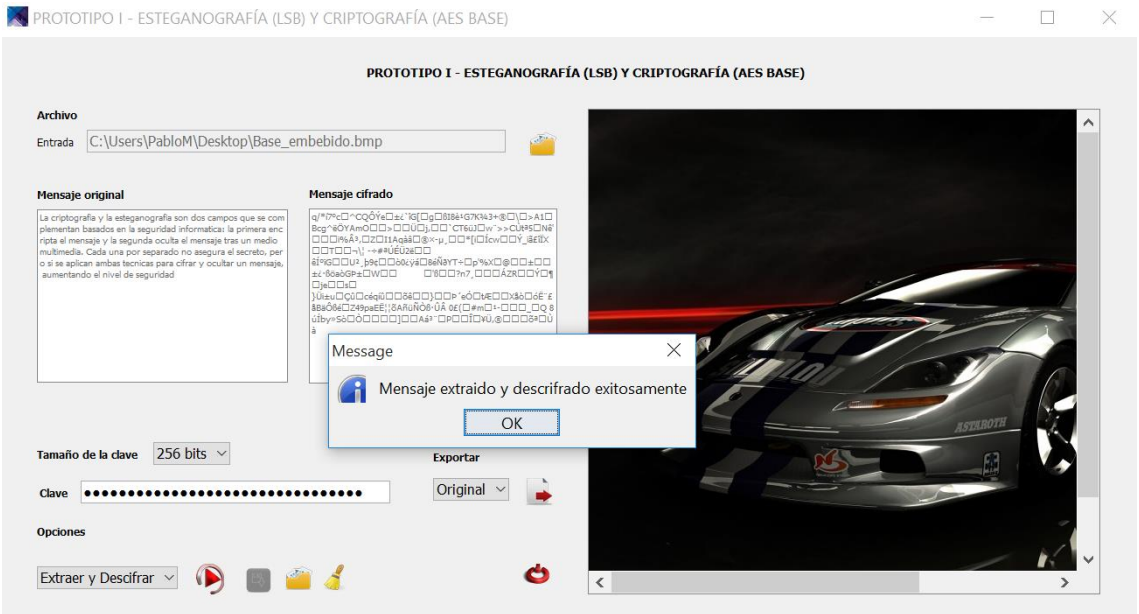


Figura 6-4 Captura del resultado de descifrado y extracción del Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

4.1.2. Prototipo II

El Prototipo II que se ha desarrollado utiliza el nuevo algoritmo criptográfico 2NAES con la incorporación del método esteganográfico LSB (Least Significant Bit).

4.1.2.1. Cifrado y embebido

4.1.2.1.1. Clave de 128 bits

Para probar el proceso de cifrado y embebido con el Prototipo II desarrollado con clave de 128 bits, se utilizan los datos de la Tabla 13-4.

Tabla 13-4 Datos para ejecución del Prototipo II con clave de 128 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles
Imagen de entrada:	
Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600
Clave (128 bits):	}yl%OckF x){gI~o
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo II con clave de 128 bits, se obtiene el resultado mostrado en la Tabla 14-4 y la Figura 7-4.

Tabla 14-4 Mensaje cifrado y embebido con el Prototipo II con clave de 128 bits

Mensaje cifrado:	]□C.X□□`ò½□□□P/□o² ¬□□{ý³□r□á7□Ô+□ÍSnYÈG4□eniHd □□□vKpy"M□|rR□ÑÉ+dg□¹BÌÐ□®□ü4□ð)□>□3□v□□□[ÿK2K □)»p`□Á°F"};3□öÒÈá□ªÙÐ0ò□£@Våê#□ÚÅÀ¦□ÃGbÒ_Ê v]`□U= 3^□ÌvÐ%□Si□ÃT□ÙÕè$□w□TÒFz□1ÿÄRò□ÞÁÉèÂ#□Ò¸i¨i□»|b_ □[□ç7ç@p□1<´«Â□□wÐ□□□□És□Iy{HO□4pÕC0□¦|□Ç¼X;_□ sp8□5ö□□□ßÁàCN{Åº|□□ÐÂfÿÑc¤ÌGþrpJt□□@Çô^Í□jþMÈèv»□: @H0qã□ÿc¾□(□'□//ó[¶þ□Ì¿□□èX[Îu□¹□ÍÒyØ□çVãk□fz8\□ù¹□ 5□oxrB□□

Realizado por: Méndez Pablo, 2015

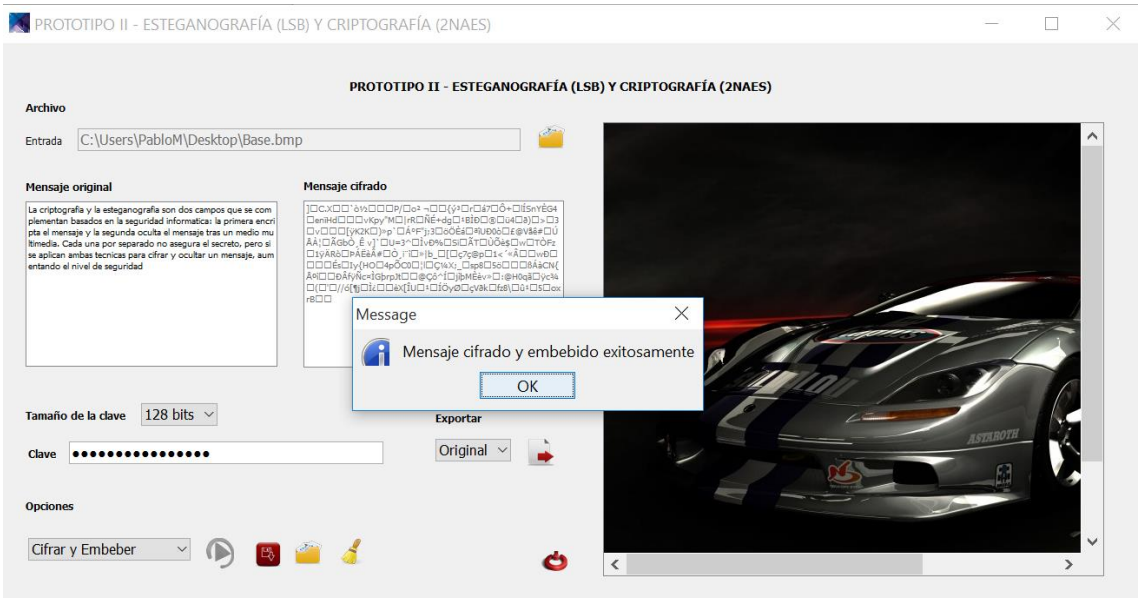


Figura 7-4 Captura del resultado de cifrado y embebido del Prototipo II con clave de 128 bits
Realizado por: Méndez Pablo, 2015

4.1.2.1.2. Clave de 192 bits

Para probar el proceso de cifrado y embebido con el Prototipo II desarrollado con clave de 192 bits, se utilizan los datos de la Tabla 15-4.

Tabla 15-4 Datos para ejecución del Prototipo II con clave de 192 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles
Imagen de entrada:	

Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600
Clave (192 bits):	<E?<E?<E?<E?<E?<E?<E?<E?
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo II con clave de 192 bits, se obtiene el resultado mostrado en la Tabla 16-4 y la Figura 8-4.

Tabla 16-4 Mensaje cifrado y embebido con el Prototipo II con clave de 192 bits

Mensaje cifrado:	¼□b®□Ý8□Z [□ÝAÓ□«Jh<Tð4¸l□'*□L ä□□Ø|ÂNÛ]□□";^□□,¶Â?Ú¾□àx□9N□4¶'ÊS AF□ß□□0□ £□□ÓÕ□ÓpG□H□gÞp N□□ï□{□□ì÷sW·áZ\□ÝZa/□□1aÒ¿ÖfºÌ±o□ý□ãÌ□#ñ□Î¢SÂA3f□ Õ<□OJ□¨¡Ü□ü□□õM□÷²□ò´□PdJÃý□ãÎ□Oö-þ□.uý□ZèQ□wÍ® yAå□□□H□□□7òÌã□.\□Æ□÷HõZıq¢ªLU□æº@□□□Ò\□□%ø□ □SÊÈ=en{EeyßÈ□-@Ù□0ÃÀF□4%□□ç¤Ü□þØaG6ÅÃp□ã¶□¼□Ù* \□/XÉ³□Å□Íñ□□□□!þ□MqéÏ□áÒ:ø□°¯(¿ùÂ"PP,□□¥RÅÂ\□Ú□L ý□©M!`'□□V´Ð¹□/?Áo□

Realizado por: Méndez Pablo, 2015

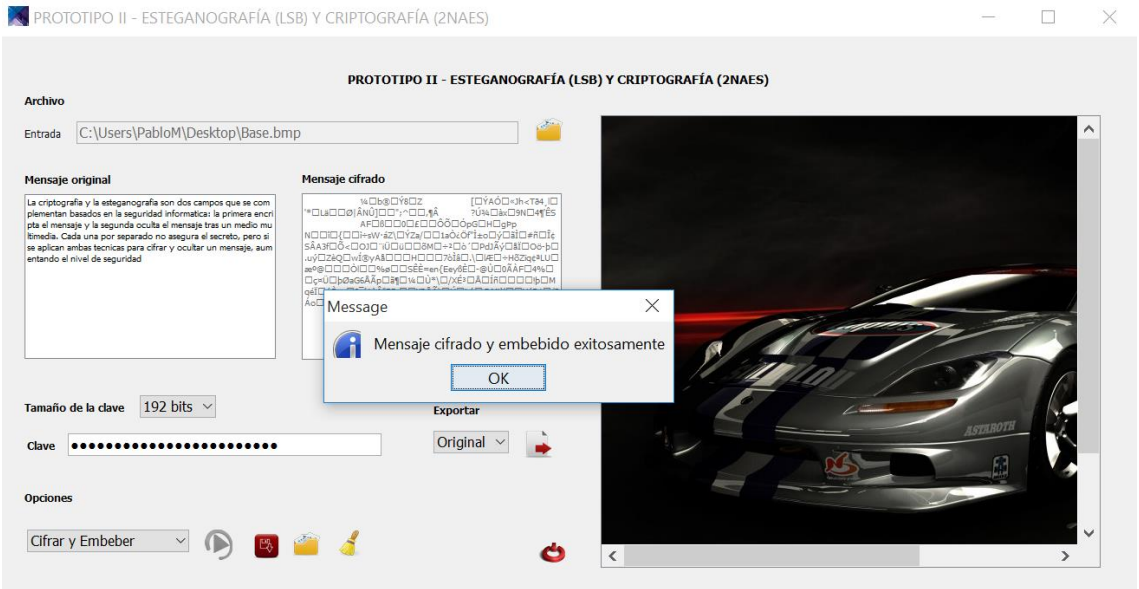


Figura 8-4 Captura del resultado de cifrado y embebido del Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.1.2.1.3. Clave de 256 bits

Para probar el proceso de cifrado y embebido con el Prototipo II desarrollado con clave de 256 bits, se utilizan los datos de la Tabla 17-4.

Tabla 17-4 Datos para ejecución del Prototipo II con clave de 256 bits

Nombre de la imagen de entrada:	Base
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles
Imagen de entrada:	
Nombre de la imagen de salida:	Base_embebido
Formato de la imagen de salida:	BMP
Dimensiones de la imagen de salida:	800x600
Clave (256 bits):	<E?<E?<E?<E?<E?<E?<E?<E?<E?<E?<E
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de cifrado, se realiza el proceso de embebido con el Prototipo II con clave de 256 bits, se obtiene el resultado mostrado en la Tabla 18-4 y la Figura 9-4.

Tabla 18-4 Mensaje cifrado y embebido con el Prototipo II con clave de 256 bits

Mensaje cifrado:	□Þí¸□□ª/(öã□W□P□9.k×□□á{ÎYó%Þð□ªGgÅd«gs□¶□□R#ù□^ Ý¶K^A□□□9□m□Ïj̃X□Z□õ[Ó□ãê 9ÅY&ÇNÞæ□r(□□°□pN¸j □Ké□□×I□□ÏÀ□a□z1=ù□ã□8(¸íx -□□%□¤□□O½□ýb□é ¼"8gµ□□D÷:<jÞF²ý□GÐÌ¿ü¦□Ï□□□□□Þ□L®~□=ÓLcæi+□yøÞ p*.´i'□"¬□□□Ô>èG0>Î×±Yíõ□ã□À÷êÂº□□q?$FM7 P□j□□È□6UÉ´□eÄ□É3□□L□□0□ªqzO¯-ù-ò¦,´ú□âèzÉ>a}□OÂld sýÃç□[□©8º6□Ñ~□□×R□Ûq÷Ëî*÷□÷□□□ÃsÞ¥3§Ì0)zÜ¬I/r□□n □□À)v¤ıþ□¸□¼ÍZbê□<¼□-"□e□□¯Æı□

Realizado por: Méndez Pablo, 2015

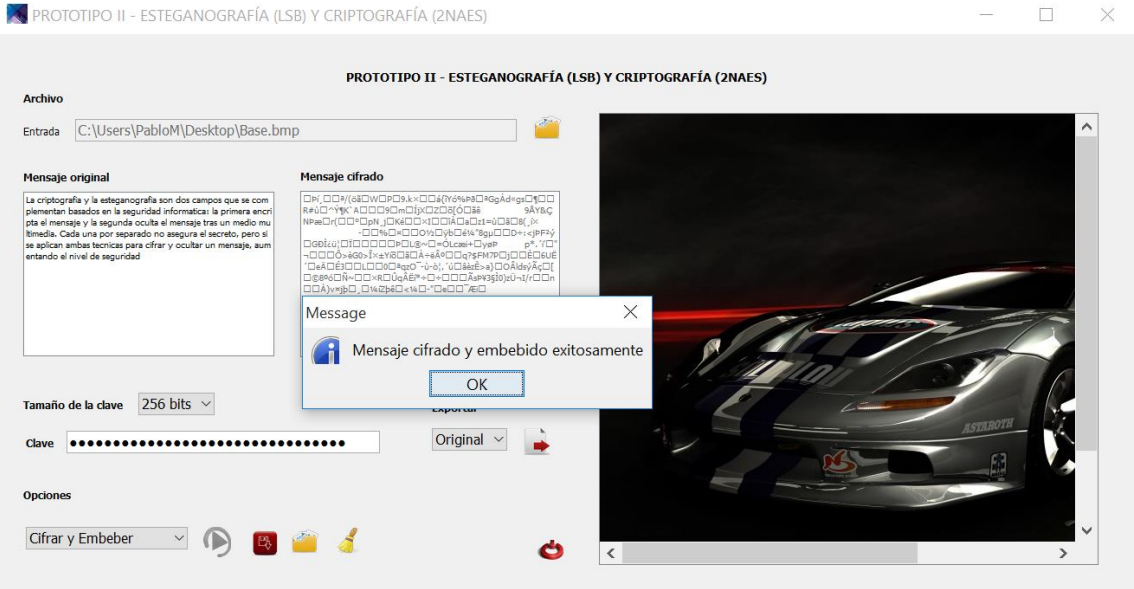


Figura 9-4 Captura del resultado de cifrado y embebido del Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

4.1.2.2. Extracción y descifrado

4.1.2.2.1. Clave de 128 bits

Para probar el proceso de extracción y descifrado con el Prototipo II desarrollado con clave de 128 bits, se utilizan los datos de la Tabla 19-4.

Tabla 19-4 Datos para ejecución del Prototipo II con clave de 128 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles
Imagen de entrada:	
Clave (128 bits):	}yl%OckF x){gI~o

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo II con clave de 128 bits, se obtiene el resultado mostrado en la Tabla 20-4 y la Figura 10-4.

Tabla 20-4 Mensaje extraído y descifrado con el Prototipo II con clave de 128 bits

Mensaje cifrado:	]□C.X□□`ò½□□□P/□o² ¬□□{ý³□r□á7□Ô+□ÍŠnYÈG4□enïHd □□□vKpy"M□|rR□ÑÉ+dg□¹BÌÐ□®□ü4□ð)□>□3□v□□□[ýK2K □)»p`□Á°F"j;3□öÖÈá□ªÎUÐ0ò□£@Våê#□ÚÅÀ¦□ÃGbÒ¸Ê v]`□U= 3^□ÌvÐ%□Si□ÃT□ÙÕè$□w□TÒFz□1ýÄRò□ÞÁËèÂ#□Ò¸ï¨ï□»|b_ □[□ç7ç@p□1<´«Â□□wÐ□□□□És□Iy{HO□4pÕC0□¦!□Ç¼X;_□ sp8□5ö□□□ßÁàCN{Å°ï□□ÐÂfýÑc¤ÍGþrpJt□□@Çô^Í□jïþMÈèv»□: @H0qã□ÿc¾□(□'□//ó[¶ï□Ìż□□èX[ÎU□¹□ÍÖyØ□çVãk□fz8\□û¹□ 5□oxrB□□
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el m ensaje tras un medio multimedia. Cada una por separa do no asegura el secreto, pero si se aplican ambas tecn icas para cifrar y ocultar un mensaje, aumentando el niv el de seguridad

Realizado por: Méndez Pablo, 2015

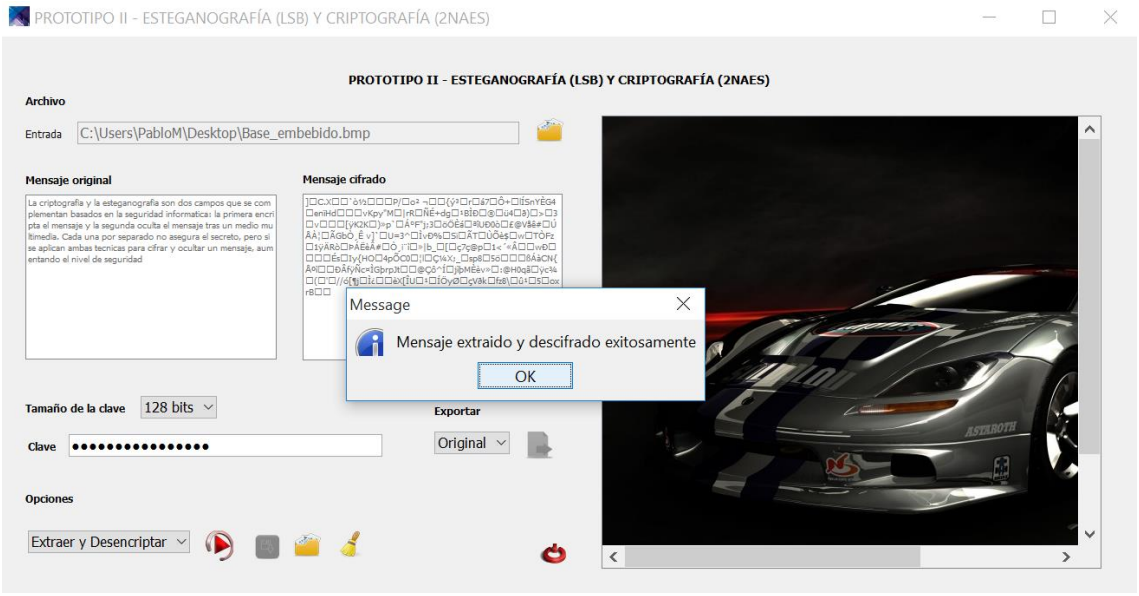


Figura 10-4 Captura del resultado de descifrado y extracción del Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.1.2.2.2. Clave de 192 bits

Para probar el proceso de extracción y descifrado con el Prototipo II desarrollado con clave de 192 bits, se utilizan los datos de la Tabla 21-4.

Tabla 21-4 Datos para ejecución del Prototipo II con clave de 192 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles

Imagen de entrada:	
Clave (192 bits):	<E?<E?<E?<E?<E?<E?<E?<E?

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo II con clave de 192 bits, se obtiene el resultado mostrado en la Tabla 22-4 y la Figura 11-4.

Tabla 22-4 Mensaje extraído y descifrado con el Prototipo II con clave de 192 bits

Mensaje cifrado:	¼□b®□Ý8□Z [□ÝÁÓ□«]h<T8ä_l□'*□L ã□□Ø|ÂNÛ]□□";^□□,¶Â ?Ú¾□àx□9N□4¶'ÊS AF□ß□□0□ £□□ÔÕ□ÓpG□H□gÞp N□□ïˆ□{□□ì÷sW·áZ\□ÝZa/□□1aÒ¿ÖfˆÌ±o□ý□ãÌ□#ñ□Î¢SÂA3f□ Õ<□OJ□¨iÚ□ü□□õM□÷²□ò´□PdJÃý□åÍ□Oö-þ□.uý□ZèQ□wÍ® yAå□□□H□□□7òÌâ□.\□Æ□÷HõZïq¢ªLU□æº@□□□Ò\□□%ø□ □SÉÈ=en{EeyßÉ□-@Ù□0ÃÂF□4%□□ç¤Ù□þØaG6ÃÃp□ã¶□¼□Ù* \□/XÉ³□Å□Íñ□□□□!þ□MqéÏ□áÒ:ø□°¯(¿ùÂ"PP,□□¥RÂÃ\□Ú□L ý□©M!`'□□V´Ð¹□/?Áo□
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

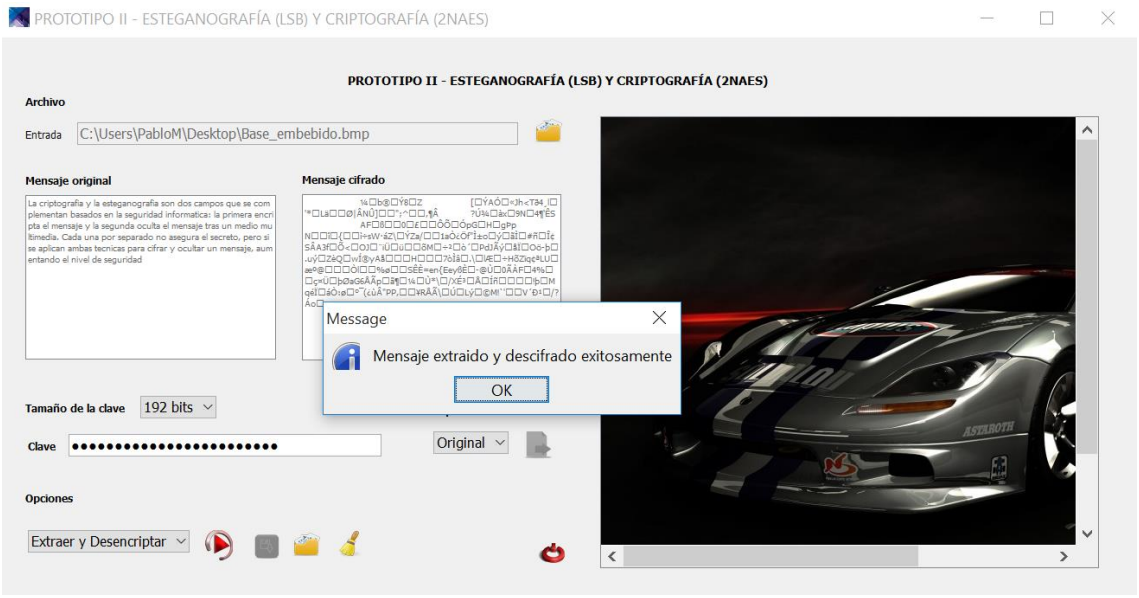


Figura 11-4 Captura del resultado de descifrado y extracción del Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.1.2.2.3. Clave de 256 bits

Para probar el proceso de extracción y descifrado con el Prototipo II desarrollado con clave de 256 bits, se utilizan los datos de la Tabla 23-4.

Tabla 23-4 Datos para ejecución del Prototipo II con clave de 256 bits

Nombre de la imagen de entrada:	Base_embebido
Formato de la imagen de entrada:	BMP
Dimensiones de la imagen de entrada:	800x600 pixeles
Imagen de entrada:	
Clave (256 bits):	<E?<E?<E?<E?<E?<E?<E?<E?<E?<E?<E

Realizado por: Méndez Pablo, 2015

Luego de ejecutar el proceso de extracción, se realiza el proceso de descifrado con el Prototipo II con clave de 256 bits, se obtiene el resultado mostrado en la Tabla 24-4 y la Figura 12-4.

Tabla 24-4 Mensaje extraído y descifrado con el Prototipo II con clave de 256 bits

Mensaje cifrado:	□Þí¸□□ª/(ôã□W□P□9.k×□□á{ÎYó%Þð□ªGgÀd«gs□¶□□R#ù□^ Ý¶K` A□□□9□m□ÎjX□Z□ð[Ó□ãê 9ÅY&ÇNÞæ□r(□□°□pN¸j □Ké□□×I□□ÌÀ□a□z1=ù□ã□8(¸íx -□□%□¤□□O½□ÿb□é ¼"8gµ□□D÷:<jÞF²ý□GÐÌ¿ü¦□Ï□□□□□Þ□L®~□=ÔLcæi+□yøÞ p*¸´ì'□"¬□□□Ô>èG0>Î×±Yï§□ã□À÷ëÂº□□q?$FM7 P□j□□È□6UÉ´□eÁ□É3□□L□□0□ªqzO¯-ù-ò¦,´ú□âèzÈ>a}□OÂld sýÃç□[□©8ºó□Ñ~□□×R□Ûq¸ÂÉí*÷□÷□□□Åsÿ¥3§Ì0)zÚ¬Ì/r□□n □□À)v¤¡þ□ .□¼ïZþê□<¼□-"□e□□¯Æi□
Mensaje original:	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el m ensaje tras un medio multimedia. Cada una por separa do no asegura el secreto, pero si se aplican ambas tecn icas para cifrar y ocultar un mensaje, aumentando el niv el de seguridad

Realizado por: Méndez Pablo, 2015

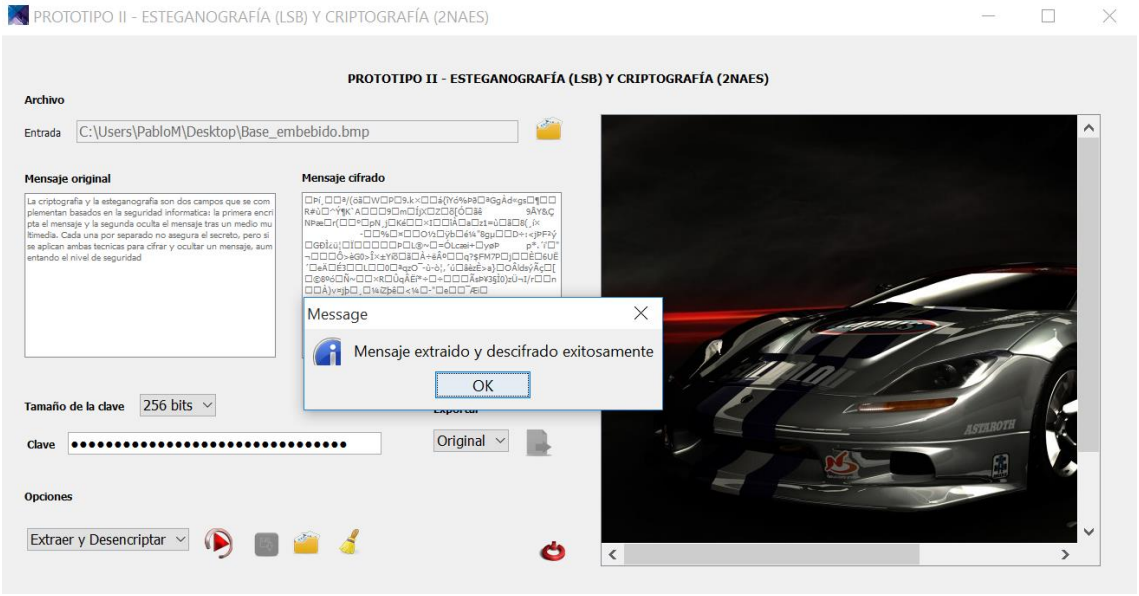


Figura 12-4 Captura del resultado de descifrado y extracción del Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

4.2. Análisis y comparación de resultados

Luego de realizar las pruebas en los escenarios establecidos con los Prototipos desarrollados, se procede a realizar el análisis y comparación de resultados obtenido en cada uno de ellos en los procesos criptográficos y esteganográficos:

4.2.1. Comparación de resultados criptográficos

4.2.1.1. Clave de 128 bits

Se comparan los mensajes cifrados que fueron generados con el Prototipo I y con el Prototipo II con clave de 128 bits, los cuales se muestran en la Tabla 25-4.

Tabla 25-4 Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 128 bits

Prototipo I	Prototipo II
X□0□{Vq□□□ß□@□}ÿ□□É;¬ô«□□□S·aØØÉ7×ª□X□4□W□é L3¶H□□□Gu□jÙ)CÍÎS□\§R□ñb□- □ÚH ëZí□YàÖÞ□1Î@□ìÀëÞ□`¶æ□F□□ïn□ □»c'□ùÿó9□D□£□□[©5 □□□«□ □□çTI□fî²□Ú□¬>□O □□£ø-½□□7Æ□öÔM□□£°`□Æ□ □§p»¯õõËuïî□□Ää□}õóñ 9Ï□/□déV°□ëÐ□öÊÑ□ïàåÑ□"Y □□Oåý¥ÈÜÚ□□x□Ë□4ê ÄÊA`□y□±½{ES□S9 □WÁÄñ-□□□Õ/Æ4|n□vÆE□□|ÍS»ùþÓ' °v3□-¯<G□ó□±J□□ê¸E□¨R□2□Ë□BN!Ô□Ò□ä&□□Û®"õ ØÊ5□O□8□V□Yv□@Éx00±#§Möë□□ÏÇ□=×YMå	]□C.X□□`ò½□□□P/□o² ¬□□{ý³□r□á7□Ô+□ÍSnYÈG4□eniHd □□□vKpy"M□|rR□ÑÉ+dg□¹BÌÐ□®□ü4□ð)□>□3□v□□□[ÿK2K □)»p`□Á°F"j;3□öÖÈá□ª1UÐ0ò□£@Våê#□ÚÅÀ¦□ÃGbÒ¸Ê v]`□U= 3^□ÌvÐ%□Si□ÃT□ÙÕè$□w□TÒFz□1ÿÄRò□ÞÁÉèÂ#□Ò¸i¨ï□»|b_ □[□ç7ç@p□1<´«Â□□wÐ□□□□És□Iy{HO□4pÕC0□¦!□Ç¼X;_□ sp8□5ö□□□ßÁàCN{Å°ï□□ÐÂfýÑc¤ÌGþrpJt□□@Çô^Í□jìþMÉèv»□: @H0qã□ÿc¾□(□'□//ó[¶□Ì¿□□èX[ÎU□¹□ÍÖyØ□çVäk□fz8\□û¹□ 5□oxrB□□

Realizado por: Méndez Pablo, 2015

4.2.1.2. Clave de 192 bits

Se comparan los mensajes cifrados que fueron generados con el Prototipo I y con el Prototipo II con clave de 192 bits, los cuales se muestran en la Tabla 26-4.

Tabla 26-4 Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 192 bits

Prototipo I	Prototipo II
¼£ZnÌE®*□R÷{9aý,Ì&üØ¼Ø<c$□ÐQ□»□i□-²i)å@□T□□ùÈQÍã□ »□Û,k{<□□(□Í-Ô×Í6á□□ {□\□[□±_ÌÐĪ9Ú□¿5□¨□AÙ=¸1VfÑî× □³ñ□ä□k¤È□È¨A§â□µ-□□Ê□NE□Û=szE□³JG É^6Xµ-+û□□□Ò«B\□□I~□´5□□Ñmr□/□□©□ñ□□□g4□§å êª(□/ □□ ï«j:□Ê¤i+ç□zï□«¨Ôê□®%-W'□u½Qþ□□µwÞ□öê=□@ì Ø¦Úó□D/V□□¥ô□´õø□Úï□□ma)ïÇq¨ □□B¶□¶#¢□NÔxXsÎY□□ß.å]»èm@Õõº□□Lâ±½ê□□-TúÎÝ□K□C □ÿhòQw÷<M@u\□□Íý□ZØä®|d□□□□¹~□□$□3□Tòµjt'©À	¼□b®□Ý8□Z [□ÝAÓ□«Jh<Tð4¸Ì□'*□L ä□□Ø|ÂNÛ]□□";^□□,¶Â?Ú¾□àx□9N□4¶'ÊS AF□ß□□0□ £□□ÔÕ□ÓpG□H□gÞp N□□ï□{□□ì÷sW·áZ\□ÝZa/□□1aÒ¿ÖfºÌ±o□ý□ãÌ□#ñ□Î¢SÂA3f□ Õ<□OJ□¨iÜ□ü□□õM□÷²□ò´□PdJÃý□åÏ□Oö-þ□.uý□ZèQ□wÍ® yAå□□□H□□□7òÌâ□.\□Æ□÷HõZïq¢ªLU□æº@□□□Ò|□□%ø□ □SÊÈ=en{EeyßÈ□-@Ù□0ÃÀF□4%□□ç¤Ü□þØaG6ÅÃp□ã¶□¼□Ù* \□/XÉ³□Å□Íñ□□□□!þ□MqéÏ□áÒ:ø□°¯(¿ùÂ"PP,□□¥RÂÃ\□Ú□L ý□©M!`'□□V´Ð¹□/?Áo□

Realizado por: Méndez Pablo, 2015

4.2.1.3. Clave de 256 bits

Se comparan los mensajes cifrados que fueron generados con el Prototipo I y con el Prototipo II con clave de 256 bits, los cuales se muestran en la Tabla 27-4.

Tabla 27-4 Comparación de mensajes cifrados con Prototipo I y Prototipo II con clave de 256 bits

Prototipo I	Prototipo II
q/*ï7°c□^CQÔÝe□±¿`ïG[□g□ßI8è¹G7K¾3+®□\□>A1□Bcg^ëÖÝ AmO□□>□□Ù□j,□□`CT6ùJ□w¨>>CÙt³5□Nê'□□□i%Â³,□Z□I 1Aqàâ□®×-µ¸□□*[ì□Ícw□□Ý_íãɪÎX□□T□□¬\¦ -÷#ªÚÉÙ2ë□□ êÍº!G□□U²¸þ9¢□□ò0¿ÿá□8éÑðYT÷□p'%X□@□□±□□±¿·ßöaòG Þ±□W□□ □'ß□□?n7¸□□□ÁZR□□Ý□¶□je□□s□ }Ùi±u□Çû□céqïû□□õê□□}□□Þ´eÓ□tÆ□□Xåò□óÉ¨£åBäÔßé□ Z49paEÉ¦¦õAñüÑÒß·ÛÂ 0£(□#m□¹-□□□_□Q 8úÍby»Sò□Ò□□□ □]□□Aá³¨□P□□Î□¥Ú,®□□□õª□Ùà	□Þí¸□□ª/(óã□W□P□9.k×□□á{ÎYó%Þð□ªGgÀd«gs□¶□□R#ù□^ Ý¶K` A□□□9□m□ÎjX□Z□õ[Ó□âê 9ÅY&ÇNÞæ□r(□□º□pN¸j □Ké□□×I□□ìÀ□a□z1=ù□ã□8(¸íx -□□%□¤□□O½□ÿb□é ¼"8gµ□□D÷:<jÞF²ý□GÐÌ¿ü¦□Í□□□□□Þ□L®~□=ÓLcæi+□yøÞ p*.´i'□"¬□□□Ô>êG0>Î×±Yïõ□ã□À÷ëÂº□□q?$FM7 P□j□□È□6UÉ´□eÃ□É3□□L□□0□ªqzO¯-ù-ò¦,´ú□âèzÈ>a}□OÃld sýÃç□[□©8ºó□Ñ~□□×R□Ûq.ÂËf*÷□÷□□□ÂsÞ¥3§Ì0)zÜ¬Ì/r□□n □□À)v¤þ□.□¼ïZþê□<¼□-"□e□□¯Æi□

Realizado por: Méndez Pablo, 2015

4.2.2. Comparación de resultados esteganográficos

4.2.2.1. Prototipo I

4.2.2.1.1. Clave de 128 bits

Se comparan las imágenes “Base.bmp” y “Base_embebido.bmp”, utilizada y generada por el Prototipo I con clave de 128 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 13-4 y Figura 14-4.

Base

Figura 13-4 Imagen "Base.bmp" con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 14-4 Imagen "Base_embebido.bmp" con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes "Base.bmp "Base_embebido.bmp", con el Prototipo I con clave de 128 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 28-4 y Figura 15-4, Figura 16-4.

Tabla 28-4 Tamaños de las imágenes con el Prototipo I con clave de 128 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

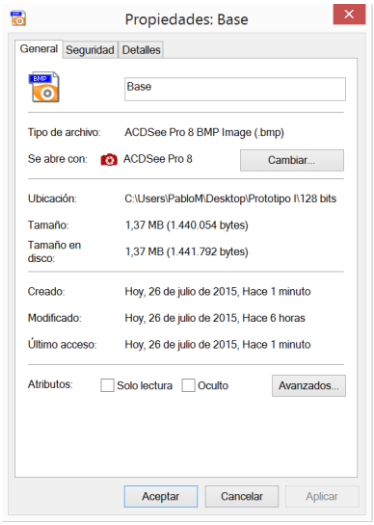


Figura 15-4 Tamaño de la imagen "Base.bmp" con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

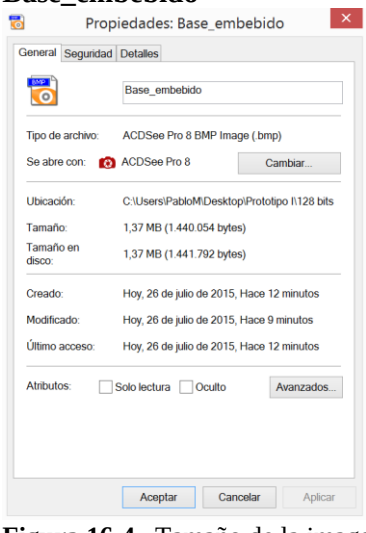


Figura 16-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo I con clave de 128 bits.

Al comparar los dos archivos hexadecimales de la imagen original “Base.bmp” y la imagen esteganografiada “Base_embebido.bmp” generada por el Prototipo I con clave de 128 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 17-4.

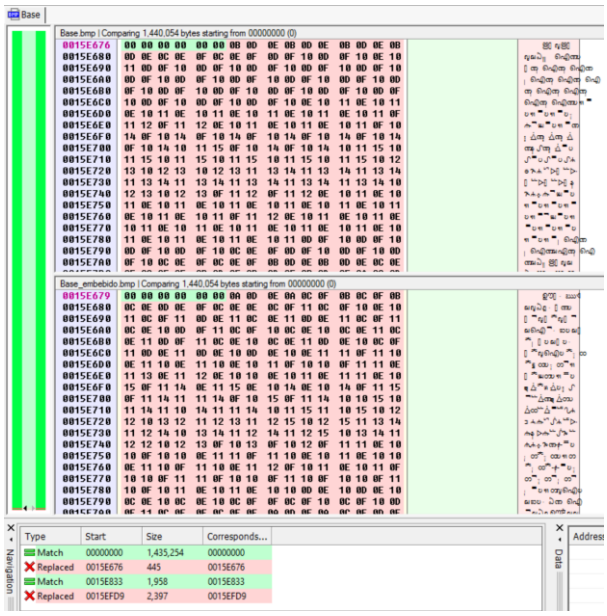


Figura 17-4 Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 128 bits
Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 128 bits con el Prototipo I, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 18-4.

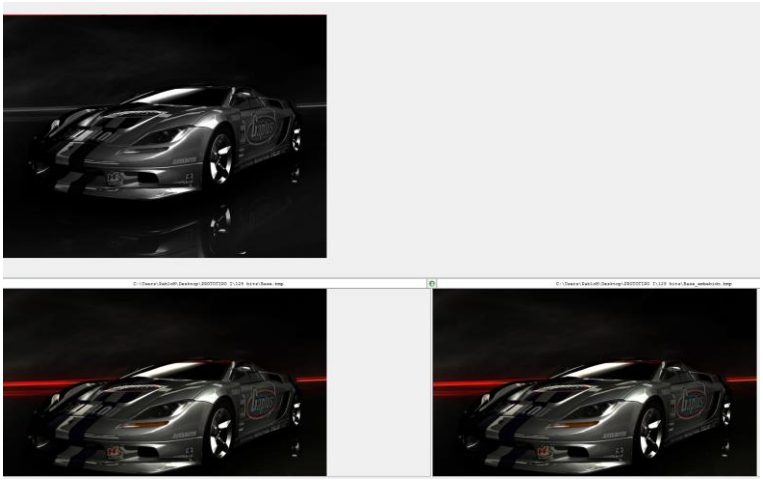


Figura 18-4 Resultado de comparación de imágenes
Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 19-4.

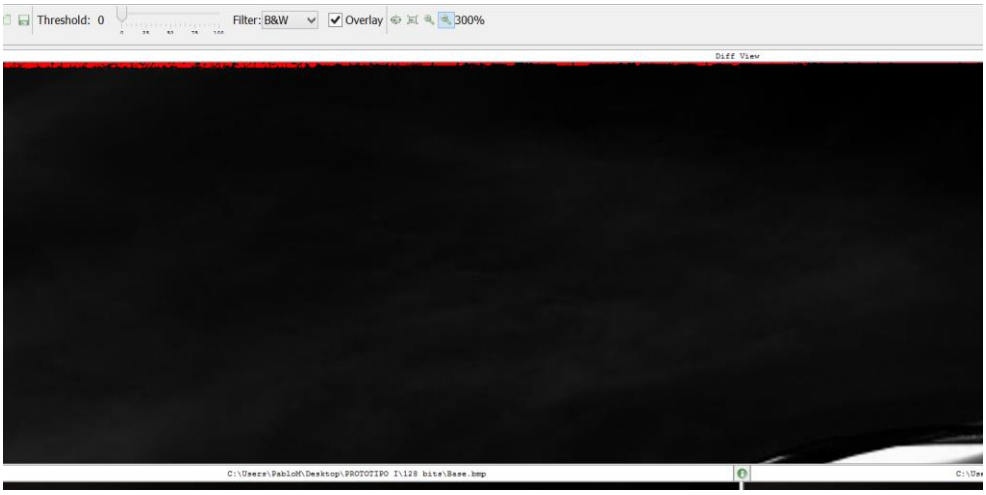


Figura 19-4 Diferencia en los pixeles entre las imágenes comparadas
Realizado por: Méndez Pablo, 2015

4.2.2.1.2. Clave de 192 bits

Se comparan las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo I con clave de 192 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 20-4 y Figura 21-4.

Base

Figura 20-4 Imagen "Base.bmp" con el Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 21-4 Imagen "Base_embebido.bmp" con el Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes "Base.bmp y "Base_embebido.bmp", con el Prototipo I con clave de 192 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 29-4 y Figura 22-4, Figura 23-4.

Tabla 29-4 Tamaños de las imágenes con el Prototipo I con clave de 192 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

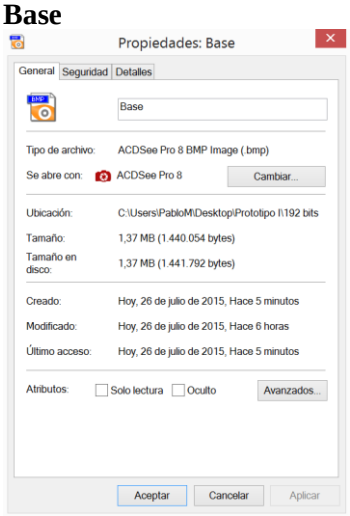


Figura 22-4 Tamaño de la imagen "Base.bmp" con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

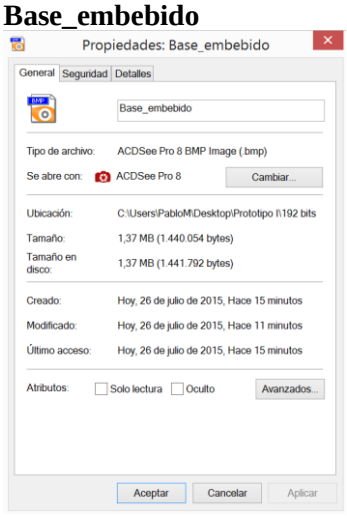


Figura 23-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo I con clave de 192 bits.

Al comparar los dos archivos hexadecimales de la imagen original "Base.bmp" y la imagen esteganografiada "Base_embebido.bmp" generada por el Prototipo I con clave de 192 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 24-4.

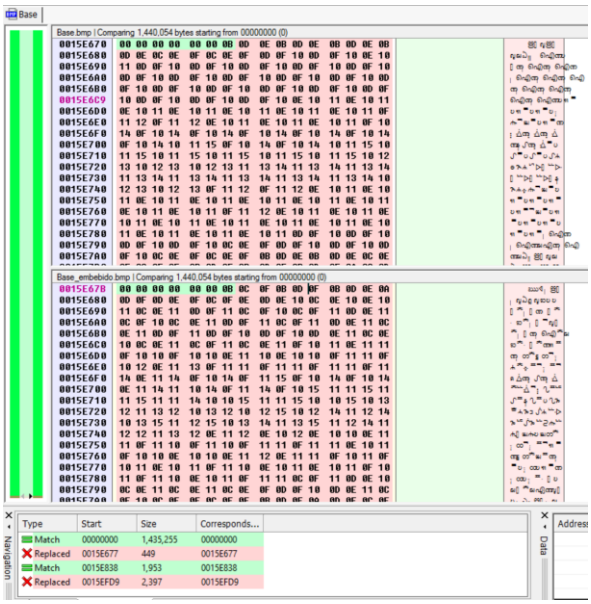


Figura 24-4 Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 192 bits con el Prototipo I, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 25-4.

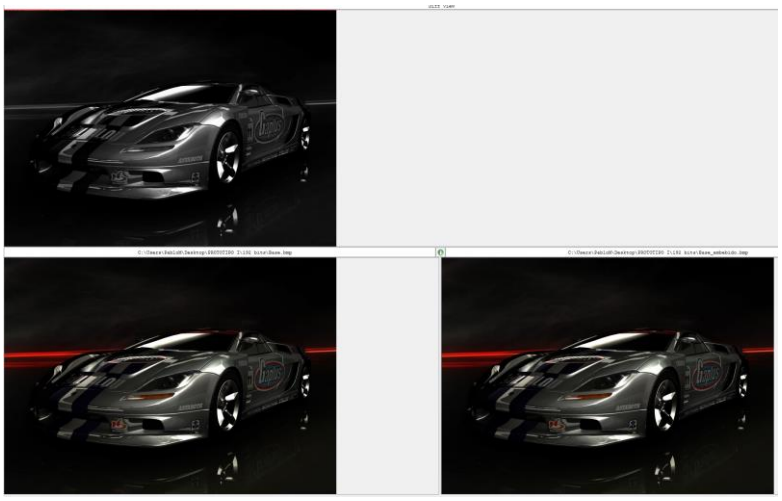


Figura 25-4 Resultado de comparación de imágenes

Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 26-4.

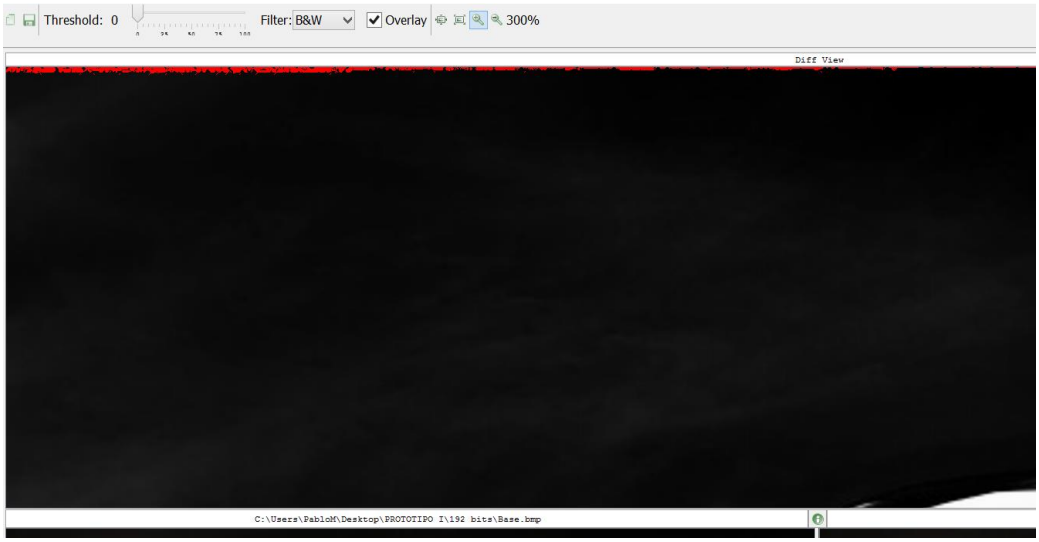


Figura 26-4 Diferencia en los pixeles entre las imágenes comparadas
Realizado por: Méndez Pablo, 2015

4.2.2.1.3. Clave de 256 bits

Se comparan las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo I con clave de 256 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 27-4 y Figura 28-4.

Base

Figura 27-4 Imagen "Base.bmp" con el Prototipo I con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 28-4 Imagen "Base_embebido.bmp" con el Prototipo I con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes "Base.bmp y "Base_embebido.bmp", con el Prototipo I con clave de 256 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 30-4 y Figura 29-4, Figura 30-4.

Tabla 30-4 Tamaños de las imágenes con el Prototipo I con clave de 256 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

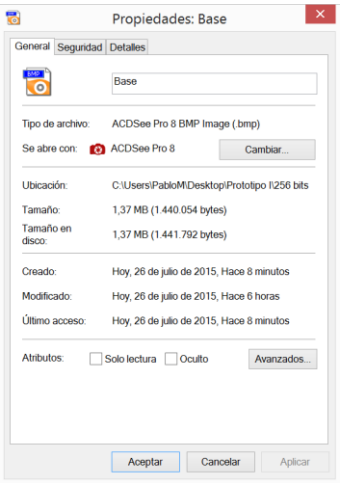


Figura 29-4 Tamaño de la imagen "Base.bmp" con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

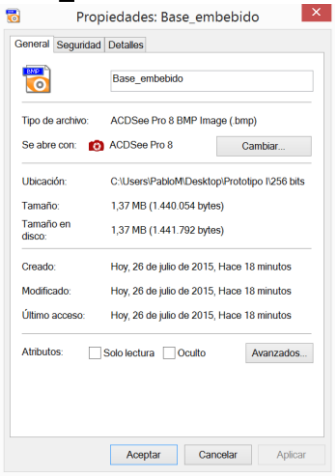


Figura 30-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo I con clave de 256 bits.

Al comparar los dos archivos hexadecimales de la imagen original "Base.bmp" y la imagen esteganografiada "Base_embebido.bmp" generada por el Prototipo I con clave de 256 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 31-4.

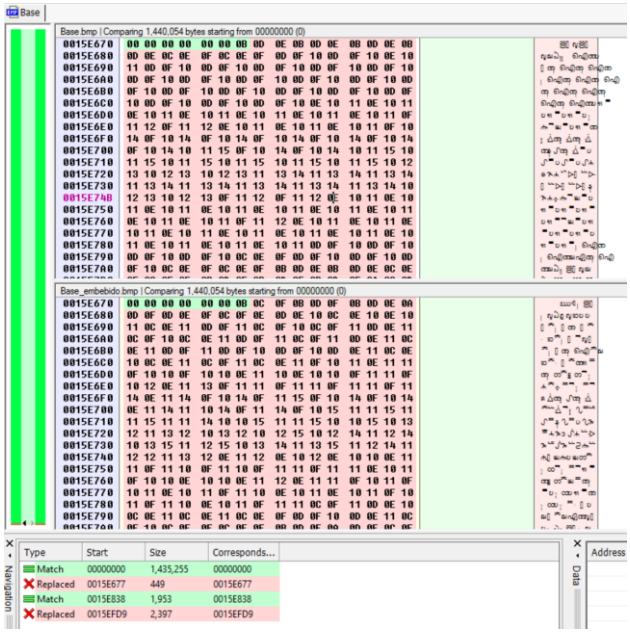


Figura 31-4 Comparación de código hexadecimal de las imágenes con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 256 bits con el Prototipo I, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 32-4.

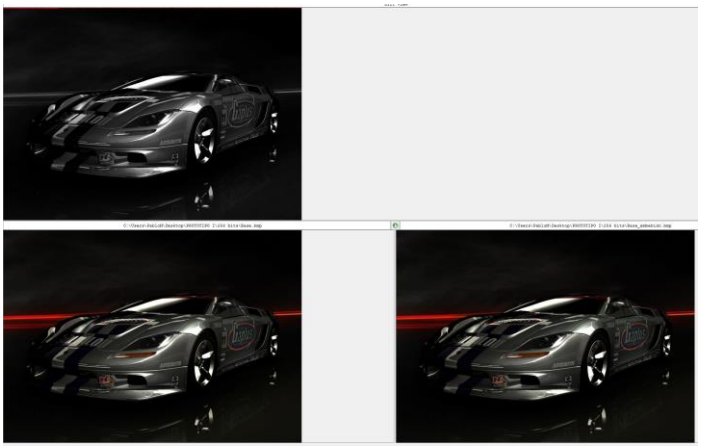


Figura 32-4 Resultado de comparación de imágenes
Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 33-4.

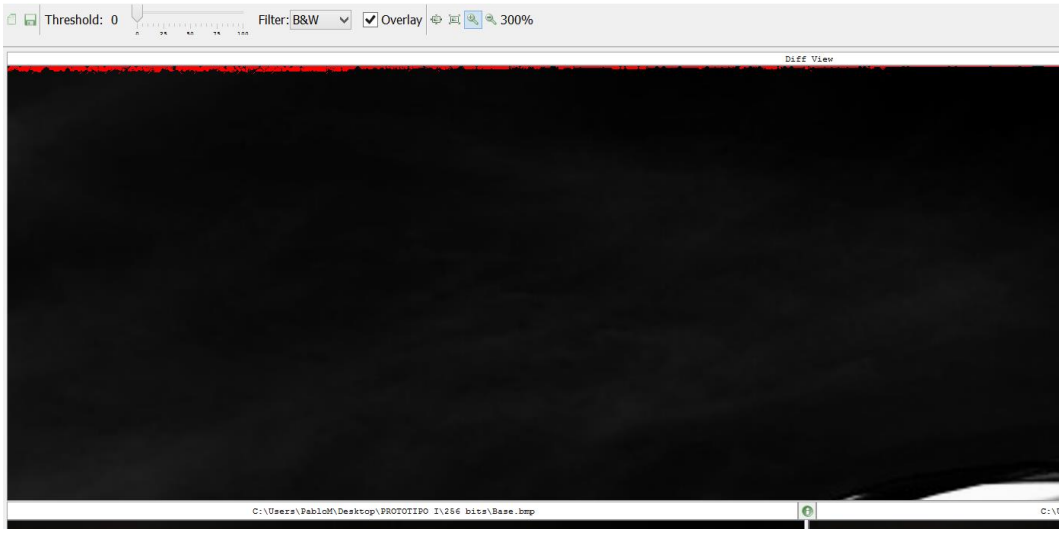


Figura 33-4 Diferencia en los pixeles entre las imágenes comparadas
|**Realizado por:** Méndez Pablo, 2015

4.2.2.2. Prototipo II

4.2.2.2.1. Clave de 128 bits

Se comparan las imágenes “Base.bmp” y “Base_embebido.bmp”, utilizada y generada por el Prototipo II con clave de 128 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 34-4 y Figura 35-4.

Base

Figura 34-4 Imagen “Base.bmp” con el Prototipo II con clave de 128 bits
Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 35-4 Imagen “Base_embebido.bmp” con el Prototipo II con clave de 128 bits
Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes “Base.bmp y “Base_embebido.bmp”, con el Prototipo II con clave de 128 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 31-4 y Figura 36-4, Figura 37-4.

Tabla 31-4 Tamaños de las imágenes con el Prototipo II con clave de 128 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

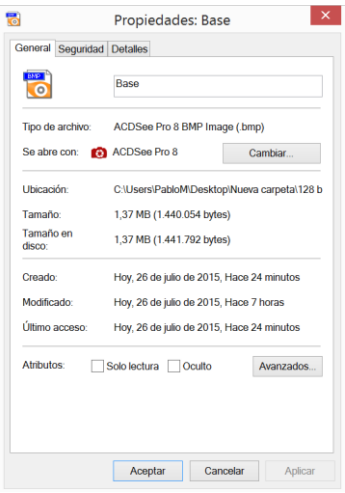


Figura 36-4 Tamaño de la imagen "Base.bmp" con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

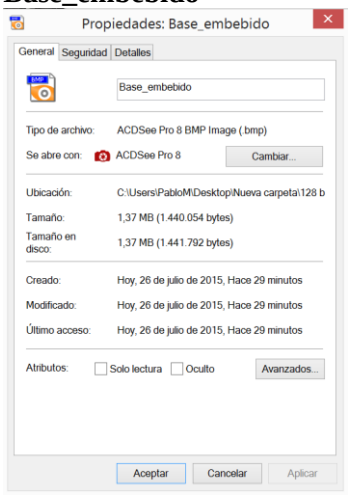


Figura 37-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo II con clave de 128 bits.

Al comparar los dos archivos hexadecimales de la imagen original "Base.bmp" y la imagen esteganografiada "Base_embebido.bmp" generada por el Prototipo II con clave de 128 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 38-4.

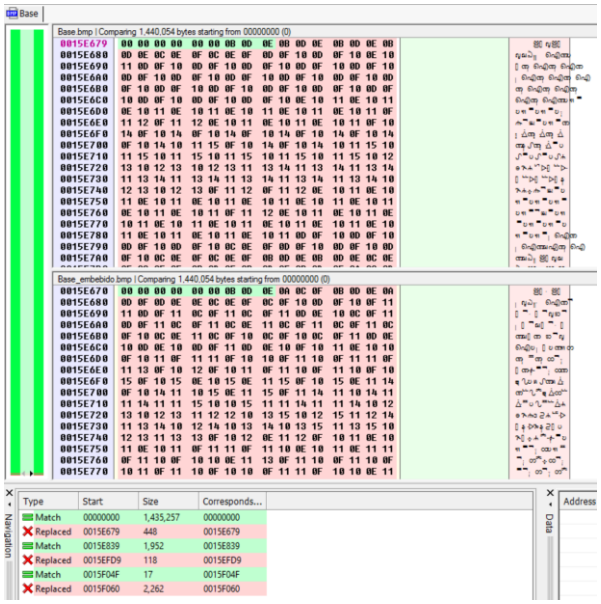


Figura 38-4 Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 128 bits con el Prototipo II, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 39-4.

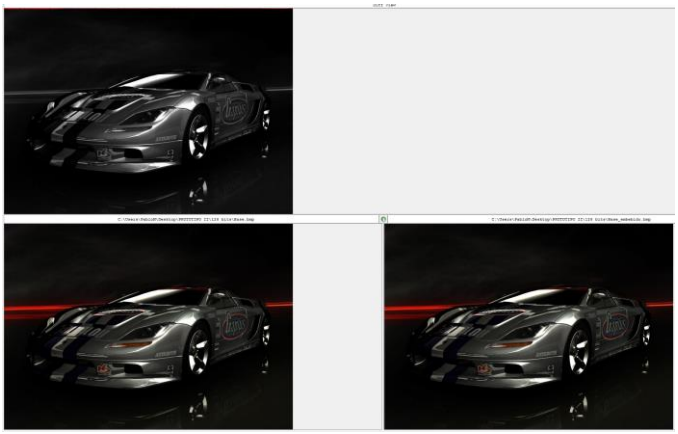


Figura 39-4 Resultado de comparación de imágenes
Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 40-4.

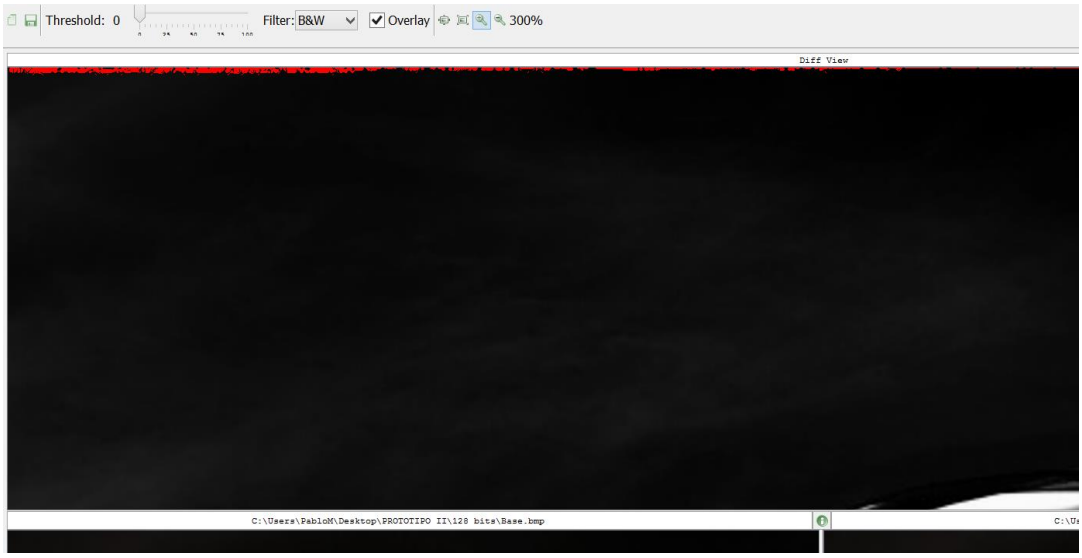


Figura 40-4 Diferencia en los pixeles entre las imágenes comparadas
Realizado por: Méndez Pablo, 2015

4.2.2.2.2. Clave de 192 bits

Se comparan las imágenes “Base.bmp” y “Base_embebido.bmp”, utilizada y generada por el Prototipo II con clave de 192 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 41-4 y Figura 42-4.

Base

Figura 41-4 Imagen “Base.bmp” con el Prototipo II con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 42-4 Imagen “Base_embebido.bmp” con el Prototipo II con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes “Base.bmp y “Base_embebido.bmp”, con el Prototipo II con clave de 192 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 32-4 y Figura 43-4, Figura 44-4.

Tabla 32-4 Tamaños de las imágenes con el Prototipo II con clave de 192 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

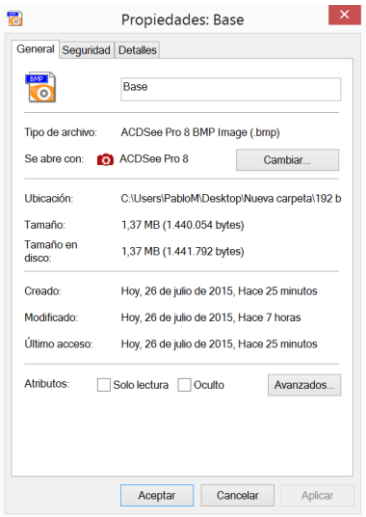


Figura 43-4 Tamaño de la imagen "Base.bmp" con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

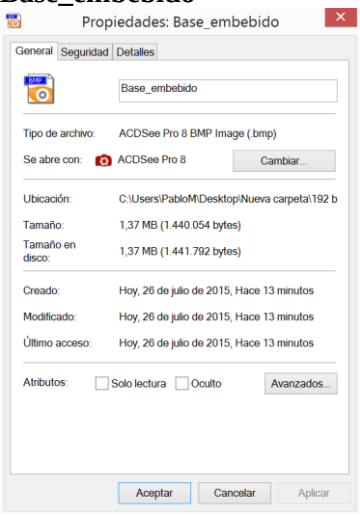


Figura 44-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo II con clave de 192 bits.

Al comparar los dos archivos hexadecimales de la imagen original "Base.bmp" y la imagen esteganografiada "Base_embebido.bmp" generada por el Prototipo II con clave de 192 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 45-4.

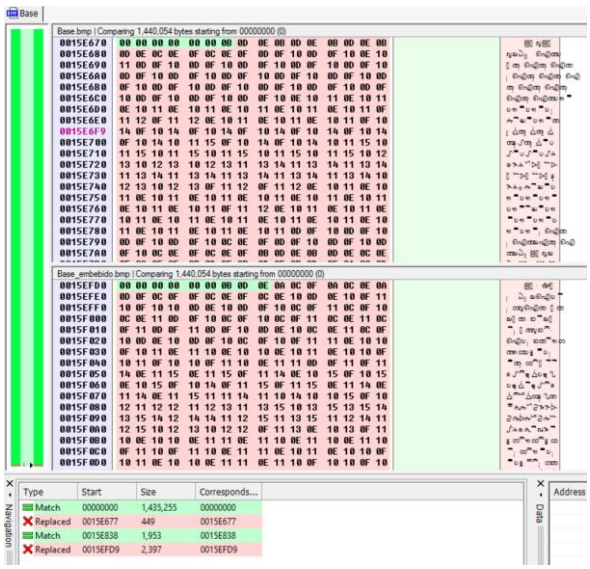


Figura 45-4 Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 192 bits con el Prototipo II, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 46-4.

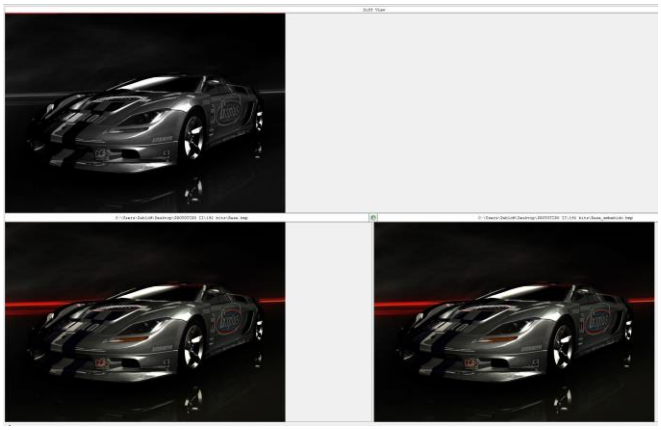


Figura 46-4 Resultado de comparación de imágenes
Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 47-4.

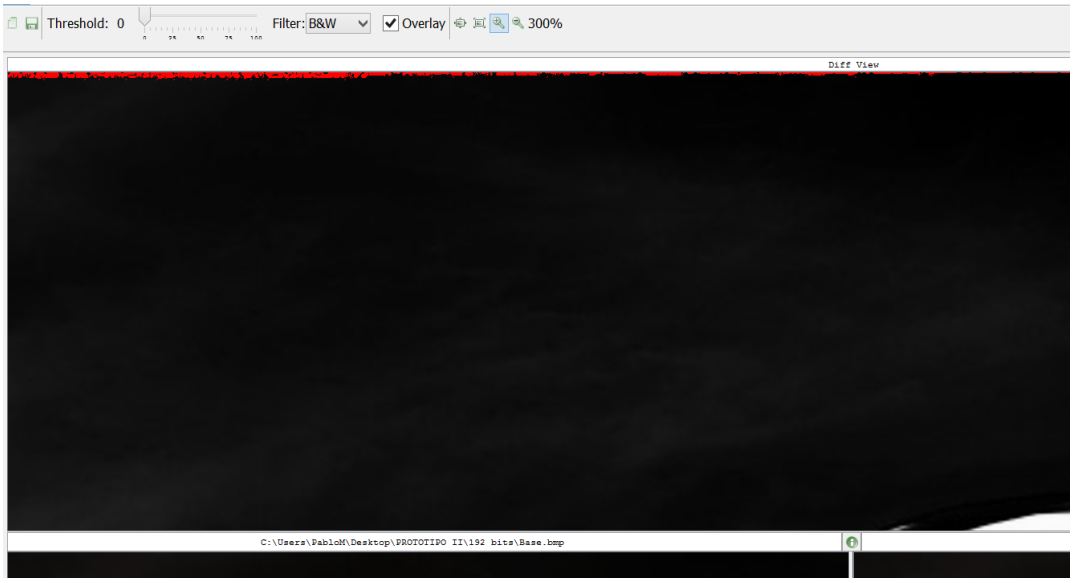


Figura 47-4 Diferencia en los pixeles entre las imágenes comparadas
Realizado por: Méndez Pablo, 2015

4.2.2.2.3. Clave de 256 bits

Se comparan las imágenes “Base.bmp” y “Base_embebido.bmp”, utilizada y generada por el Prototipo II con clave de 256 bits, en las cuales no se puede apreciar una diferencia a simple vista, como se muestra en las Figura 48-4 y Figura 49-4.

Base

Figura 48-4 Imagen “Base.bmp” con el Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Base_embebido

Figura 49-4 Imagen “Base_embebido.bmp” con el Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Se comparan los tamaños de las imágenes “Base.bmp y “Base_embebido.bmp”, con el Prototipo II con clave de 256 bits, demostrando que con la técnica esteganográfica LSB no existe variación de las mismas, como se muestran en la Tabla 33-4 y Figura 50-4, Figura 51-4.

Tabla 33-4 Tamaños de las imágenes con el Prototipo II con clave de 256 bits

Nombre de Imagen	Tamaño (bytes)	Tamaño en disco (bytes)	Tamaño (MB)
Base.bmp	1.440.054	1.441.792	1,37
Base_embebido.bmp	1.440.054	1.441.792	1,37

Realizado por: Méndez Pablo, 2015

Base

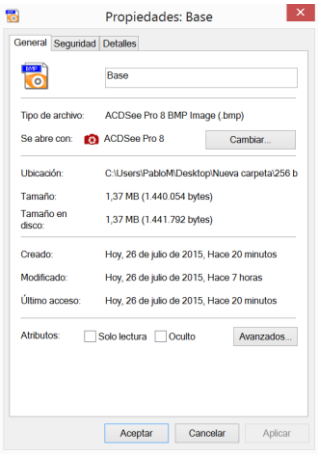


Figura 50-4 Tamaño de la imagen "Base.bmp" con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Base_embebido

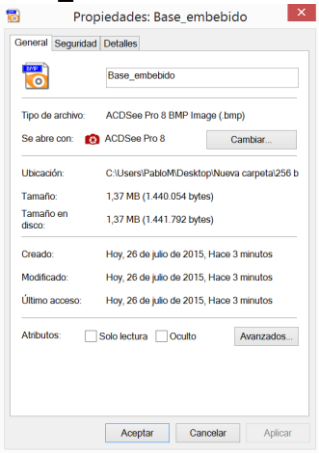


Figura 51-4 Tamaño de la imagen "Base_embebido.bmp" con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa FlexHEX para determinar el código hexadecimal de las imágenes "Base.bmp" y "Base_embebido.bmp", utilizada y generada por el Prototipo II con clave de 256 bits.

Al comparar los dos archivos hexadecimales de la imagen original "Base.bmp" y la imagen esteganografiada "Base_embebido.bmp" generada por el Prototipo II con clave de 256 bits, se determinan diferencias marcadas con color rojo, como se muestra en la Figura 52-4.

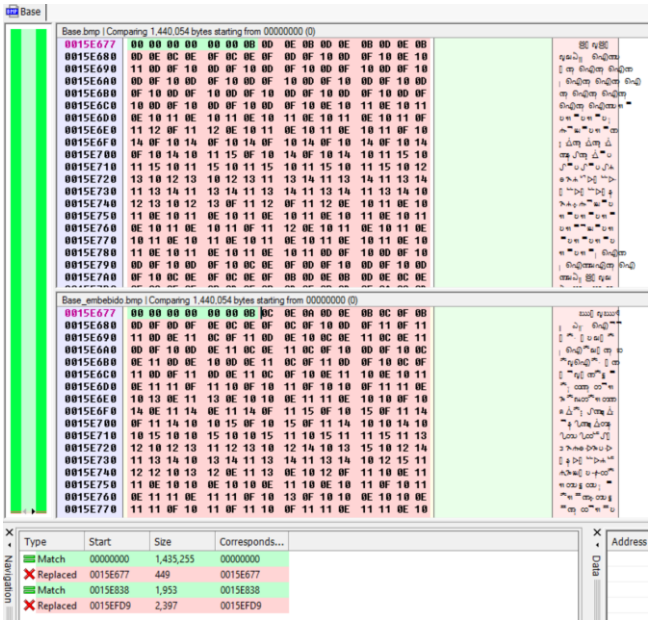


Figura 52-4 Comparación de código hexadecimal de las imágenes con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Se utiliza el programa Guiffy Image Diff para comparar las dos imágenes y comprobar que la información del mensaje cifrado con clave de 256 bits con el Prototipo II, se encuentra dentro de la imagen esteganografiada.

Luego de comparar las imágenes, se muestra en la parte superior la diferencia entre pixeles, como se muestra en la Figura 53-4.

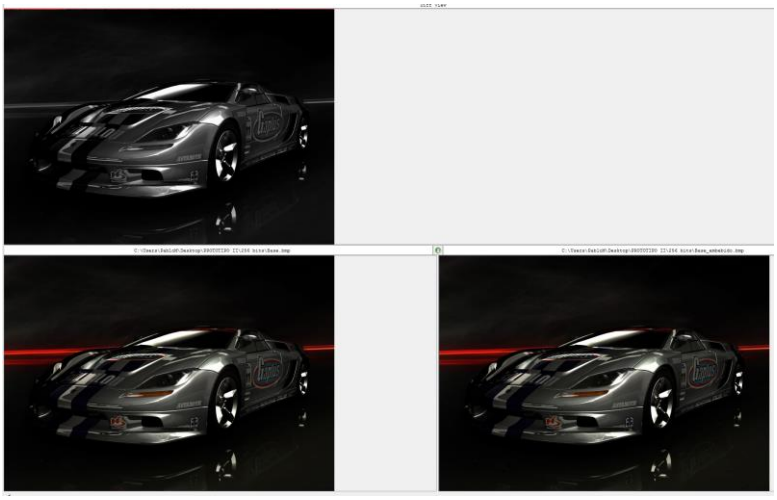


Figura 53-4 Resultado de comparación de imágenes
Realizado por: Méndez Pablo, 2015

Si se incrementa el zoom a 300%, se puede visualizar que, en la parte superior de la imagen, se encuentran diferencias en los pixeles, como se muestra en la Figura 54-4.

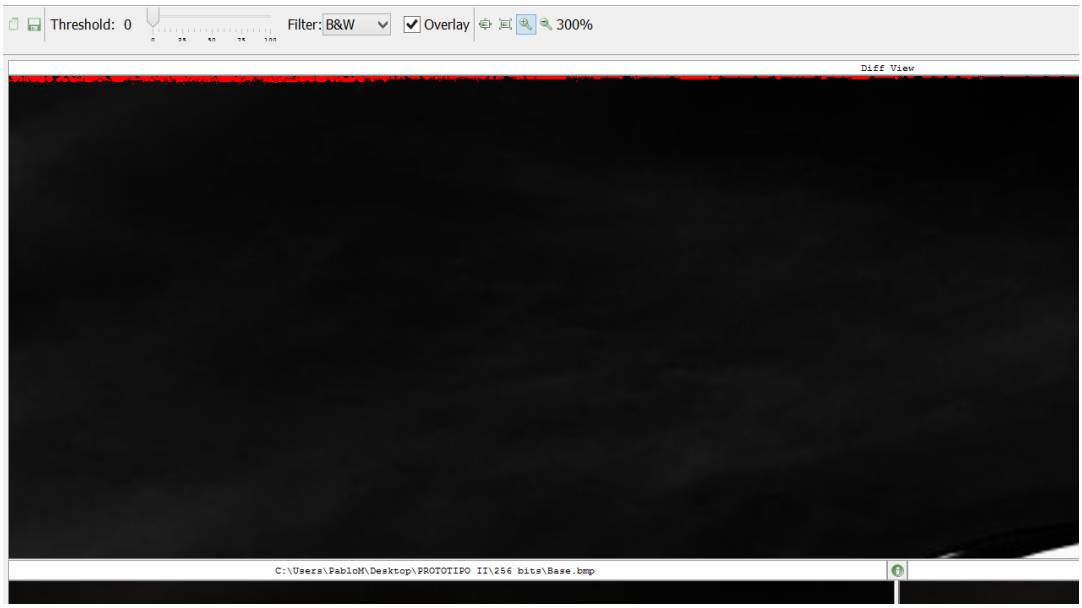


Figura 54-4 Diferencia en los pixeles entre las imágenes comparadas
Realizado por: Méndez Pablo, 2015

4.3. Prueba de hipótesis

4.3.1. Pruebas

Para la comprobación de la hipótesis se realizarán las pruebas en dos partes:

- Análisis de las características del algoritmo AES base implementado en el Prototipo I y del nuevo algoritmo desarrollado 2NAES implementado en el Prototipo II, para 4 indicadores de la variable dependiente (seguridad): No. claves utilizadas, No. Rondas, No. Funciones usadas por el algoritmo, No. Funciones ejecutadas por el algoritmo.
- Criptoanálisis de los mensajes cifrados con el algoritmo AES base implementado en el Prototipo I y del nuevo algoritmo desarrollado 2NAES implementado en el Prototipo II, para 5 indicadores de la variable dependiente (seguridad): Entropía, Histograma, N-Grama, Autocorrelación, Fuerza bruta.

Para medir los indicadores se utilizará la escala de Likert con valores promedio de los resultados obtenidos con el Prototipo I y el Prototipo II.

4.3.1.1. Análisis de características de los algoritmos

Se compara el algoritmo AES base implementado en el Prototipo I y el nuevo algoritmo 2NAES implementado en el Prototipo II, para los siguientes indicadores de la variable dependiente definida:

- No. claves utilizadas
- No. Rondas
- No. Funciones usadas por el algoritmo
- No. Funciones ejecutadas por el algoritmo

En la Tabla 34-4 se determina cada uno de los indicadores definidos anteriormente para la comparación del algoritmo AES base implementado en el Prototipo I y el nuevo algoritmo implementado 2NAES implementado en el Prototipo II (que incluye la nueva función ShiftColumn y la repetición de todo el proceso), con tamaños de bloque de 128 bits y tamaños de clave de 128 bits, 192 bits y 256 bits.

Tabla 34-4 Determinación de los indicadores definidos para comparar los algoritmos

No.	INDICADORES	PROTOTIPO I ALGORITMO AES BASE	PROTOTIPO II NUEVO ALGORITMO 2NAES	
			Con la inclusión de la nueva función ShiftColumn	Propuesta 2NAES
1	No. claves utilizadas	• 1 clave	• 1 clave	• 2 * (1 clave) = 2 claves
2	No. Rondas	Clave de: • 128 bits: 10 rondas • 192 bits: 12 rondas • 256 bits: 14 rondas	Clave de: • 128 bits: 10 rondas • 192 bits: 12 rondas • 256 bits: 14 rondas	Clave de: • 128 bits: 2* (10 rondas) = 20 rondas • 192 bits: 2 * (12 rondas) = 24 rondas • 256 bits: 2* (14 rondas) = 28 rondas
3	No. Funciones usadas por el algoritmo	Usa 4 funciones • AddRoundKey • SubByte • ShiftRow • MixColumn	Usa 5 funciones • AddRoundKey • SubByte • ShiftRow • ShiftColumn • MixColumn	Usa 5 funciones en la primera vuelta • AddRoundKey • SubByte • ShiftRow • ShiftColumn • MixColumn Usa 5 funciones en la segunda vuelta • AddRoundKey • SubByte • ShiftRow • ShiftColumn • MixColumn
4	No. Funciones ejecutadas por el algoritmo	Para determinar el número de rondas ejecutadas para el algoritmo base se utiliza la siguiente fórmula: $NFE = Nfi + \sum_{i=1}^{Nr} Nfp_i + Ff$ **NFE:** Número de funciones ejecutadas **Nr:** Número de rondas **NFi:** Número de funciones iniciales (AddRoundKey)	Para determinar el número de rondas ejecutadas para el algoritmo base se utiliza la siguiente fórmula: $NFE = Nfi + \sum_{i=1}^{Nr} Nfp_i + Ff$ **NFE:** Número de funciones ejecutadas **Nr:** Número de rondas **NFi:** Número de funciones iniciales (AddRoundKey, ShiftColumn)	Para determinar el número de rondas ejecutadas para el nuevo algoritmo se utiliza la siguiente fórmula: $NFE = 2 * \left(Nfi + \sum_{i=1}^{Nr} Nfp_i + Ff \right)$ **NFE:** Número de funciones ejecutadas **Nr:** número de rondas **NFi:** Número de funciones iniciales (AddRoundKey, ShiftColumn)

		NFp: Número de funciones parciales (SubByte, ShiftRow, MixColumn, AddRoundKey) **NFf:** Número de funciones finales (SubByte, ShiftRow, AddRoundKey) **Clave de 128 bits** $NFE_{128\ bits} = 1 + \sum_{i=1}^{9} 4 + 3 = 40$ **Clave de 192 bits** $NFE_{192\ bits} = 1 + \sum_{i=1}^{11} 4 + 3 = 48$ **Clave de 256 bits** $NFE_{256\ bits} = 1 + \sum_{i=1}^{13} 4 + 3 = 56$	**NFp:** Número de funciones parciales (SubByte, ShiftRow, ShiftColumn, MixColumn, AddRoundKey) **NFf:** Número de funciones finales (SubByte, ShiftRow, ShiftColumn, AddRoundKey) **Clave de 128 bits** $NFE_{128\ bits} = 2 + \sum_{i=1}^{9} 5 + 4 = 51$ **Clave de 192 bits** $NFE_{192\ bits} = 2 + \sum_{i=1}^{11} 5 + 4 = 61$ **Clave de 256 bits** $NFE_{256\ bits} = 2 + \sum_{i=1}^{13} 5 + 4 = 71$	**NFp:** Número de funciones parciales (SubByte, ShiftRow, ShiftColumn, MixColumn, AddRoundKey) **NFf:** Número de funciones finales (SubByte, ShiftRow, ShiftColumn, AddRoundKey) **Clave de 128 bits** $NFE_{128\ bits} = 2 * \left(2 + \sum_{i=1}^{9} 5_i + 4\right) = 102$ **Clave de 192 bits** $NFE_{192\ bits} = 2 * \left(2 + \sum_{i=1}^{11} 5_i + 4\right) = 122$ **Clave de 256 bits** $NFE_{256\ bits} = 2 * \left(2 + \sum_{i=1}^{13} 5_i + 4\right) = 142$

Realizado por: Méndez Pablo, 2015

4.3.1.1.1. Resultados de análisis de característica

En la Tabla 35-4 se muestra los resultados de la comparación de los indicadores de análisis de características entre el algoritmo AES base implementado en el Prototipo I y el nuevo algoritmo 2NAES implementado en el Prototipo II.

Tabla 35-4 Resultados de comparación de indicadores

No.	Indicadores	Prototipo I (Incluye Algoritmo AES base)			Prototipo II (Incluye nuevo algoritmo criptográfico 2NAES)		
		Longitud de la clave					
		128 bits	192 bits	256 bits	128 bits	192 bits	256 bits
1	No. claves utilizadas	1	1	1	2	2	2
2	No. Rondas	10	12	14	20	24	28
3	No. funciones usadas por el algoritmo	4	4	4	10	10	10
4	No. funciones ejecutadas por el algoritmo	40	48	56	102	122	142

Realizado por: Méndez Pablo, 2015

Una vez obtenidos los valores de los indicadores con cada Prototipo se calculan los valores promedios de los resultados de los indicadores con las 3 claves utilizadas, como se muestra en la Tabla 36-4.

Tabla 36-4 Valores promedio de resultados de los indicadores

No.	Indicadores	Prototipo I	Prototipo II
1	No. claves utilizadas	1	2
2	No. Rondas	12	24
3	No. funciones usadas por el algoritmo	4	10
4	No. funciones ejecutadas por el algoritmo	48	122

Realizado por: Méndez Pablo, 2015

En la Figura 55-4 se muestran los resultados promedios de la comparación realizada por cada uno de los indicadores del 1 al 4.

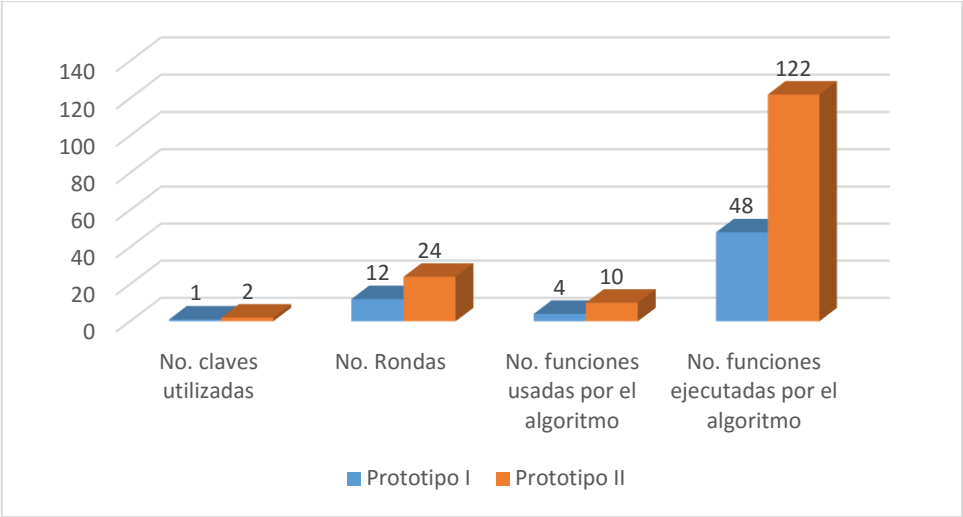


Figura 55-4 Resultados de la comparación de los indicadores del 1 al 4
Realizado por: Méndez Pablo, 2015

4.3.1.2. Criptoanálisis de los mensajes cifrados de los algoritmos

4.3.1.2.1. Ambiente de pruebas

Se compara el algoritmo AES base implementado en el Prototipo I y el nuevo algoritmo 2NAES implementado en el Prototipo II, para los siguientes indicadores de la variable dependiente definida:

- Entropía
- Histograma
- N-Grama
- Autocorrelación
- Fuerza bruta

Se realiza el criptoanálisis utilizando la herramienta Cryptool se comparan los mensajes cifrados con llaves de 128 bits, 192 bits y 256 bits por:

- Algoritmo criptográfico AES base implementado en el Prototipo I
- Algoritmo criptográfico 2NAES implementado en el Prototipo II

Los datos que se utilizaron para las pruebas con el Prototipo I y Prototipo II se muestran en la Tabla 37-4.

Tabla 37-4 Datos para realizar las pruebas

Clave de 128 bits	}yl%OckF x){gI~o
Clave de 192 bits	<E?<E?<E?<E?<E?<E?<E?<E?
Clave de 256 bits	<E?<E?<E?<E?<E?<E?<E?<E?<E?<E?<E
Mensaje	La criptografia y la esteganografia son dos campos que se complementan basados en la seguridad informatica: la primera encripta el mensaje y la segunda oculta el mensaje tras un medio multimedia. Cada una por separado no asegura el secreto, pero si se aplican ambas tecnicas para cifrar y ocultar un mensaje, aumentando el nivel de seguridad

Realizado por: Méndez Pablo, 2015

4.3.1.2.1.1. Mensajes cifrados

4.3.1.2.1.1.1. Clave de 128 bits

En la Figura 56-4 se muestra el mensaje cifrado con el Prototipo I con clave de 128 bits.

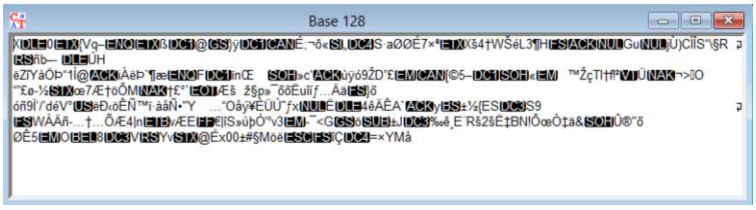


Figura 56-4 Mensaje cifrado con el Prototipo I con clave de 128 bits
Realizado por: Méndez Pablo, 2015

En la Figura 57-4 se muestra el mensaje cifrado con el Prototipo II con clave de 128 bits.

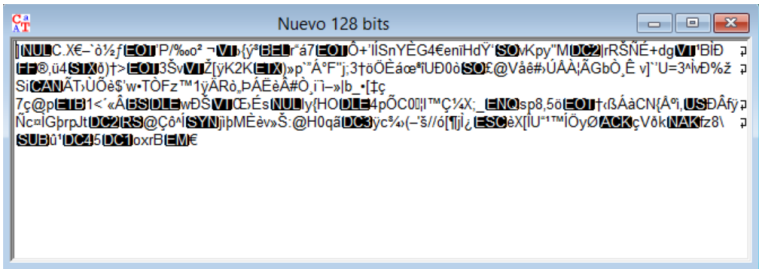


Figura 57-4 Mensaje cifrado con el Prototipo II con clave de 128 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.1.1.2. Clave de 192 bits

En la Figura 58-4 se muestra el mensaje cifrado con el Prototipo I con clave de 192 bits.

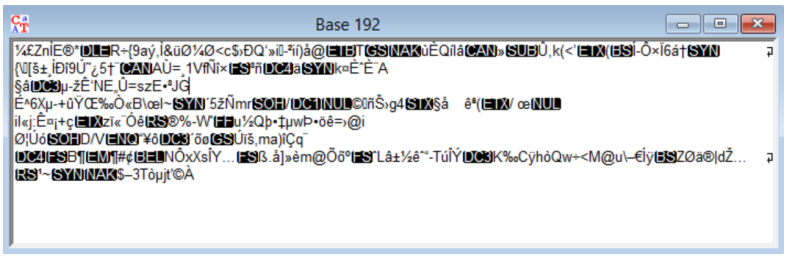


Figura 58-4 Mensaje cifrado con el Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

En la Figura 59-4 se muestra el mensaje cifrado con el Prototipo II con clave de 192 bits.

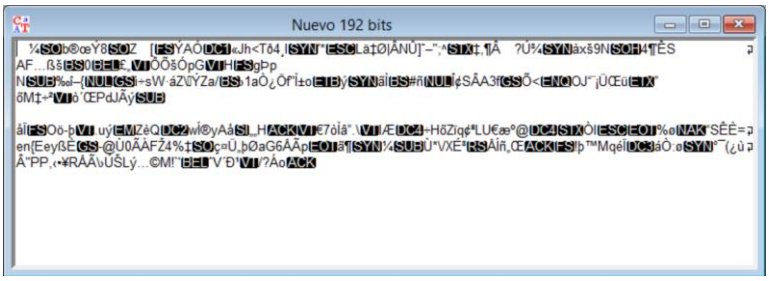


Figura 59-4 Mensaje cifrado con el Prototipo II con clave de 192 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.1.1.3. Clave de 256 bits

En la Figura 60-4 se muestra el mensaje cifrado con el Prototipo I con clave de 256 bits.

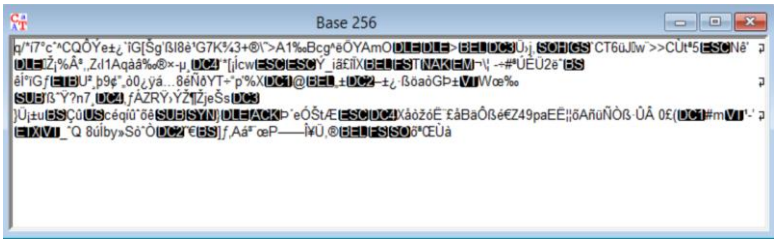


Figura 60-4 Mensaje cifrado con el Prototipo I con clave de 256 bits
Realizado por: Méndez Pablo, 2015

En la Figura 61-4 se muestra el mensaje cifrado con el Prototipo II con clave de 256 bits.

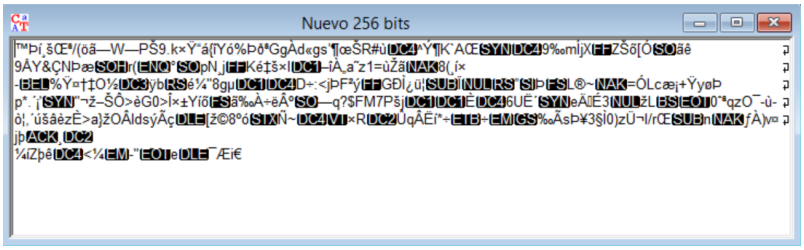


Figura 61-4 Mensaje cifrado con el Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.1.2. Alfabeto

Para las pruebas realizadas de criptoanálisis se utiliza un alfabeto extendido de 98 caracteres que incluyen: letras mayúsculas, letras minúsculas, espacios, números, puntuación, diéresis, para lo cual se configuran las opciones de texto en el programa Cryptool, como se muestra en la Figura 62-4.

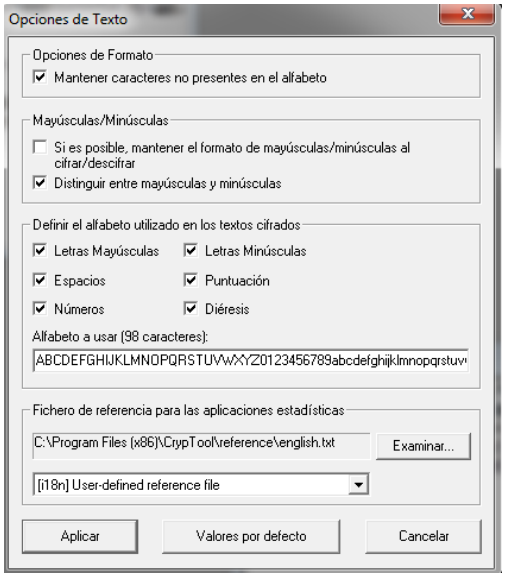


Figura 62-4 Alfabeto utilizado para realizar las pruebas de criptoanálisis
Fuente: Software Cryptool

4.3.1.2.2. Criptoanálisis

Los resultados de las pruebas de criptoanálisis realizado a los indicadores del 5 al 9, los cuales se muestran a continuación:

4.3.1.2.2.1. Indicador 5: Entropía

Se realizan las pruebas de entropía para determinar el nivel de difusión existente en los mensajes cifrados con el Prototipo I y Prototipo II.

4.3.1.2.2.1.1. Clave 128 bits

Se analizan los mensajes cifrados con clave de 128 bits, cuyos resultados se muestran en la Figura 63-4 y Figura 64-4.

Prototipo I

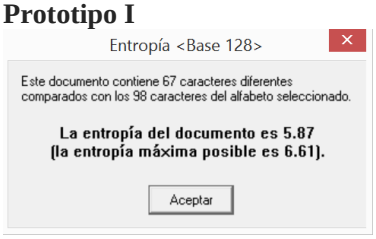


Figura 63-4 Entropía del mensaje cifrado con el Prototipo I con clave de 128 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

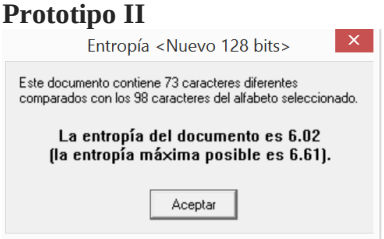


Figura 64-4 Entropía del mensaje cifrado con el Prototipo II con clave de 128 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.1.2. Clave 192 bits

Se analizan los mensajes cifrados con clave de 192 bits, cuyos resultados se muestran en la Figura 65-4 y Figura 66-4.

Prototipo I

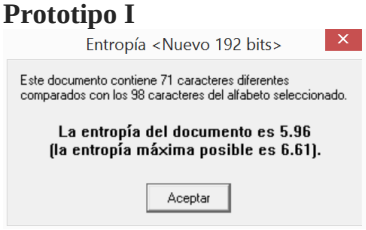


Figura 65-4 Entropía del mensaje cifrado con el Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

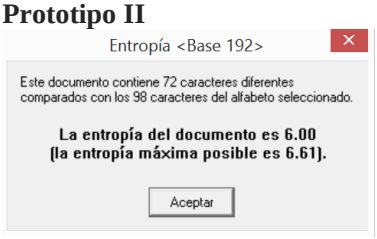


Figura 66-4 Entropía del mensaje cifrado con el Prototipo II con clave de 192 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.1.3. Clave 256 bits

Se analizan los mensajes cifrados con clave de 256 bits, cuyos resultados se muestran en la Figura 67-4 y Figura 68-4.

Prototipo I

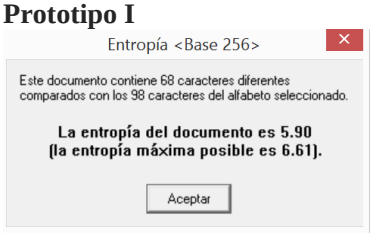


Figura 67-4 Entropía del mensaje cifrado con el Prototipo I con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

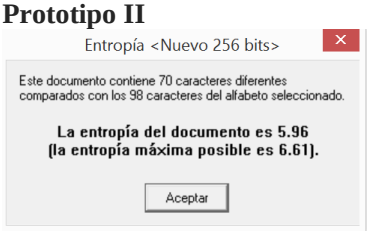


Figura 68-4 Entropía del mensaje cifrado con el Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

En la Tabla 38-4 se muestran el resumen de los caracteres diferentes y el valor de entropía de los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 38-4 Caracteres diferentes y valores de entropía de mensajes cifrados

Clave	Algoritmo	Caracteres diferentes	Entropía máxima	Valor
128 bits	Prototipo I	67	6,61	5,87
	Prototipo II	73	6,61	6,02
192 bits	Prototipo I	71	6,61	5,96
	Prototipo II	72	6,61	6,00
256 bits	Prototipo I	68	6,61	5,90
	Prototipo II	70	6,61	5,96

Realizado por: Méndez Pablo, 2015

Una vez obtenido el valor del Indicador Entropía con cada Prototipo se calculan los valores promedios de los resultados con las 3 claves utilizadas, como se muestra en la Tabla 39-4.

Tabla 39-4 Valor promedio de resultados del Indicador entropía

No.	Indicador	Prototipo I	Prototipo II
5	Entropía	5,91	6,00

Realizado por: Méndez Pablo, 2015

En la Figura 69-4 se muestran los resultados promedios del Indicador 5: Entropía.

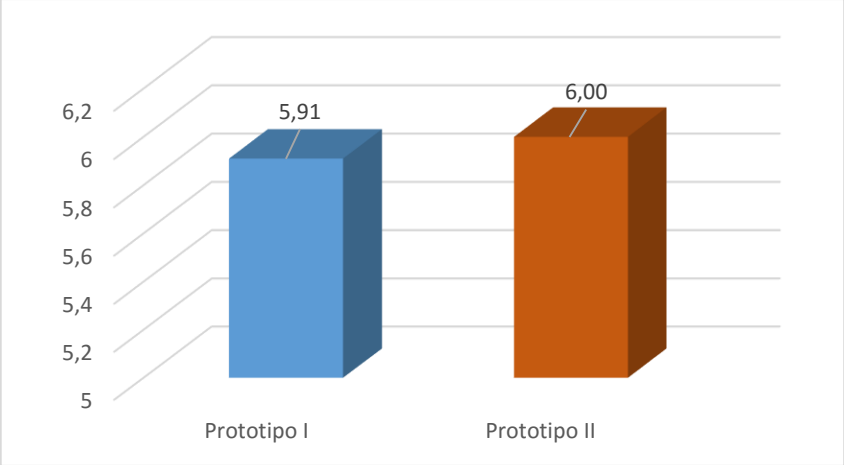


Figura 69-4 Resultados promedios del Indicador 5: Entropía
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.2. Indicador 6: Histograma

Se realizan las pruebas de histograma que relaciona el porcentaje de frecuencia con los valores contenidos en los mensajes cifrados con el Prototipo I y Prototipo II.

4.3.1.2.2.2.1. Clave 128 bits

Se analizan los mensajes cifrados con clave de 128 bits, cuyos resultados se muestran en la Figura 70-4 y Figura 71-4.

Prototipo I

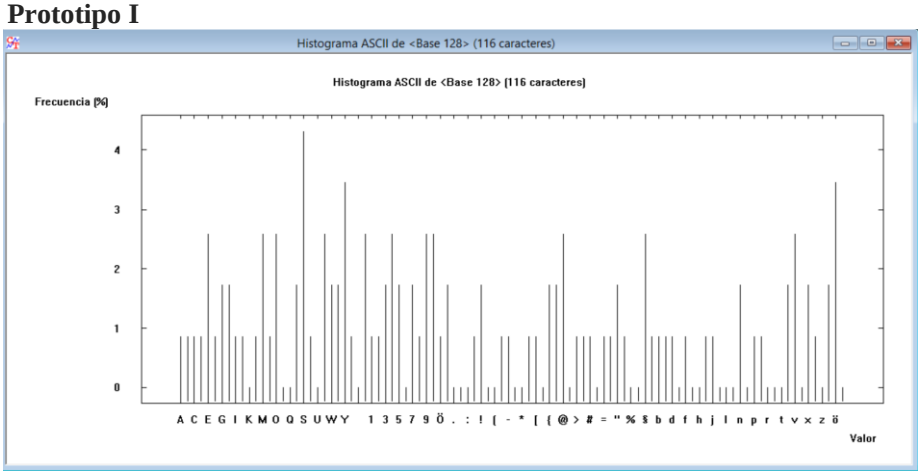


Figura 70-4 Histograma del mensaje cifrado con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

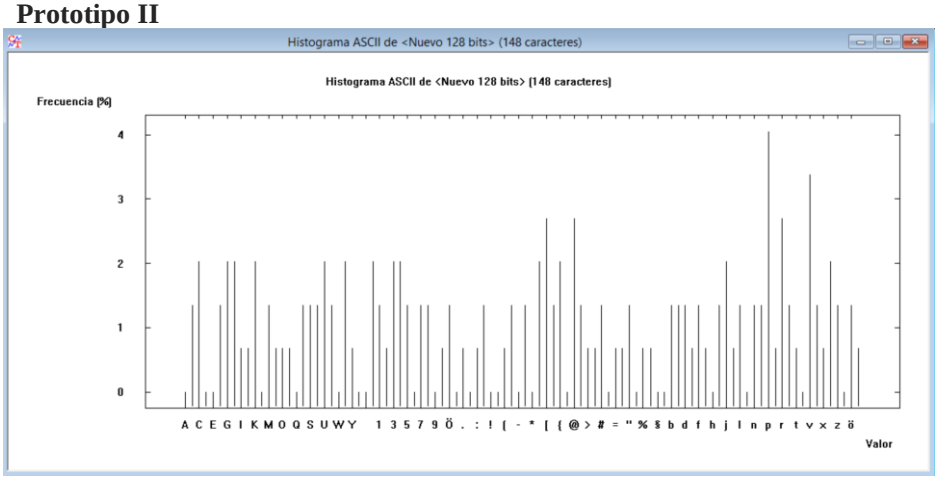


Figura 71-4 Histograma del mensaje cifrado con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.2.2. Clave 192 bits

Se analizan los mensajes cifrados con clave de 192 bits, cuyos resultados se muestran en la Figura 72-4 y Figura 73-4.

Prototipo I

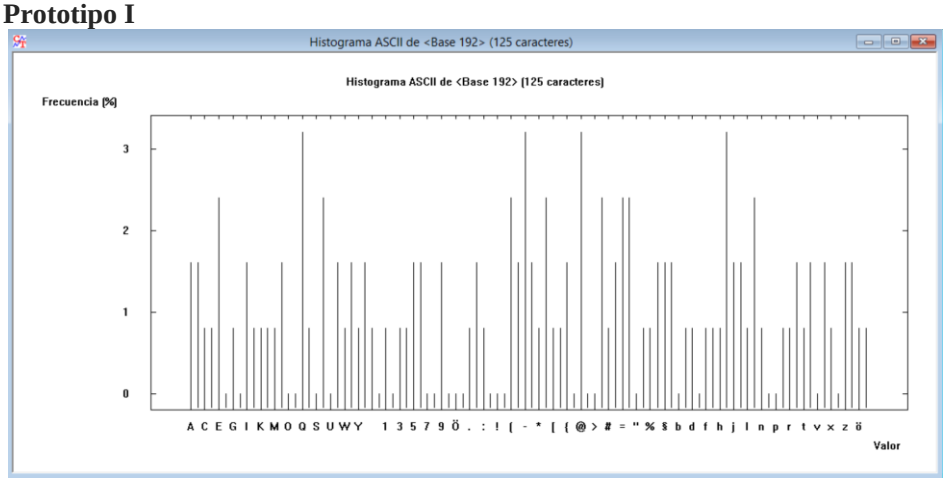


Figura 72-4 Histograma del mensaje cifrado con el Prototipo I con clave de 192 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

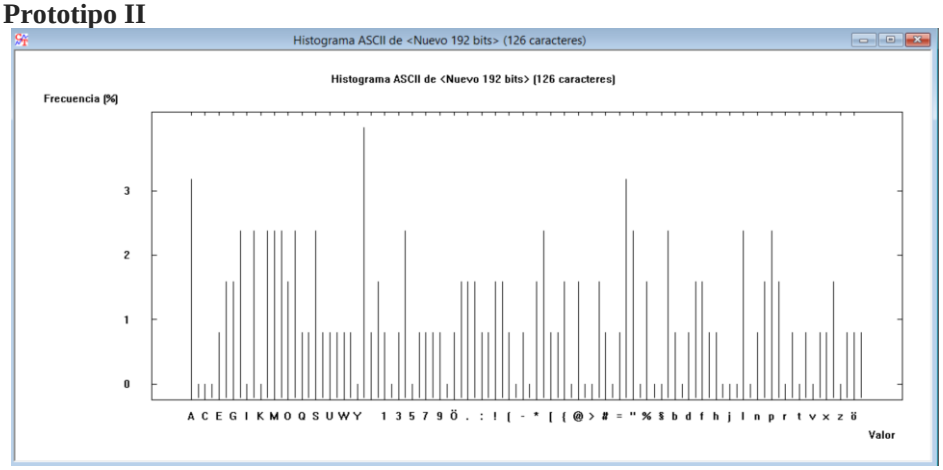


Figura 73-4 Histograma del mensaje cifrado con el Prototipo II con clave de 192 bits
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.2.3. Clave 256 bits

Se analizan los mensajes cifrados con clave de 256 bits, cuyos resultados se muestran en la Figura 74-4 y Figura 75-4.

Prototipo I

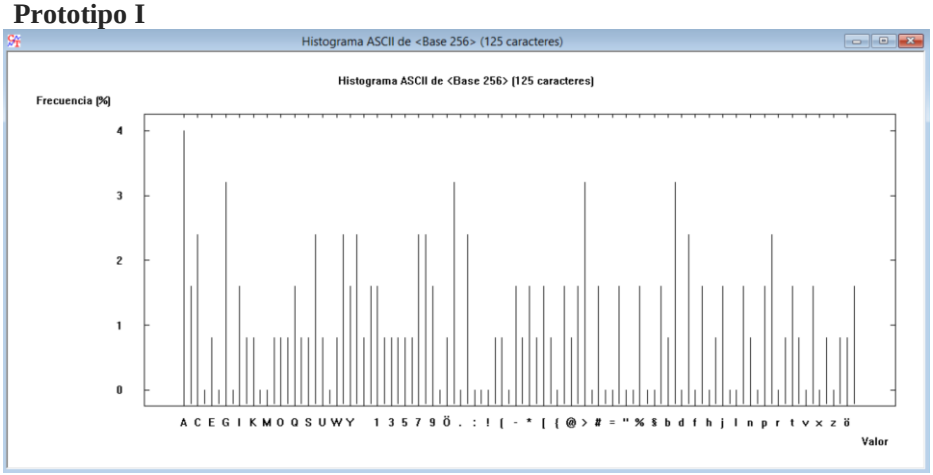


Figura 74-4 Histograma del mensaje cifrado con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

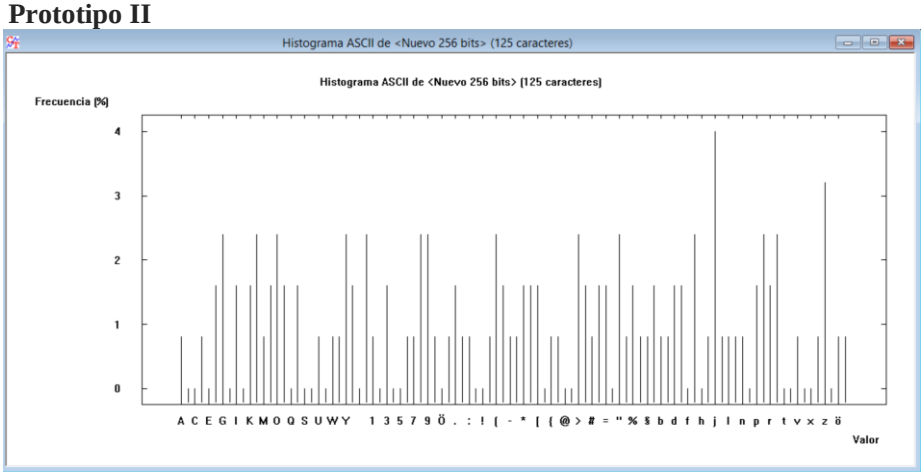


Figura 75-4 Histograma del mensaje cifrado con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

En la Tabla 40-4 se muestran el resumen del histograma en referencia a los caracteres de los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 40-4 Número de caracteres del histograma de los mensajes cifrados

Clave	Algoritmo	Caracteres
128 bits	Prototipo I	116
	Prototipo II	148
192 bits	Prototipo I	125
	Prototipo II	126
256 bits	Prototipo I	125
	Prototipo II	125

Realizado por: Méndez Pablo, 2015

Una vez obtenido el valor del Indicador Histograma en referencia a los caracteres utilizados, con cada Prototipo se calculan los valores promedios con las 3 claves utilizadas, como se muestra en la Tabla 41-4.

Tabla 41-4 Valor promedio de resultados del Indicador histograma

No.	Indicador	Prototipo I	Prototipo II
6	Histograma	122	133

Realizado por: Méndez Pablo, 2015

En la Figura 76-4 se muestran los resultados promedios del Indicador 6: Histograma.

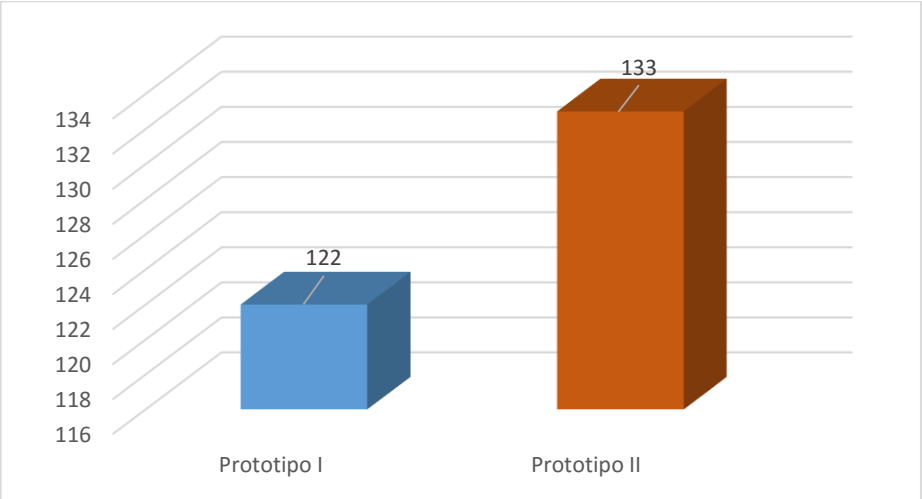


Figura 76-4 Resultados promedios del Indicador 6: Histograma
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.3. Indicador 7: N-Grama

Se realizan las pruebas de N-grama que muestra el número y detalle de secuencia, frecuencia en porcentaje y frecuencia de los n-gramas contenidos en los mensajes cifrados con el Prototipo I y Prototipo II.

4.3.1.2.2.3.1. Clave de 128 bits

Se analizan los mensajes cifrados con clave de 128 bits, cuyos resultados se muestran en la Figura 77-4 y Figura 78-4.

Prototipo I

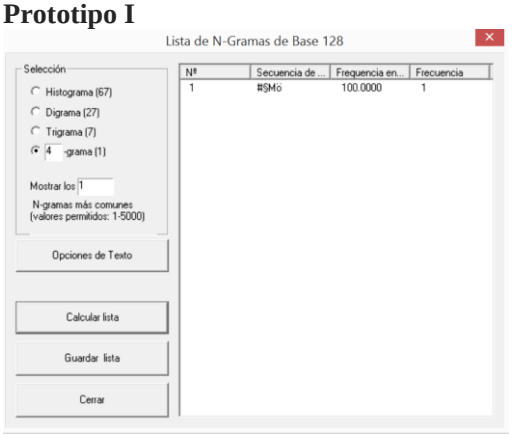


Figura 77-4 N-grama del mensaje cifrado con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

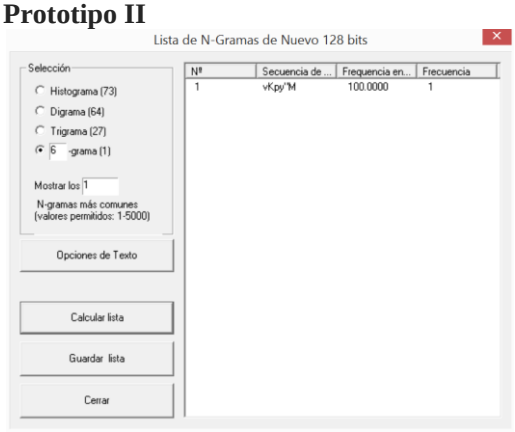


Figura 78-4 N-grama del mensaje cifrado con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.3.2. Clave de 192 bits

Se analizan los mensajes cifrados con clave de 192 bits, cuyos resultados se muestran en la Figura 79-4 y Figura 80-4.

Prototipo I

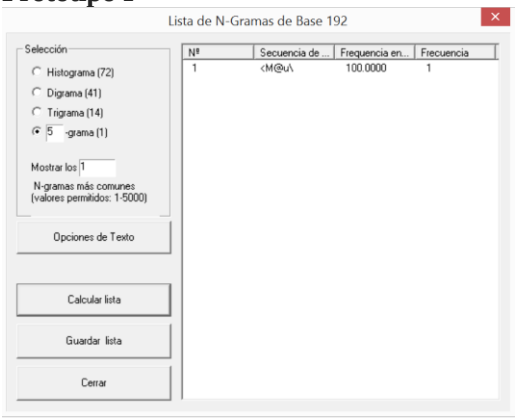


Figura 79-4 N-grama del mensaje cifrado con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

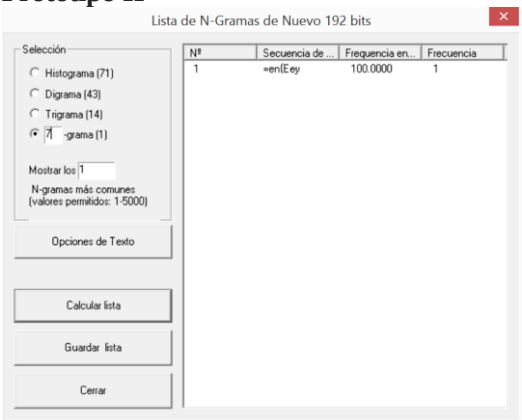


Figura 80-4 N-grama del mensaje cifrado con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.3.3. Clave de 256 bits

Se analizan los mensajes cifrados con clave de 256 bits, cuyos resultados se muestran en la Figura 81-4 y Figura 82-4.

Prototipo I

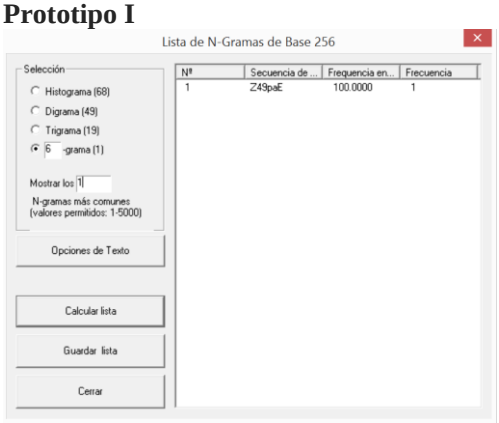


Figura 81-4 N-grama del mensaje cifrado con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

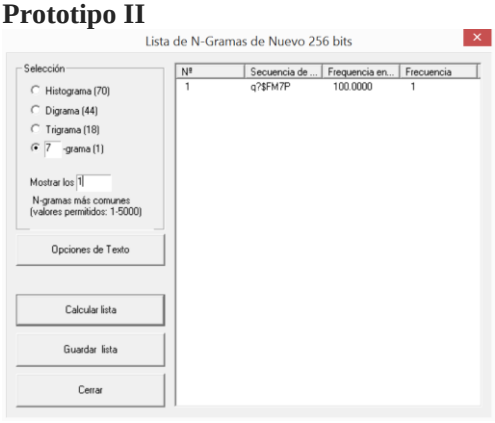


Figura 82-4 N-grama del mensaje cifrado con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

En la Tabla 42-4 se muestran el detalle de los N-gramas de los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 42-4 Detalle de n-gramas de los mensajes cifrados

CLAVE	ALGORITMO	HISTOGRAMA	DIAGRAMA	TRIGRAMA	4-GRAMA	5-GRAMA	6-GRAMA	7-GRAMA
128 bits	Prototipo I	67	27	7	1	0	0	0
	Prototipo II	73	64	27	12	5	1	0
192 bits	Prototipo I	72	41	14	4	1	0	0
	Prototipo II	71	43	14	6	3	2	1
256 bits	Prototipo I	68	49	19	8	4	1	0
	Prototipo II	70	44	18	5	3	2	1

Realizado por: Méndez Pablo, 2015

En la Tabla 43-4 se muestran el resumen de los N-gramas de los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 43-4 Resumen de n-gramas de los mensajes cifrados

Clave	Algoritmo	N-grama (máximo)
128 bits	Prototipo I	4
	Prototipo II	6
192 bits	Prototipo I	5
	Prototipo II	7
256 bits	Prototipo I	6
	Prototipo II	7

Realizado por: Méndez Pablo, 2015

Una vez obtenido el valor del Indicador N-gramas con cada Prototipo se calculan los valores promedios de los resultados con las 3 claves utilizadas, como se muestra en la Tabla 44-4.

Tabla 44-4 Valor promedio de resultados del Indicador n-gramas máximo

No.	Indicador	Prototipo I	Prototipo II
7	N-gramas	5,00	6,66

Realizado por: Méndez Pablo, 2015

En la Figura 83-4 se muestran los resultados promedios del Indicador 7: N-gramas.

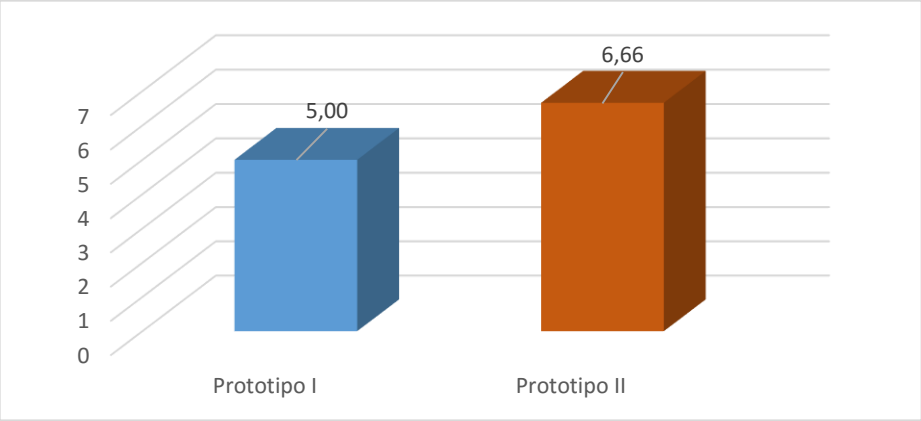


Figura 83-4 Resultados promedios del Indicador 7: N-gramas
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.4. Indicador 8: Autocorrelación

Se realizan las pruebas de correlación que relaciona el número de caracteres que concuerdan con el desplazamiento de los mensajes cifrados con el Prototipo I y Prototipo II.

4.3.1.2.2.4.1. Clave de 128 bits

Se analizan los mensajes cifrados con clave de 128 bits, cuyos resultados se muestran en la Figura 84-4 y Figura 85-4.

Prototipo I

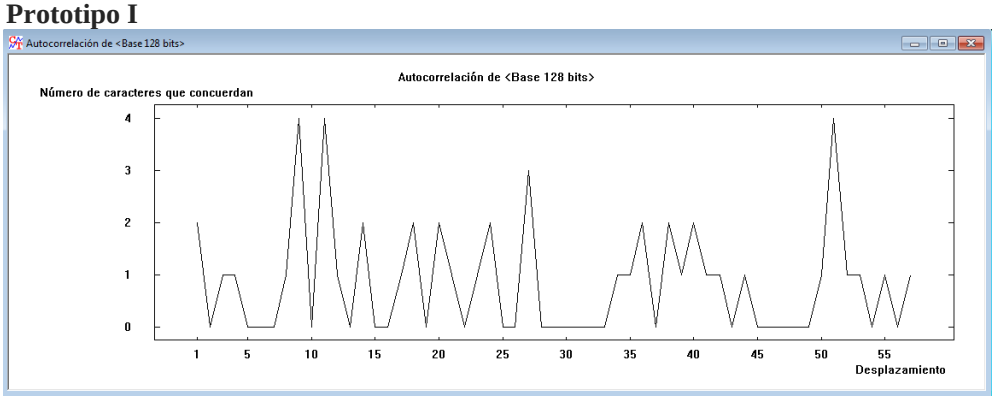


Figura 84-4 Autocorrelación del mensaje cifrado con el Prototipo I con clave de 128 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

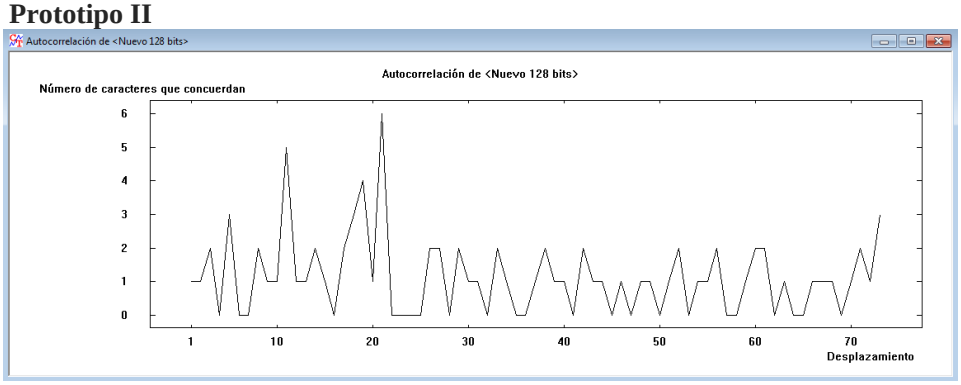


Figura 85-4 Autocorrelación del mensaje cifrado con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.4.2. Clave de 192 bits

Se analizan los mensajes cifrados con clave de 192 bits, cuyos resultados se muestran en la Figura 86-4 y Figura 87-4.

Prototipo I

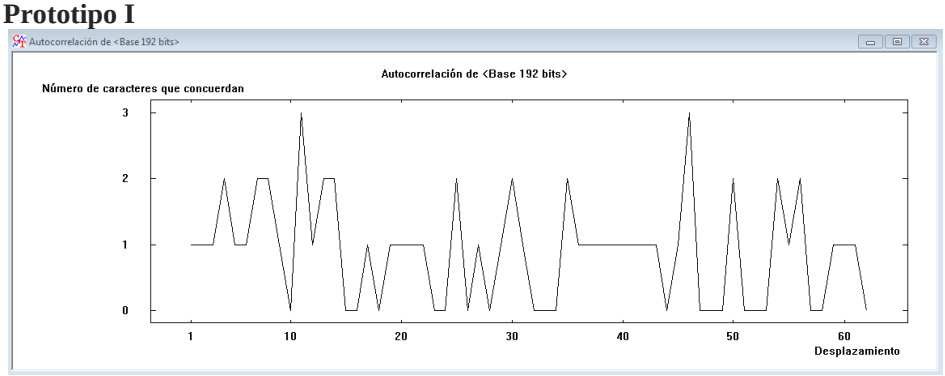


Figura 86-4 Autocorrelación del mensaje cifrado con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

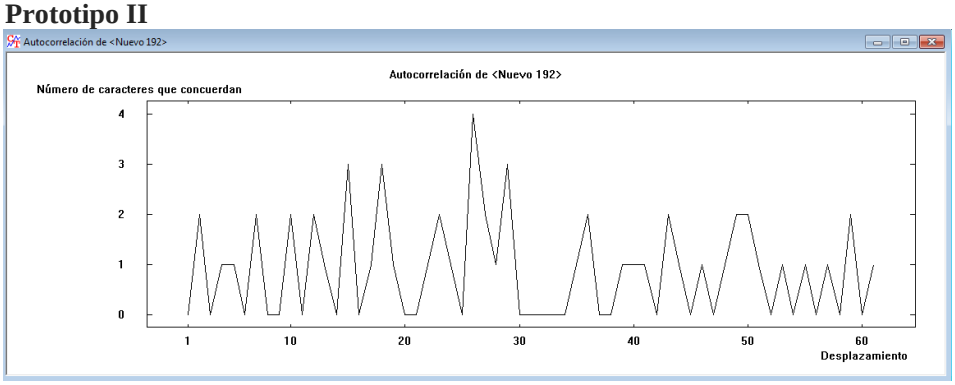


Figura 87-4 Autocorrelación del mensaje cifrado con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.4.3. Clave de 256 bits

Se analizan los mensajes cifrados con clave de 256 bits, cuyos resultados se muestran en la Figura 88-4 y Figura 89-4.

Prototipo I

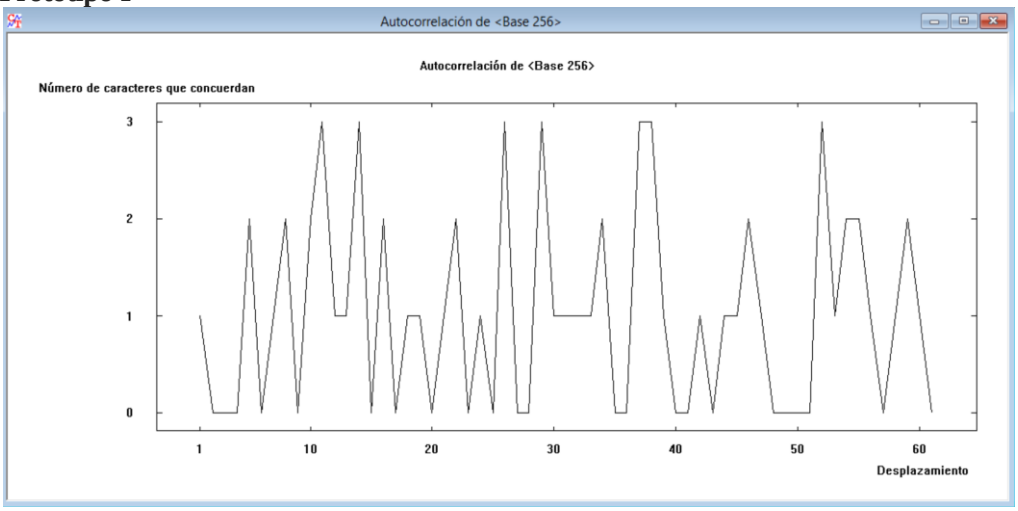


Figura 88-4 Autocorrelación del mensaje cifrado con el Prototipo I con clave de 256 bits
Realizado por: Méndez Pablo, 2015

Prototipo II

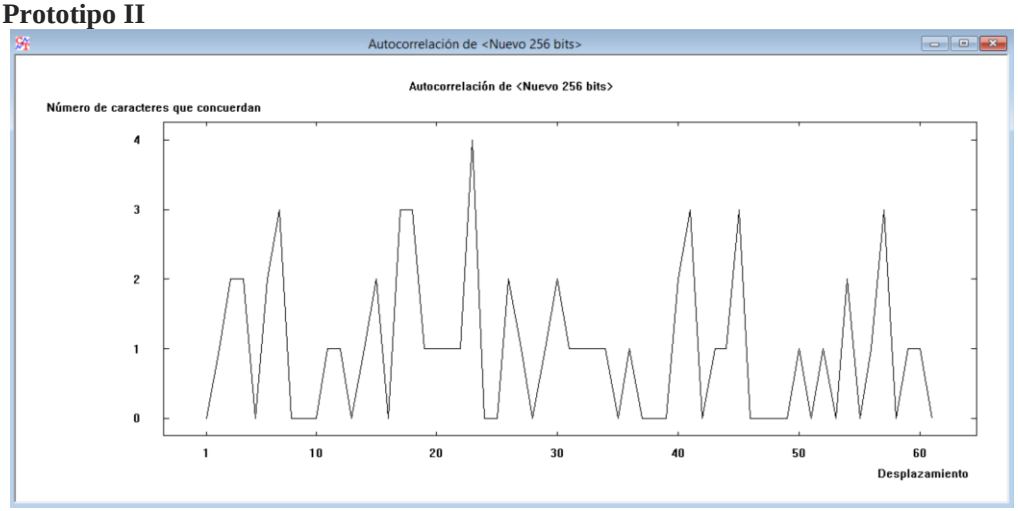


Figura 89-4 Autocorrelación del mensaje cifrado con el Prototipo II con clave de 256 bits
Realizado por: Méndez Pablo, 2015

En la Tabla 45-4 se muestran el resumen de los caracteres que concuerdan en la correlación de los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 45-4 Número de caracteres que concuerdan en la correlación de los mensajes cifrados

Clave	Algoritmo	Caracteres concuerdan
128 bits	Prototipo I	4
	Prototipo II	6
192 bits	Prototipo I	3
	Prototipo II	4
256 bits	Prototipo I	3
	Prototipo II	4

Realizado por: Méndez Pablo, 2015

Una vez obtenido el valor del Indicador Autocorrelación en referencia a los caracteres que concuerdan, con cada Prototipo se calculan los valores promedios de los resultados con las 3 claves utilizadas, como se muestra en la Tabla 46-4.

Tabla 46-4 Valor promedio de resultados del Indicador autocorrelación

No.	Indicador	Prototipo I	Prototipo II
8	Autocorrelación	3,33	4,66

Realizado por: Méndez Pablo, 2015

En la Figura 90-4 se muestran los resultados promedios del Indicador 8: Autocorrelación.

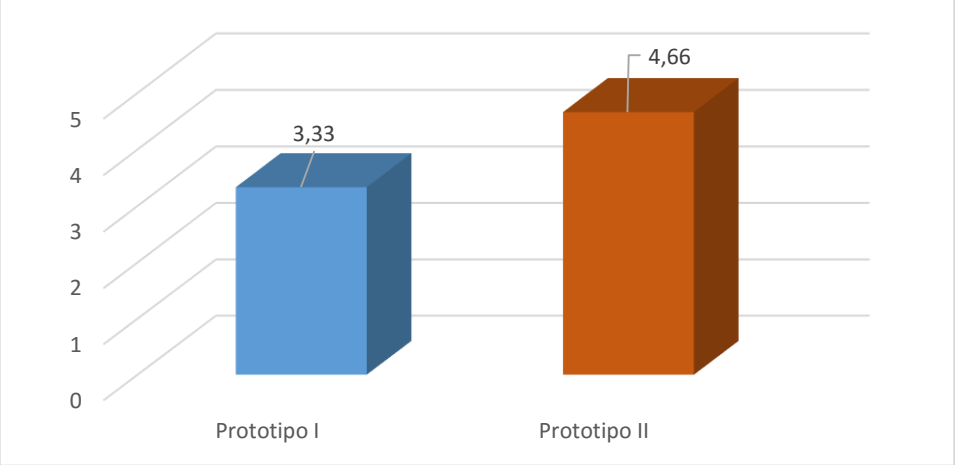


Figura 90-4 Resultados promedios del Indicador 8: Autocorrelación
Realizado por: Méndez Pablo, 2015

4.3.1.2.2.5. Indicador 9: Análisis de fuerza bruta

Se realizan las pruebas de fuerza bruta que se basa en probar todas las combinaciones posibles de la clave, para poder determinar los mensajes cifrados con el Prototipo I y Prototipo II.

En la Figura 91-4 se muestra la interfaz para ingresar el patrón que se desea generar, en caso de no conocerlo se utiliza comodines (representados con *) para que el programa los genere con todas las combinaciones posibles, se debe seleccionar la longitud de la clave que en esta prueba fue de 128 bits.

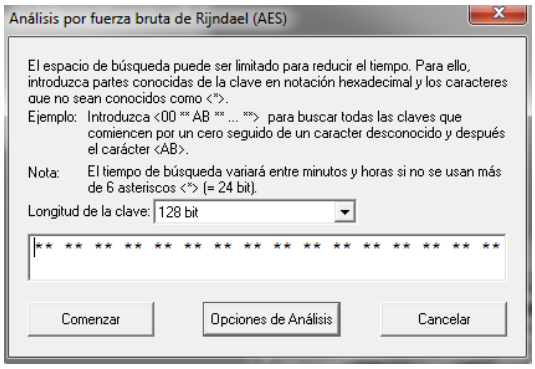


Figura 91-4 Interfaz para ingresar el patrón para determinar la clave por fuerza bruta
Fuente: Software Cryptool

En la Figura 92-4 se muestran las opciones de análisis por fuerza bruta, la cual incluye los análisis de César, Vigenére, XOR, SUMA y el alfabeto que se desea utilizar.

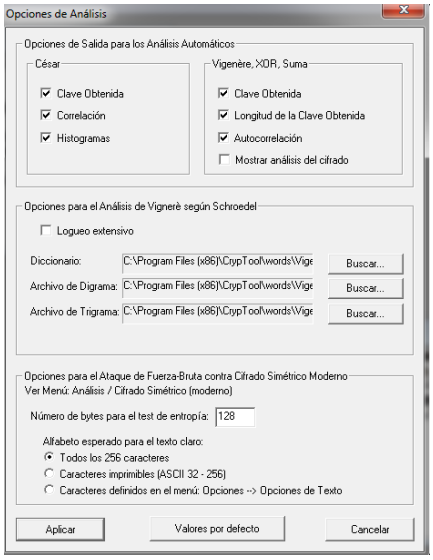


Figura 92-4 Opciones de análisis para determinar la clave por fuerza bruta
Fuente: Software Cryptool

4.3.1.2.2.5.1. Clave de 128 bits

Se realiza la prueba de fuerza bruta con los mensajes cifrados por los prototipos II y II con clave de 128 bits, el tiempo para obtener la clave y descifrado del mensaje con todas las combinaciones posibles se muestran en la Figura 93-4 y Figura 94-4.

Prototipo I

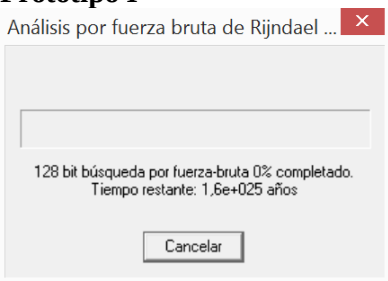


Figura 93-4 Progreso del análisis por fuerza bruta con el Prototipo I con clave de 128 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

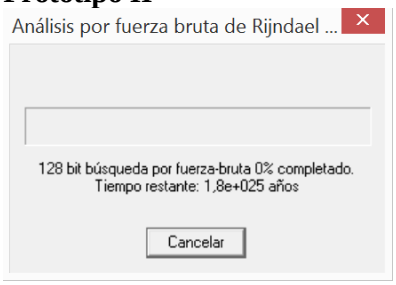


Figura 94-4 Progreso del análisis por fuerza bruta con el Prototipo II con clave de 128 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.5.2. Clave de 192 bits

Se realiza la prueba de fuerza bruta con los mensajes cifrados por los prototipos II y II con clave de 192 bits, el tiempo para obtener la clave y descifrado del mensaje con todas las combinaciones posibles se muestran en la Figura 95-4 y Figura 96-4.

Prototipo I

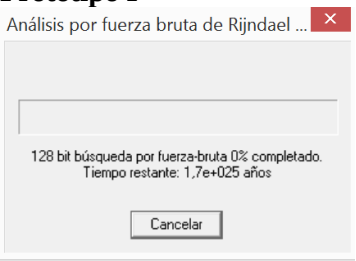


Figura 95-4 Progreso del análisis por fuerza bruta con el Prototipo I con clave de 192 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

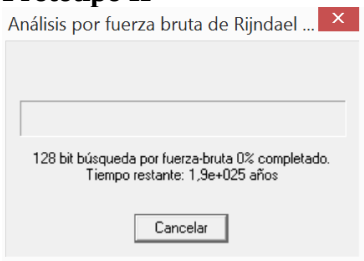


Figura 96-4 Progreso del análisis por fuerza bruta con el Prototipo II con clave de 192 bits

Realizado por: Méndez Pablo, 2015

4.3.1.2.2.5.3. Clave de 256 bits

Se realiza la prueba de fuerza bruta con los mensajes cifrados por los prototipos II y II con clave de 1256 bits, el tiempo para obtener la clave y descifrado del mensaje con todas las combinaciones posibles se muestran en la Figura 97-4 y Figura 98-4.

Prototipo I

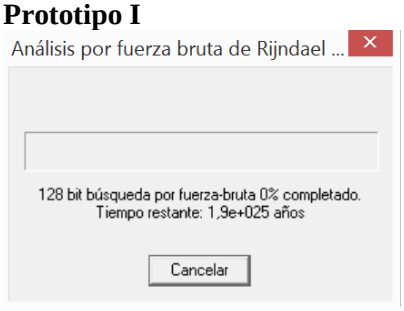


Figura 97-4 Progreso del análisis por fuerza bruta con el Prototipo I con clave de 256 bits

Realizado por: Méndez Pablo, 2015

Prototipo II

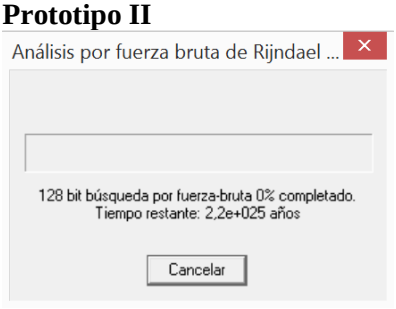


Figura 98-4 Progreso del análisis por fuerza bruta con el Prototipo II con clave de 256 bits

Realizado por: Méndez Pablo, 2015

En la Figura 99-4 se muestra el resultado parcial de las posibles combinaciones generadas hasta el momento de la clave por fuerza bruta, indicando la entropía, el descifrado en hexadecimal, en representación ASCII.

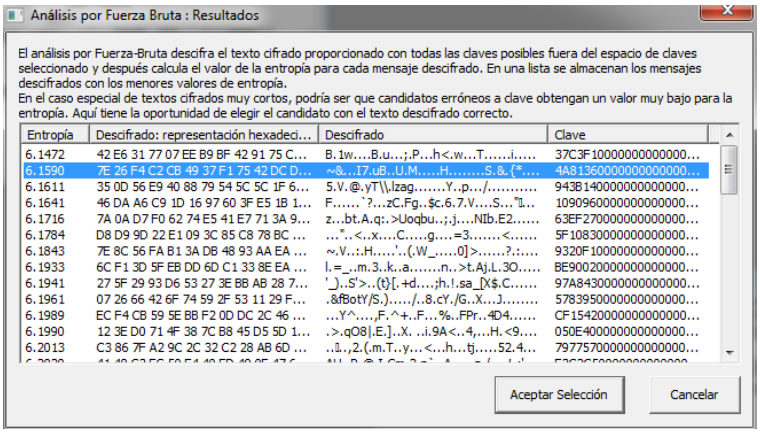


Figura 99-4 Resultado parcial de la ejecución por fuerza bruta

Realizado por: Méndez Pablo, 2015

En la Tabla 47-4 se muestran el resumen del tiempo que tomaría generar la clave necesaria para descifrar los mensajes cifrados por el Prototipo I y Prototipo II con claves de 128 bits, 192 bits y 256 bits.

Tabla 47-4 Número de caracteres del histograma de los mensajes cifrados

Clave	Algoritmo	Tiempo descifrar (años)
128 bits	Prototipo I	$1,6x10^{25}$
	Prototipo II	$1,8x10^{25}$
192 bits	Prototipo I	$1,7x10^{25}$
	Prototipo II	$1,9x10^{25}$
256 bits	Prototipo I	$1,9x10^{25}$
	Prototipo II	$2,2x10^{25}$

Realizado por: Méndez Pablo, 2015

Una vez obtenido el valor del Indicador Fuerza Bruta con cada Prototipo se calculan los valores promedios de los resultados con las 3 claves utilizadas, como se muestra en la Tabla 48-4.

Tabla 48-4 Valor promedio de resultados del Indicador fuerza bruta

No.	Indicador	Prototipo I (años)	Prototipo II (años)
9	Fuerza bruta	$1,73x10^{25}$	$1,97x10^{25}$

Realizado por: Méndez Pablo, 2015

En la Figura 100-4 se muestran los resultados promedios del Indicador 9: Fuerza bruta.

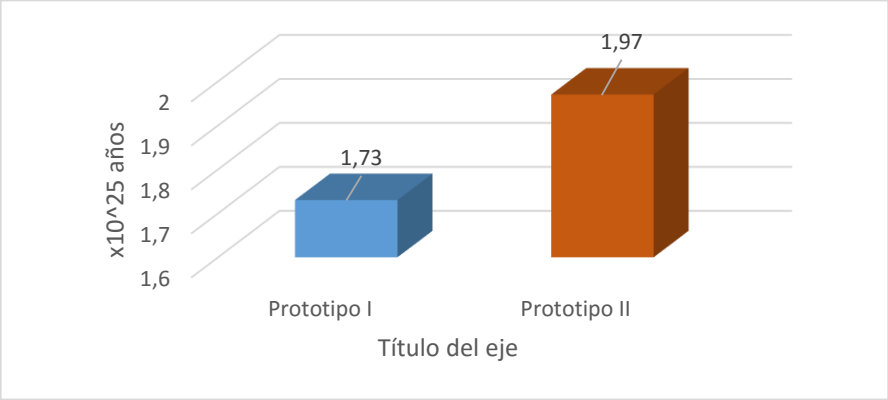


Figura 100-4 Resultados promedios del Indicador 9: Fuerza bruta

Realizado por: Méndez Pablo, 2015

4.3.2. Escalas de calificación

Para realizar la comparación de los resultados obtenidos se utilizará la escala de Likert para cada uno de los indicadores.

4.3.2.1. Indicador 1: No. claves utilizadas

Para medir el Indicador 1: No de claves utilizadas, se utilizará la escala mostrada en la Tabla 49-4.

Tabla 49-4 Tabla de escalas para el Indicador 1: No. claves utilizadas

No. claves	Código
>= 3	4
2	3
1	2
0	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación entre el número de claves y la seguridad es directamente proporcional ya que esta es utilizada en las diferentes rondas con subclaves, lo que difumina más el mensaje.

4.3.2.2. Indicador 2: No. Rondas

Para medir el Indicador 2: No. Rondas, se utilizará la escala mostrada en la Tabla 50-4.

Tabla 50-4 Tabla de escalas para el Indicador 2: No. rondas

No. rondas	Código
>= 20	4
15 … 19	3
10 … 14	2
< 10	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación entre el número de rondas y la seguridad es directamente proporcional ya que con una mayor cantidad de rondas se difumina más el mensaje.

4.3.2.3. Indicador 3: No. funciones usadas por el algoritmo

Para medir el Indicador 3: No. funciones usadas por el algoritmo, se utilizará la escala mostrada en la Tabla 51-4.

Tabla 51-4 Tabla de escalas para el Indicador 3: No. funciones usadas por el algoritmo

No. funciones usadas	Código
>= 10	4
7 … 9	3
4 … 6	2
< 4	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación entre el número de funciones usadas por el algoritmo y la seguridad es directamente proporcional ya que con una mayor cantidad de funciones que utiliza el algoritmo se difumina más el mensaje.

4.3.2.4. Indicador 4: No. funciones ejecutadas por el algoritmo

Para medir el Indicador 4: No. funciones ejecutadas por el algoritmo, se utilizará la escala mostrada en la Tabla 52-4.

Tabla 52-4 Tabla de escalas para el Indicador 4: No. funciones ejecutadas por el algoritmo

No. funciones ejecutadas	Código
> =120	4
80 … 119	3
40 … 79	2
< 40	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación entre el número de funciones ejecutadas por el algoritmo y la seguridad es directamente proporcional ya que con una mayor cantidad de funciones ejecutadas en el algoritmo se difumina más el mensaje.

4.3.2.5. Indicador 5: Entropía

Para medir el Indicador 5: Entropía, se utilizará la escala mostrada en la Tabla 53-4.

Tabla 53-4 Tabla de escalas para el Indicador 5: Entropía

Entropía	Código
>= 6,00	4
5,50 … 5,99	3
5,00 … 5,49	2
< 5,00	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación de entropía y la seguridad es directamente proporcional ya que mientras mayor sea el valor de entropía mayor es la seguridad ya que mide el desorden del mensaje.

4.3.2.6. Indicador 6: Histograma

Para medir el Indicador 6: Histograma, se utilizará la escala mostrada en la Tabla 54-4.

Tabla 54-4 Tabla de escalas para el Indicador 6: Histogramas

Número de caracteres	Código
>= 130	4
100 … 129	3
60 … 99	2
< 60	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación de que el número de caracteres y la seguridad es directamente proporcional ya que mientras mayor sea el número de caracteres utilizado en el mensaje mayor es la seguridad.

4.3.2.7. Indicador 7: N-gramas

Para medir el Indicador 7: N-gramas, se utilizará la escala mostrada en la Tabla 55-4.

Tabla 55-4 Tabla de escalas para el Indicador 7: N-gramas

Número máximo de n-gramas	Código
> 6	4
4 … 5	3
2 … 3	2
1	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación de que el número máximo de n-gramas y la seguridad es directamente proporcional ya que mientras mayor sea el número de gramas o secuencias de elementos utilizados en el mensaje mayor es la seguridad.

4.3.2.8. Indicador 8: Autocorrelación

Para medir el Indicador 8: Autocorrelación, se utilizará la escala mostrada en la Tabla 56-4.

Tabla 56-4 Tabla de escalas para el Indicador 8: Autocorrelación

Número de caracteres que concuerdan	Código
> 4,00	4
3,00 … 4,00	3
2,00 … 2,99	2
< 1,99	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación de que el número de caracteres que concuerdan y la seguridad es directamente proporcional ya que mientras mayor sea el número de caracteres que concuerdan es más difícil determinarlos, por lo que el mensaje es más seguro.

4.3.2.9. Indicador 9: Fuerza bruta

Para medir el Indicador 9: Fuerza Bruta, se utilizará la escala mostrada en la Tabla 57-4.

Tabla 57-4 Tabla de escalas para el Indicador 9: Fuerza bruta

Tiempo descifrar (años)	Código
>= $1,9x10^{25}$	4
$1,7x10^{25}$ … $1,89x10^{25}$	3
$1,5x10^{25}$ … $1,69x10^{25}$	2
< $1,5x10^{25}$	1

Realizado por: Méndez Pablo, 2015

La escala definida se basa en que la relación de que el número de caracteres que concuerdan y la seguridad es directamente proporcional ya que mientras mayor sea el número de caracteres que concuerdan es más difícil determinarlos, por lo que el mensaje es más seguro.

4.3.3. Ponderación de indicadores

Los resultados obtenidos en las pruebas para cada Indicador en el punto 4.3.1 son cuantificados con las escalas definidas en el punto 4.3.2.

4.3.3.1. Indicador 1: No. Claves utilizadas

Utilizando los valores promedios del Indicador 1: No. Claves utilizadas, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 58-4.

Tabla 58-4 Códigos del Indicador 1: No. Claves utilizadas

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
1	No. claves utilizadas	1	2	2	3

Realizado por: Méndez Pablo, 2015

En la Figura 101-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

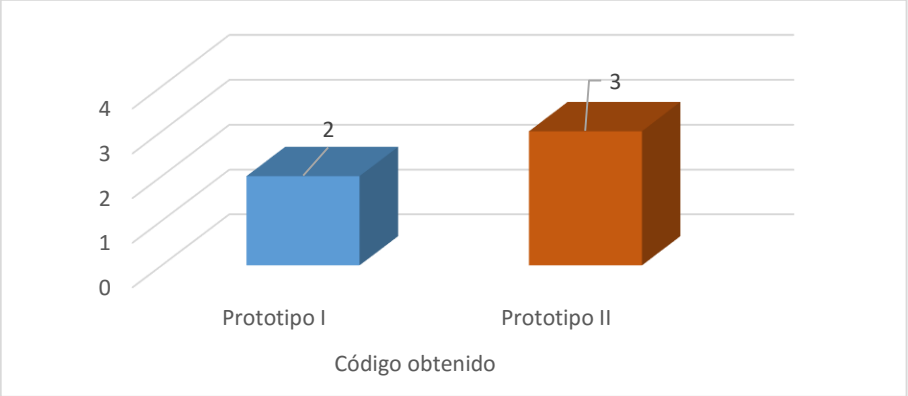


Figura 101-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 1: No. Claves utilizadas

Realizado por: Méndez Pablo, 2015

El Prototipo I utiliza una clave, el Prototipo II repite el proceso 2 veces y utiliza una clave en cada uno, la clave A ingresada por el usuario y la clave B generada a partir de la clave A.

Se concluye que el No. claves utilizadas por el Prototipo II tiene un código de 3 ya que utiliza un número de mayor de claves en comparación con el Prototipo I que tiene un código de 2.

4.3.3.2. Indicador 2: No. Rondas

Utilizando los valores promedios del Indicador 2: No. Rondas, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 59-4.

Tabla 59-4 Códigos del Indicador 2: No. Rondas

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
2	No. rondas	12	24	2	4

Realizado por: Méndez Pablo, 2015

En la Figura 102-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

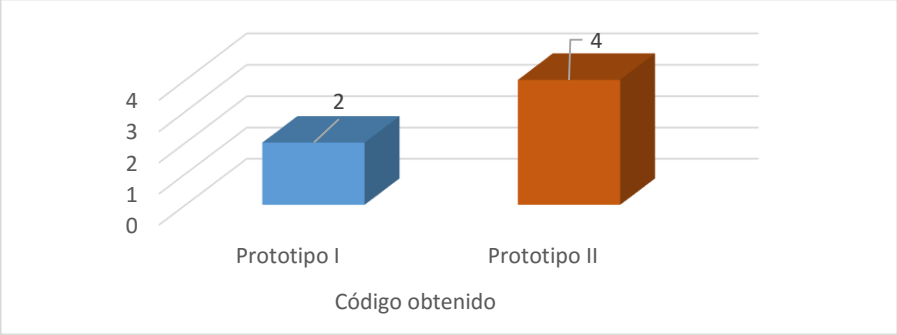


Figura 102-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 2: No. Rondas

Realizado por: Méndez Pablo, 2015

En el Prototipo II se duplican el No. Rondas de acuerdo a las claves utilizadas de 128 bits (20 rondas), 192 bits (24 rondas) y 296 bits (28 rondas) en relación al Prototipo I de acuerdo a las claves utilizadas de 128 bits (10 rondas), 192 bits (12 rondas) y 296 bits (14 rondas).

Se concluye que el No. Rondas por el Prototipo II tiene un código de 4 ya que utiliza un número de mayor de rondas en comparación con el Prototipo I que tiene un código de 2.

4.3.3.3. Indicador 3: No. funciones usadas por el algoritmo

Utilizando los valores promedios del Indicador 3: No. Funciones usadas por el algoritmo, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 60-4.

Tabla 60-4 Códigos del Indicador 3: No. Funciones utilizadas por el algoritmo

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
3	No. funciones usadas por el algoritmo	4	10	2	4

Realizado por: Méndez Pablo, 2015

En la Figura 103-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

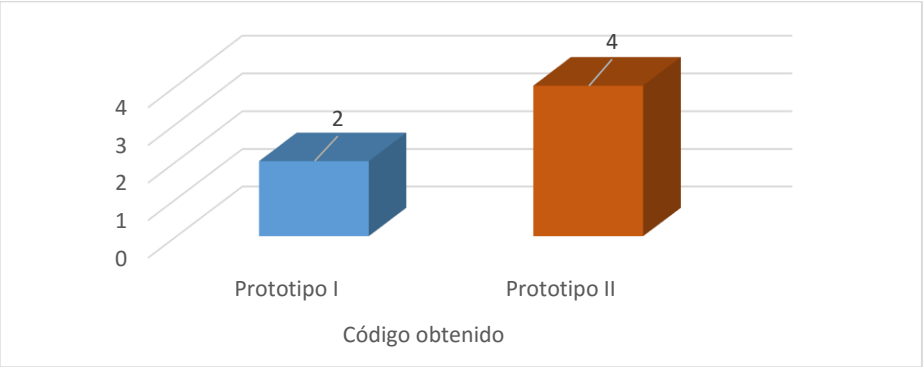


Figura 103-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 3: No. Funciones usadas por el algoritmo

Realizado por: Méndez Pablo, 2015

En el Prototipo I se utilizan 4 funciones definidas y en el Prototipo II se utilizan 5 funciones ya que se incrementa una nueva función (ShiftColumn) y adicionalmente se repite el proceso 2 veces. Se concluye que el No. funciones usadas por el algoritmo con el Prototipo II tiene un código de 4 ya que utiliza un número mayor de funciones en comparación con el Prototipo I que tiene un código de 2.

4.3.3.4. Indicador 4: No. funciones ejecutadas por el algoritmo

Utilizando los valores promedios del Indicador 4: No. Funciones ejecutadas por el algoritmo, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 61-4.

Tabla 61-4 Códigos del Indicador 4: No. Funciones ejecutadas por el algoritmo

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
4	No. funciones ejecutadas por el algoritmo	48	122	2	4

Realizado por: Méndez Pablo, 2015

En la Figura 104-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

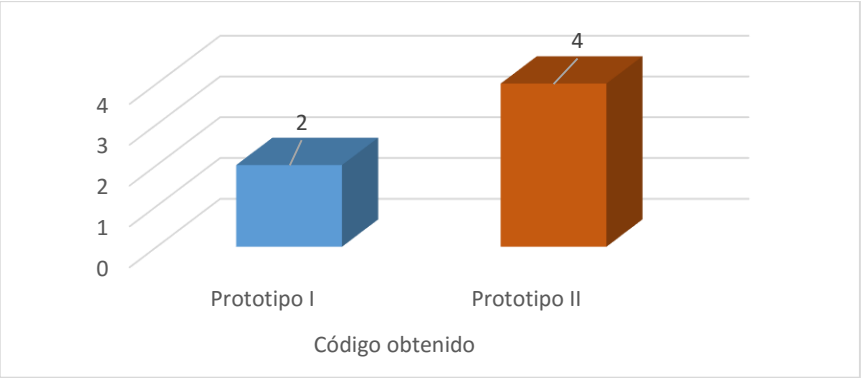


Figura 104-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 4: No. Funciones ejecutadas por el algoritmo

Realizado por: Méndez Pablo, 2015

En el Prototipo I se ejecuta las 4 funciones definidas con el número de rondas establecido para claves de 128 bits (10 rondas), 192 bits (12 rondas), 256 bits (14 rondas). En el Prototipo II se ejecutan las 5 funciones con el número de rondas establecido para claves de 128 bits (20 rondas), 192 bits (24 rondas), 256 bits (28 rondas) y adicionalmente se repite el proceso 2 veces.

Se concluye que el No. funciones ejecutadas por el algoritmo con el Prototipo II tiene un código de 4 ya que ejecuta un número mayor de funciones en comparación con el Prototipo I que tiene un código de 2.

4.3.3.5. Indicador 5: Entropía

Utilizando los valores promedios del Indicador 5: Entropía, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 62-4.

Tabla 62-4 Códigos del Indicador 5: Entropía

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
5	Entropía	5,91	6,00	3	4

Realizado por: Méndez Pablo, 2015

En la Figura 105-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

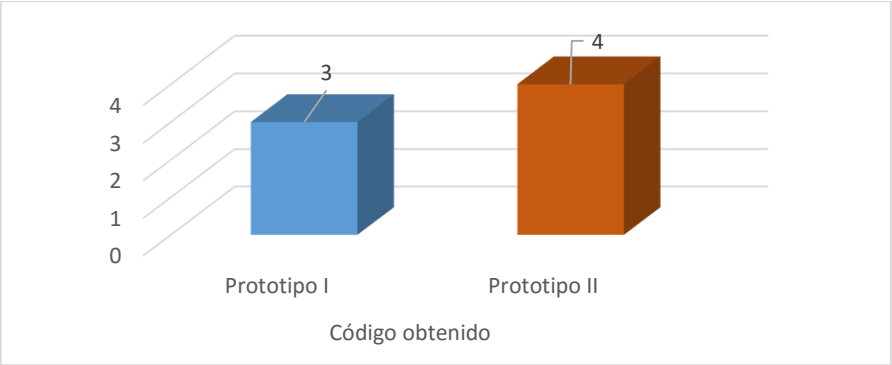


Figura 105-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 5: Entropía

Realizado por: Méndez Pablo, 2015

La entropía máxima alcanzable considerando el alfabeto utilizado es 6.61, para el análisis de los mensajes cifrado con los Prototipos I y II, la relación entre el valor de entropía es directamente proporcional a la seguridad ya que los mensajes son más difusos.

Se concluye que la entropía del mensaje cifrado por el Prototipo II tiene un código de 4 ya que existe mayor entropía en comparación con el Prototipo I que tiene un código de 3.

4.3.3.6. Indicador 6: Histogramas

Utilizando los valores promedios del Indicador 6: Histogramas, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 63-4.

Tabla 63-4 Códigos del Indicador 6: Histogramas

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
6	Histogramas	122	133	3	4

Realizado por: Méndez Pablo, 2015

En la Figura 106-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

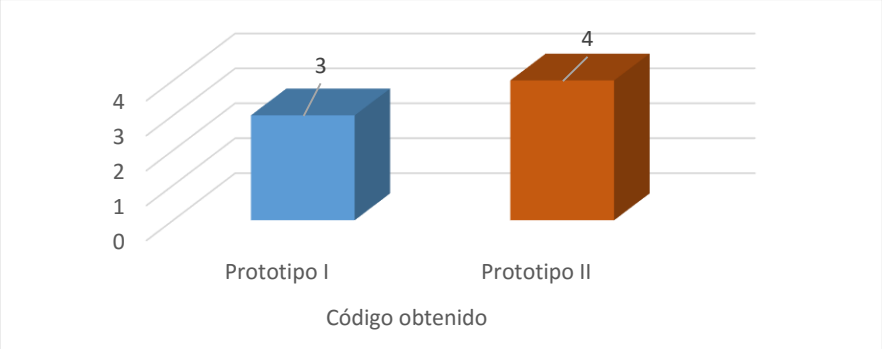


Figura 106-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 6: Histogramas

Realizado por: Méndez Pablo, 2015

Los mensajes cifrados por el Prototipo II contiene una mayor cantidad promedio de caracteres (133) en comparación con los mensajes cifrados por el Prototipo I (122), la relación entre el número de caracteres es directamente proporcional a la seguridad ya que los mensajes son más difusos.

Se concluye que la cantidad de caracteres utilizados que refleja el diagrama de histogramas del mensaje cifrado por el Prototipo II tiene un código de 4 ya que en utiliza un mayor número de caracteres en comparación con el Prototipo I que tiene un código de 3.

4.3.3.7. Indicador 7: N-gramas

Utilizando los valores promedios del Indicador 7: N-gramas, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 64-4.

Tabla 64-4 Códigos del Indicador 7: N-gramas

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
7	N-gramas	5,00	6,66	3	4

Realizado por: Méndez Pablo, 2015

En la Figura 107-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

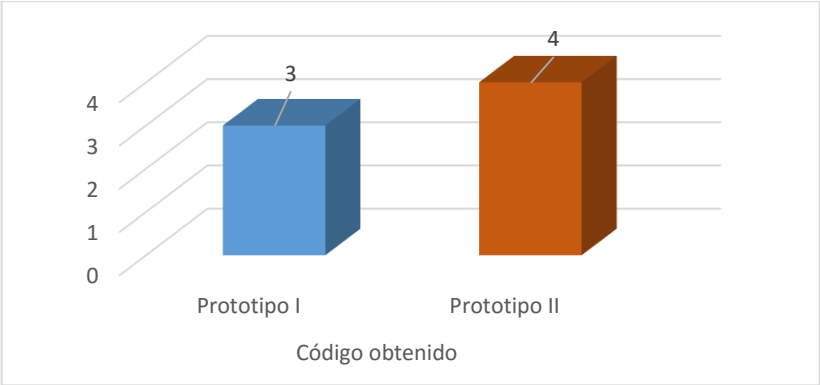


Figura 107-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 7: N-gramas

Realizado por: Méndez Pablo, 2015

Los mensajes cifrados por el Prototipo II contiene un mayor número de n-gramas en comparación con los mensajes cifrados por el Prototipo I, la relación entre el número de caracteres que concuerdan es directamente proporcional a la seguridad ya que los mensajes son más difusos.

Se concluye que la cantidad de gramas utilizados que refleja el diagrama de n-gramas del mensaje cifrado por el Prototipo II tiene un código de 4 ya que en utiliza un mayor número de gramas en comparación con el Prototipo I que tiene un código de 3.

4.3.3.8. Indicador 8: Autocorrelación

Utilizando los valores promedios del Indicador 8: Autocorrelación, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 65-4.

Tabla 65-4 Códigos del Indicador 8: Autocorrelación

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I	Prototipo II	Prototipo I	Prototipo II
8	Autocorrelación	3,33	4,66	3	4

Realizado por: Méndez Pablo, 2015

En la Figura 108-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

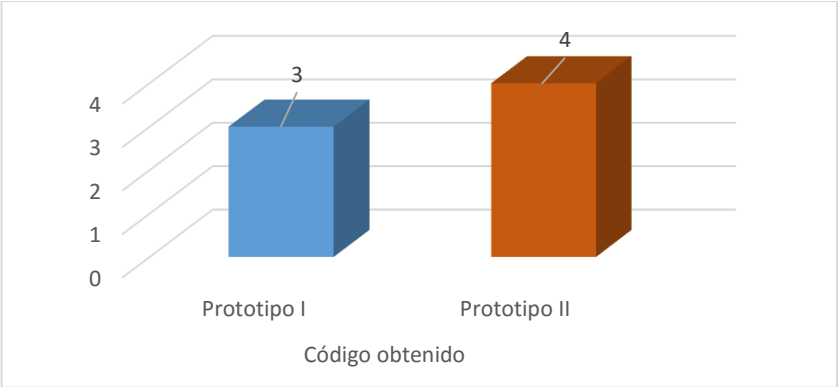


Figura 108-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 8: Autocorrelación

Realizado por: Méndez Pablo, 2015

Los mensajes cifrados por el Prototipo II contiene una mayor cantidad promedio de n-gramas en comparación con los mensajes cifrados por el Prototipo I, la relación entre el número de n-gramas es directamente proporcional a la seguridad ya que los mensajes son más difusos y complejos de determinar las combinaciones posibles.

Se concluye que la cantidad de caracteres que coinciden en el mensaje cifrado por el Prototipo II tiene un código de 4 ya que se incrementa la dificultad de determinarlos en comparación con el Prototipo I que tiene un código de 3.

4.3.3.9. Indicador 9: Fuerza bruta

Utilizando los valores promedios del Indicador 9: Fuerza bruta, con cada prototipo se cuantifica los resultados de acuerdo a la escala definida, con lo que se obtiene los valores mostrados en la Tabla 66-4.

Tabla 66-4 Códigos del Indicador 9: Fuerza Bruta

No.	Indicador	Valor promedio		Código obtenido (de acuerdo a la escala)	
		Prototipo I (años)	Prototipo II (años)	Prototipo I	Prototipo II
9	Fuerza bruta	$1{,}73x10^{25}$	$1{,}97x10^{25}$	3	4

Realizado por: Méndez Pablo, 2015

En la Figura 109-4 se muestran los códigos obtenidos (de acuerdo a la escala) del Indicador.

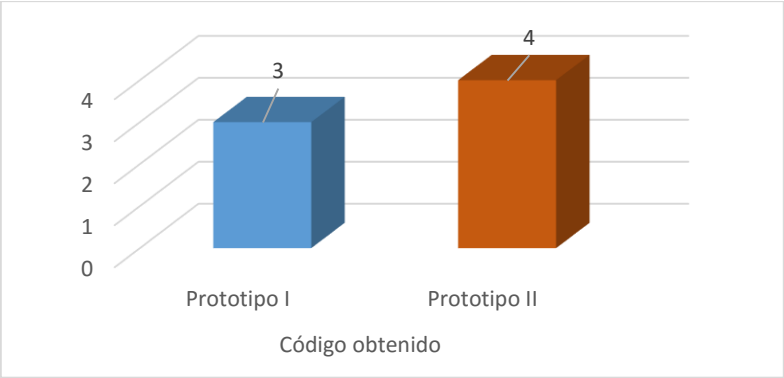


Figura 109-4 Resultados obtenidos (de acuerdo a la escala) del Indicador 9: Fuerza bruta

Realizado por: Méndez Pablo, 2015

Los mensajes cifrados por el Prototipo II requieren de un mayor tiempo para ser descifrados por fuerza bruta con todas las combinaciones posibles en comparación con los mensajes cifrados por el Prototipo I, el nivel de difusión del mensaje es directamente proporcional a la seguridad y al tiempo para descifrarlo por fuerza bruta.

Se concluye que la cantidad de caracteres que coinciden en el mensaje cifrado por el Prototipo II tiene un código de 4 ya que se requiere un mayor tiempo para probar todas las combinaciones posibles por fuerza bruta en comparación con el Prototipo I que tiene un código de 3.

4.3.4. Comprobación de hipótesis

La hipótesis definida en la presente investigacion es “La implementación del nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes mejora la seguridad en comparación con el algoritmo criptográfico base”.

Para la demostración de la hipótesis se utilizará la estadística descriptiva y la estadística inferencial.

4.3.4.1. Estadística descriptiva

Para la comprobación de la hipótesis se utilizará la estadística descriptiva en la que se cuantifican los resultados obtenidos en las pruebas realizadas de cada uno de los indicadores definidos utilizando la escala de Likert, como se muestra en la Tabla 67-4.

Tabla 67-4 Resultados de indicadores

No.	Indicadores	Prototipo I	Prototipo II
1	No. claves utilizadas	2	3
2	No. rondas	2	4
3	No. funciones usadas por el algoritmo	2	4
4	No. funciones ejecutadas por el algoritmo	2	4
5	Entropía	3	4
6	Histograma	3	4
7	N-grama	3	4
8	Autocorrelación	3	4
9	Fuerza bruta	3	4
	TOTAL	**23**	**35**

Realizado por: Méndez Pablo, 2015

En la Figura 110-4 se muestran los resultados de la comparación realizada por cada uno de los indicadores.

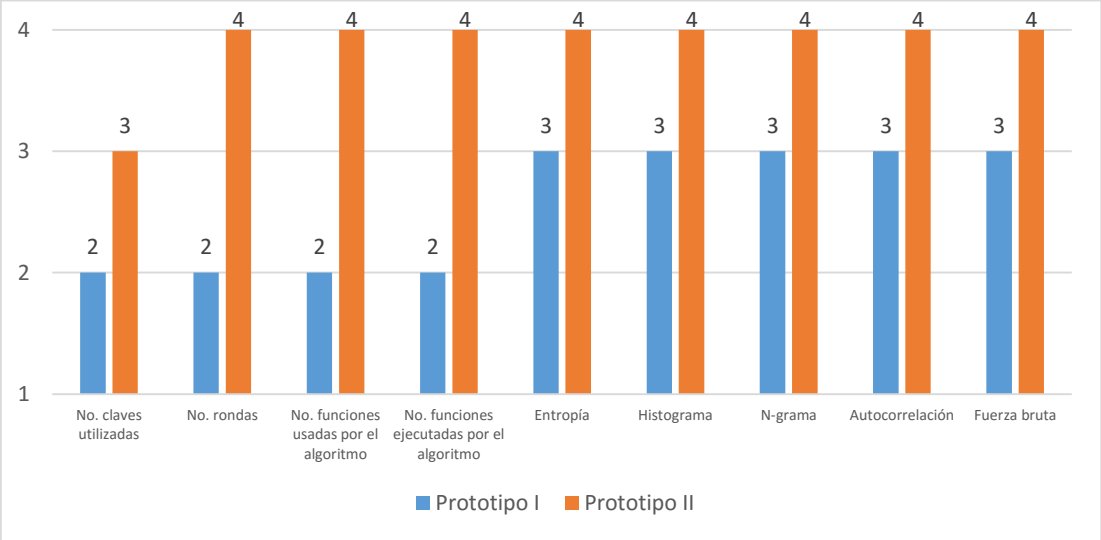


Figura 110-4 Resultados de la comparación por Indicador
Realizado por: Méndez Pablo, 2015

En la Figura 111-4 se muestran los resultados totales de la comparación realizada por cada prototipo.

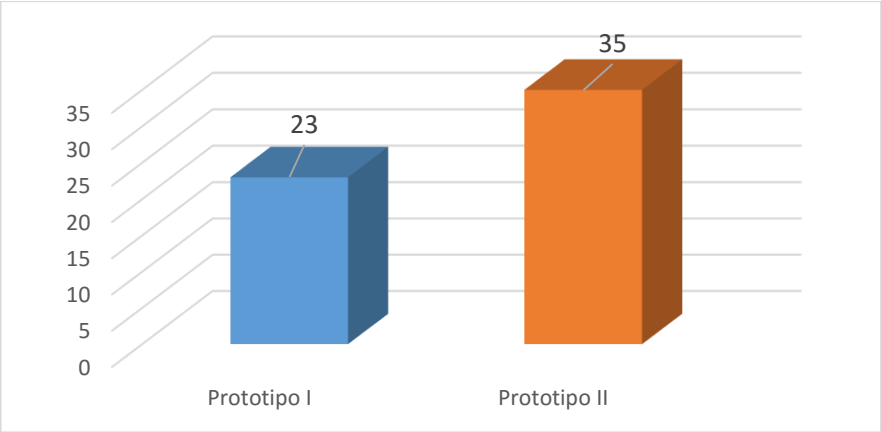


Figura 111-4 Resultados totales de la comparación
Realizado por: Méndez Pablo, 2015

Se concluye que los mensajes cifrados por el Prototipo II son más seguro en un 52% en comparación con los mensajes cifrados por el Prototipo I.

4.3.4.2. Estadística inferencial

Para la comprobación de la hipótesis de investigación se da los siguientes valores a la variable independiente X los siguientes valores:

X = Seguridad

X_1 = Mejora la seguridad

X_2 = No mejora la seguridad

Los mismos en los que se comprobará el impacto en relación a la variable dependiente que son los algoritmos de seguridad implementados en el Prototipo I y Prototipo II.

Para la prueba de hipótesis planteada se utilizó la prueba de chi cuadrado o X^2, que es una prueba no paramétrica a través de la cual se mide la relación entre la variable dependiente e independiente.

Además, se considera la hipótesis nula Ho y la hipótesis de investigación Hi.

- **Hi:** *La implementación del nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes* ***mejora la seguridad*** *en comparación con el algoritmo criptográfico base.*

- **Ho:** *La implementación del nuevo algoritmo criptográfico con la incorporación de la esteganografía en imágenes* ***no mejora la seguridad*** *en comparación con el algoritmo criptográfico base.*

La tabla de contingencia creada para el cálculo de chi cuadrado, se muestra en la Tabla 68-4, en la que se ubican las frecuencias observadas de cada Indicador.

Tabla 68-4 Tabla de contingencia de frecuencias observadas

V. Dependiente / V. Independiente	Indicadores	Prototipo I	Prototipo II	Total
Mejora la seguridad	No. Claves utilizadas	0	3	**3**
	No. rondas	0	4	**4**
	No. funciones usadas por el algoritmo	0	4	**4**
	No. funciones ejecutadas por el algoritmo	0	4	**4**
	Entropía	0	4	**4**
	Histograma	0	4	**4**
	N-grama	0	4	**4**
	Autocorrelación	0	4	**4**
	Fuerza bruta	0	4	**4**
No mejora la seguridad	No. Claves utilizadas	2	0	**2**
	No. rondas	2	0	**2**
	No. funciones usadas por el algoritmo	2	0	**2**
	No. funciones ejecutadas por el algoritmo	2	0	**2**
	Entropía	3	0	**3**
	Histograma	3	0	**3**
	N-grama	3	0	**3**
	Autocorrelación	3	0	**3**
	Fuerza bruta	3	0	**3**
TOTAL		**23**	**35**	**58**

Realizado por: Méndez Pablo, 2015

La tabla de contingencia de frecuencias esperadas son los valores que se esperaría encontrar si las variables no estuvieran relacionadas. Chi cuadrado parte del supuesto de "no relación entre las variables" y se evaluará si es cierto o no, analizando si sus frecuencias observadas son diferentes de lo que pudiera esperarse en caso de ausencia de correlación.

La frecuencia esperada de cada celda, se calcula mediante la siguiente fórmula aplicada a la tabla de frecuencias observadas.

$$fe = \frac{(total\_fil) * (total\_columna)}{N}$$

Donde:

N: Número total de frecuencias observadas.

Aplicando la fórmula a los valores de la Tabla 68-4 se obtiene la tabla de contingencia de valores esperados, como se muestra en la Tabla 69-4.

Tabla 69-4 Tabla de contingencia de frecuencias esperadas

V. Dependiente / V. Independiente	Indicadores	Prototipo I	Prototipo II	Total
Mejora la seguridad	No. Claves utilizadas	1,19	1,81	**3**
	No. rondas	1,59	2,41	**4**
	No. funciones usadas por el algoritmo	1,59	2,41	**4**
	No. funciones ejecutadas por el algoritmo	1,59	2,41	**4**
	Entropía	1,59	2,41	**4**
	Histograma	1,59	2,41	**4**
	N-grama	1,59	2,41	**4**
	Autocorrelación	1,59	2,41	**4**
	Fuerza bruta	1,59	2,41	**4**
No mejora la seguridad	No. Claves utilizadas	0,79	1,21	**2**
	No. rondas	0,79	1,21	**2**
	No. funciones usadas por el algoritmo	0,79	1,21	**2**
	No. funciones ejecutadas por el algoritmo	0,79	1,21	**2**
	Entropía	1,19	1,81	**3**
	Histograma	1,19	1,81	**3**
	N-grama	1,19	1,81	**3**
	Autocorrelación	1,19	1,81	**3**
	Fuerza bruta	1,19	1,81	**3**
TOTAL		**23**	**35**	**58**

Realizado por: Méndez Pablo, 2015

Una vez obtenida la tabla de frecuencias esperadas, se aplica la siguiente fórmula de chi cuadrado.

$$x^2 = \sum \frac{(o - E)^2}{E}$$

Donde:

O: Frecuencia observada en cada celda

E: Frecuencia esperada en cada celda

En la Tabla 70-4 se calcula el valor de X^2

Tabla 70-4 Cálculo de X^2

	Indicadores	O	E	$O-E$	$(O-E)^2$	$\frac{(O-E)^2}{E}$
Prototipo I	Mejora/No. claves utilizadas	0	1,19	-1,19	1,42	1,19
	Mejora/No. rondas	0	1,59	-1,59	2,53	1,59
	Mejora/ No. funciones usadas por el algoritmo	0	1,59	-1,59	2,53	1,59
	Mejora/ No. funciones ejecutadas por el algoritmo	0	1,59	-1,59	2,53	1,59
	Mejora/ Entropía	0	1,59	-1,59	2,53	1,59
	Mejora/ Histograma	0	1,59	-1,59	2,53	1,59
	Mejora/ N-grama	0	1,59	-1,59	2,53	1,59
	Mejora/ Autocorrelación	0	1,59	-1,59	2,53	1,59
	Mejora/ Fuerza bruta	0	1,59	-1,59	2,53	1,59
Prototipo II	Mejora/No. claves utilizadas	3	1,81	1,19	1,42	0,78
	Mejora/No. rondas	4	2,41	1,59	2,53	1,05
	Mejora/ No. funciones usadas por el algoritmo	4	2,41	1,59	2,53	1,05
	Mejora/ No. funciones ejecutadas por el algoritmo	4	2,41	1,59	2,53	1,05
	Mejora/ Entropía	4	2,41	1,59	2,53	1,05
	Mejora/ Histograma	4	2,41	1,59	2,53	1,05
	Mejora/ N-grama	4	2,41	1,59	2,53	1,05
	Mejora/ Autocorrelación	4	2,41	1,59	2,53	1,05
	Mejora/ Fuerza bruta	4	2,41	1,59	2,53	1,05
Prototipo I	No mejoras/ No. Claves utilizadas	2	0,79	1,21	1,46	1,85
	No mejoras/ No. rondas	2	0,79	1,21	1,46	1,85
	No mejoras/ No. funciones usadas por el algoritmo	2	0,79	1,21	1,46	1,85
	No mejoras/ No. funciones ejecutadas por el algoritmo	2	0,79	1,21	1,46	1,85
	No mejoras/ Entropía	3	1,19	1,81	3,28	2,75
	No mejoras/ Histograma	3	1,19	1,81	3,28	2,75
	No mejoras/ N-grama	3	1,19	1,81	3,28	2,75
	No mejoras/ Autocorrelación	3	1,19	1,81	3,28	2,75
	No mejoras/ Fuerza bruta	3	1,19	1,81	3,28	2,75
Prototipo II	No mejoras/ No. Claves utilizadas	0	1,21	-1,21	1,46	1,21
	No mejoras/ No. rondas	0	1,21	-1,21	1,46	1,21
	No mejoras/ No. funciones usadas por el algoritmo	0	1,21	-1,21	1,46	1,21
	No mejoras/ No. funciones ejecutadas por el algoritmo	0	1,21	-1,21	1,46	1,21
	No mejoras/ Entropía	0	1,81	-1,81	3,28	1,81
	No mejoras/ Histograma	0	1,81	-1,81	3,28	1,81
	No mejoras/ N-grama	0	1,81	-1,81	3,28	1,81
	No mejoras/ Autocorrelación	0	1,81	-1,81	3,28	1,81
	No mejoras/ Fuerza bruta	0	1,81	-1,81	3,28	1,81
	X^2					**58,15**

Realizado por: Méndez Pablo, 2015

Interpretación

Para determinar si el valor de X^2 es o no significativo, se debe determinar los grados de libertad mediante la siguiente fórmula.

$$GI = (f - 1)(c - 1)$$

Donde:

f: Número de filas de la tabla de continencia

c: Número de columnas de la tabla de contingencia

Por lo tanto:

$$GI = (18 - 1)(2 - 1) = 17$$

De acuerdo la tabla de distribución X^2 que se muestra en la Tabla 71-4 y eligiendo como nivel de significancia de $\alpha = 0.1\% = 0.001$ para obtener un nivel de confianza del 99.9%, se obtiene como punto crítico de X^2 para 17 grados de libertad $X^2_{Crìtico} = 40.7911$.

Tabla 71-4 Tabla de distribución de X^2

P = Probabilidad de encontrar un valor mayor o igual que el chi cuadrado tabulado, v = Grados de Libertad

v/p	0,001	0,0025	0,005	0,01	0,025	0,05	0,1	0,15	0,2	0,25	0,3	0,35	0,4	0,45	0,5
1	10,8274	9,1404	7,8794	6,6349	5,0239	3,8415	2,7055	2,0722	1,6424	1,3233	1,0742	0,8735	0,7083	0,5707	0,4549
2	13,8150	11,9827	10,5965	9,2104	7,3778	5,9915	4,6052	3,7942	3,2189	2,7726	2,4079	2,0996	1,8326	1,5970	1,3863
3	16,2660	14,3202	12,8381	11,3449	9,3484	7,8147	6,2514	5,3170	4,6416	4,1083	3,6649	3,2831	2,9462	2,6430	2,3660
4	18,4662	16,4238	14,8602	13,2767	11,1433	9,4877	7,7794	6,7449	5,9886	5,3853	4,8784	4,4377	4,0446	3,6871	3,3567
5	20,5147	18,3854	16,7496	15,0863	12,8325	11,0705	9,2363	8,1152	7,2893	6,6257	6,0644	5,5731	5,1319	4,7278	4,3515
6	22,4575	20,2491	18,5475	16,8119	14,4494	12,5916	10,6446	9,4461	8,5581	7,8408	7,2311	6,6948	6,2108	5,7652	5,3481
7	24,3213	22,0402	20,2777	18,4753	16,0128	14,0671	12,0170	10,7479	9,8032	9,0371	8,3834	7,8061	7,2832	6,8000	6,3458
8	26,1239	23,7742	21,9549	20,0902	17,5345	15,5073	13,3616	12,0271	11,0301	10,2189	9,5245	8,9094	8,3505	7,8325	7,3441
9	27,8767	25,4625	23,5893	21,6660	19,0228	16,9190	14,6837	13,2880	12,2421	11,3887	10,6564	10,0060	9,4136	8,8632	8,3428
10	29,5879	27,1119	25,1881	23,2093	20,4832	18,3070	15,9872	14,5339	13,4420	12,5489	11,7807	11,0971	10,4732	9,8922	9,3418
11	31,2635	28,7291	26,7569	24,7250	21,9200	19,6752	17,2750	15,7671	14,6314	13,7007	12,8987	12,1836	11,5298	10,9199	10,3410
12	32,9092	30,3182	28,2997	26,2170	23,3367	21,0261	18,5493	16,9893	15,8120	14,8454	14,0111	13,2661	12,5838	11,9463	11,3403
13	34,5274	31,8830	29,8193	27,6882	24,7356	22,3620	19,8119	18,2020	16,9848	15,9839	15,1187	14,3451	13,6356	12,9717	12,3398
14	36,1239	33,4262	31,3194	29,1412	26,1189	23,6848	21,0641	19,4062	18,1508	17,1169	16,2221	15,4209	14,6853	13,9961	13,3393
15	37,6978	34,9494	32,8015	30,5780	27,4884	24,9958	22,3071	20,6030	19,3107	18,2451	17,3217	16,4940	15,7332	15,0197	14,3389
16	39,2518	36,4555	34,2671	31,9999	28,8453	26,2962	23,5418	21,7931	20,4651	19,3689	18,4179	17,5646	16,7795	16,0425	15,3385
17	40,7911	37,9462	35,7184	33,4087	30,1910	27,5871	24,7690	22,9770	21,6146	20,4887	19,5110	18,6330	17,8244	17,0646	16,3382
18	42,3119	39,4220	37,1564	34,8052	31,5264	28,8693	25,9894	24,1555	22,7595	21,6049	20,6014	19,6993	18,8679	18,0860	17,3379
19	43,8194	40,8847	38,5821	36,1908	32,8523	30,1435	27,2036	25,3289	23,9004	22,7178	21,6891	20,7638	19,9102	19,1069	18,3376
20	45,3142	42,3358	39,9969	37,5663	34,1696	31,4104	28,4120	26,4976	25,0375	23,8277	22,7745	21,8265	20,9514	20,1272	19,3374
21	46,7963	43,7749	41,4009	38,9322	35,4789	32,6706	29,6151	27,6620	26,1711	24,9348	23,8578	22,8876	21,9915	21,1470	20,3372
22	48,2676	45,2041	42,7957	40,2894	36,7807	33,9245	30,8133	28,8224	27,3015	26,0393	24,9390	23,9473	23,0307	22,1663	21,3370
23	49,7276	46,6231	44,1814	41,6383	38,0756	35,1725	32,0069	29,9792	28,4288	27,1413	26,0184	25,0055	24,0689	23,1852	22,3369
24	51,1790	48,0336	45,5584	42,9798	39,3641	36,4150	33,1962	31,1325	29,5533	28,2412	27,0960	26,0625	25,1064	24,2037	23,3367
25	52,6187	49,4351	46,9280	44,3140	40,6465	37,6525	34,3816	32,2825	30,6752	29,3388	28,1719	27,1183	26,1430	25,2218	24,3366
26	54,0511	50,8291	48,2898	45,6416	41,9231	38,8851	35,5632	33,4295	31,7946	30,4346	29,2463	28,1730	27,1789	26,2395	25,3365
27	55,4751	52,2152	49,6450	46,9628	43,1945	40,1133	36,7412	34,5736	32,9117	31,5284	30,3193	29,2266	28,2141	27,2569	26,3363
28	56,8918	53,5939	50,9936	48,2782	44,4608	41,3372	37,9159	35,7150	34,0266	32,6205	31,3909	30,2791	29,2486	28,2740	27,3362
29	58,3006	54,9662	52,3355	49,5878	45,7223	42,5569	39,0875	36,8538	35,1394	33,7109	32,4612	31,3308	30,2825	29,2908	28,3361

Fuente: http://labrad.fisica.edu.uy/docs/tabla_chi_cuadrado.pdf

El valor X^2 calculado $X^2_{Calculado}$ en esta investigación es de 58.15 que es superior al valor de la tabla de distribución de 40.7911, como se muestra en la Figura 112-4.

$$X^2_{Crìtico}(40.7911) < X^2_{Calculado}(58.15)$$

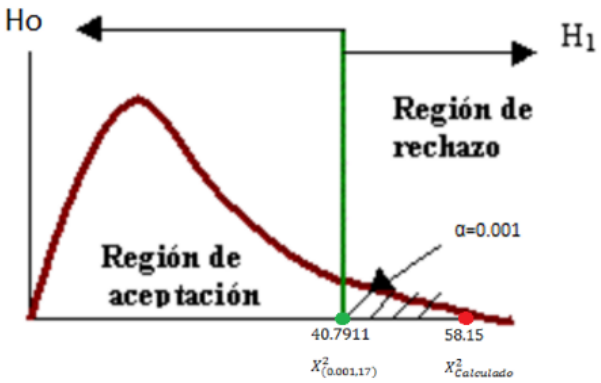


Figura 112-4 Curva de X^2
Realizado por: Méndez Pablo, 2015

Por lo que el valor calculado de X^2 se encuentra en el sector de rechazo de la hipótesis nula H_0, y se acepta la hipótesis de investigación que es significativa, con un nivel de significancia de $\alpha = 0.1\% = 0.001$ para obtener un nivel de confianza del 99.9%,

CONCLUSIONES

- Se analizaron los algoritmos criptográficos simétricos existentes y se seleccionó el algoritmo criptográfico AES, debido a que es el más adecuado por sus ventajas porque permite utilizar claves de 128 bits, 192 bits y 256 bits, tamaños del bloque variables, número de rondas variables, resistente al criptoanálisis diferencial, truncado diferencial, lineal y posee fortaleza contra fuerza bruta, por lo que es más seguro y resistente.

- De las técnicas esteganográficas existentes en la actualidad, se seleccionó la técnica esteganográfica LSB en imágenes debido a que es una de las más utilizadas por sus ventajas porque es sencilla de implementar, rápida, utiliza menos recursos, minimiza la variación en los colores que se crea la incrustación, la distorsión de la imagen se mantiene al mínimo y puede utilizarse en imágenes a color y escala de grises.

- Con la incorporación de la función propuesta ShifColumn en las rondas del algoritmo, repetición del proceso de cifrado y la generación de la clave B en función de la clave A que fueron utilizadas en el proceso del algoritmo 2NAES, el mensaje cifrado se difuminó más en comparación con el algoritmo AES base, demostrando así que es más seguro.

- Con la generación de la clave B a partir de la clave A que fueron utilizadas en el proceso del algoritmo 2NAES, se incrementó las posibilidades de que sea vulnerada por terceros que no conozcan el proceso con el que fue generada.

- Con el cálculo de la complejidad del nuevo algoritmo criptográfico (2NAES) y del algoritmo criptográfico AES base se determinó que tienen el mismo orden de complejidad lineal O(n).

- Con la utilización de la técnica esteganográfica LSB no existe variación en el tamaño de la imagen al ocultar el mensaje porque este es distribuido en el bit menos significativo de los componentes RGB de cada pixel. Además, la distorsión de la imagen que oculta el mensaje se mantiene al mínimo por lo que a simple vista no existe diferencia.

- Para determinar las diferencias existentes en los pixeles de las imágenes original y esteganografiada se utilizó el software Guiffy Image Diff con el que fueron comparados pixel a pixel y como resultados se mostraron los que han sido modificados.

- Para determinar las diferencias existentes en los códigos hexadecimales entre la imagen original y la imagen esteganografiada se utilizó el software FlexHEX y como resultado se mostraron los códigos hexadecimales que han sido modificados.

- El Prototipo II en relación al análisis de las características del nuevo algoritmo criptográfico utiliza un mayor número de claves, mayor número de rondas, mayor cantidad de funciones usadas y ejecutadas en comparación con el Prototipo I.

- Los mensajes cifrados con el Prototipo II poseen mayor entropía, utilizan mayor cantidad de caracteres y gramas, contienen mayor cantidad de caracteres que coinciden y se requiere un mayor tiempo para obtener la clave y con ella el mensaje original utilizando fuerza bruta en comparación a los mensajes cifrados con el Prototipo I.

- Las pruebas de criptoanálisis: entropía, histogramas, n-gramas, autocorrelación y fuerza bruta permitieron determinar que las modificaciones propuestas que posee el algoritmo 2NAES presentan mayor seguridad que las del algoritmo AES base.

- Con la incorporación de la criptografía con la esteganografía en imágenes se incrementa la seguridad de la información porque la misma se encuentra cifrada y oculta dentro de estos archivos multimedia.

- Con el manejo de la estadística descriptiva, utilizando escalas de Likert se demuestra que los mensajes cifrados con el Prototipo II son más seguros en comparación con los mensajes cifrados con el Prototipo I en un 52%.

- Con la utilización de la estadística inferencial con chi cuadrado y un nivel de significancia α=0,001, de acuerdo la tabla de distribución se obtiene que X^2=40,790 y el valor calculado es de $X^{2=}$58,15 que es superior al valor de la tabla de distribución, por lo que el valor calculado de X^2 se encuentra en el sector de rechazo de la hipótesis nula H_0 y resulta estadísticamente significativa aceptando la hipótesis de investigación.

RECOMENDACIONES

- La utilización del algoritmo criptográfico AES como base de otros estudios, debido a que utiliza tamaños de bloque variables, número de rondas variable, resistente a criptoanálisis diferencial, truncado diferencial, lineal y es seguro contra ataques de fuerza bruta.

- El uso de la técnica esteganográfica LSB en imágenes ya que es la más utilizadas debido a que es sencillo de implementar, rápido, utiliza menos recursos, minimiza la variación de los colores, la distorsión de la imagen la mantienen al mínimo, no varía el tamaño de la imagen, puede utilizarse en imágenes a color y escala de grises.

- El análisis de los procesos de los algoritmos criptográficos y esteganográficos con la finalidad de comprenderlos e implementarlos de forma adecuada.

- La definición de los escenarios, ambientes de pruebas e indicadores de las variables dependientes e independientes de forma adecuada con el objetivo de que la ejecución y resultados de las pruebas realizadas sean confiables.

- La comparación de los prototipos en situaciones similares con el objetivo de verificar que las modificaciones realizadas permiten obtener valores significativos de mejora.

- La determinación del cálculo de la complejidad de los algoritmos criptográficos desarrollados para medir la eficiencia del mismo.

- La utilización del software Guiffy Image Diff porque permite identificar los cambios realizados en las imágenes comparándolos píxel a pixel.

- El uso del software FlexHEX porque permite identificar los cambios realizados en las imágenes en el código hexadecimal.

- El manejo de herramientas de criptoanálisis para la evaluación y comparación de mensajes cifrados mediante análisis de entropía, histogramas, n-gramas, autocorrelación, fuerza bruta, entre otros para medir su difusión.

- El uso de las escalas de Likert para medir los indicadores de la variable dependiente con el objetivo de unificar valores para su comparación.

- La utilización del Prototipo II que incluye el nuevo algoritmo criptográfico desarrollado (2NAES) y la técnica LSB, cuando la información sea altamente confidencial debido a que los mensajes cifrados son más seguro en comparación con los mensajes cifrados con el algoritmo AES base.

- El manejo de la estadística descriptiva e inferencial para que la demostración de la hipótesis tenga fundamento científico y fiable de los valores obtenidos en la investigación.

Para trabajos de investigación futuros se recomienda considerar las siguientes sugerencias:

- La utilización de diferentes tamaños de bloques para las claves de 128 bits, 192 bits, 256 bits.

- La implementación de claves de longitud de 512 bits con diferentes tamaños de bloque.

- La modificación en: el orden de ejecución de las funciones definidas por el algoritmo, número de rondas que son utilizadas en el algoritmo, de la matriz S-BOX para generar nuevas combinaciones.

- Debido a que las funciones Mixcolum y ShiftRow proporcionan la principal fuente de difusión en el cifrado se pueden proponer modificaciones a estas funciones para crear nuevos algoritmos criptográficos.

- La creación de nuevas funciones e incorporarlas en las diferentes rondas del algoritmo para difuminar el mensaje.

- La modificación de las funciones existentes de los algoritmos criptográficos con la finalidad de mejorarlas.

- La utilización de otras técnicas esteganográficas que sean incorporadas a la criptografía para mejorar la seguridad.

- La creación de nuevas técnicas esteganográficas que permitan ocultar información a las que se incorpore algoritmos criptográficos existentes o nuevos para mejorar la seguridad.

- El manejo de indicadores de calidad de imagen PSNR (Peak Signal to Noise Ratio) y MSE (Mean Square Error).

- El uso de la esteganografía en imágenes para ocultar otros archivos multimedia como imágenes, archivos, documentos, etc., con nuevas propuestas de algoritmos criptográficos.

BIBLIOGRAFÍA

ALAÍZ, C., FERNÁNDEZ, A., & RODRÍGUEZ, I. (2011). *Ingeniería Informática*. Obtenido de http://arantxa.ii.uam.es/~aa/practicas/recursos/ordenesComplejidad.html

ANGEL, J. (2005). *AES - Advanced Encryption Standard.* Obtenido de http://computacion.cs.cinvestav.mx/~jjangel/aes/AES_v2005_jjaa.pdf

BLANCO, R. (2010). *Unidad Profesional Interdisciplinaria de Ingeniería y Ciencias Sociales y Administrativas.* Obtenido de http://itzamna.bnct.ipn.mx:8080/dspace/bitstream/123456789/6239/1/IF2.45.pdf

CASTILLO, L., CASTILLO, M., & NÚÑEZ, G. (2013). *CCNS.* Obtenido de http://ccns.jimdo.com/encriptaci%C3%B3n-de-datos/

COMUNIDAD DOCSETOOLS. (2013). *Doc Setools.* Obtenido de http://docsetools.com/articulos-noticias-consejos/article_129171.html

COMUNIDAD ECURED. (2014). *EcuRed.* Obtenido de http://www.ecured.cu/index.php/Criptograf%C3%ADa_asim%C3%A9trica

COMUNIDAD EXPRESIÓN BINARIA. (2011). *Expresión Binaria.* Obtenido de http://www.expresionbinaria.com/glosario/criptoanalisis/

COMUNIDAD EXPRESIÓN BINARIA. (2012). *Expresión Binaria*. Obtenido de http://www.expresionbinaria.com/el-arte-de-ocultar-informacion-esteganografia/

COMUNIDAD GITS INFORMÁTICA. (2003). *Gits Informatica.* Obtenido de http://www.gitsinformatica.com/descargas/Esteganografia.doc

COMUNIDAD GUIFFY SOFTWARE. (2014). *Guiffy Software*. Obtenido de http://www.guiffy.com/Image-Diff-Tool.html

COMUNIDAD MONOGRAFÍAS. (2014). *Monografías*. Obtenido de http://www.monografias.com/trabajos27/complejidad-algoritmica/complejidad-algoritmica.shtml#ixzz3hc0D4wQF

COMUNIDAD NETBEANS. (2015). *Netbeans*. Obtenido de https://www.netbeans.org

COMUNIDAD WIKIPEDIA. (2015). *Wikipedia.* Obtenido de https://es.wikipedia.org/wiki/Advanced_Encryption_Standard

COMUNIDAD WIKIPEDIA. (2015). *Wikipedia.* Obtenido de http://es.wikipedia.org/wiki/Data_Encryption_Standard

DE LUZ, S. (2010). *Redes Zone*. Obtenido de http://www.redeszone.net/2010/11/16/criptografia-algoritmos-de-cifrado-de-clave-asimetrica/

DE LUZ, S. (2010). *Redes Zone.* Obtenido de http://www.redeszone.net/2010/11/04/criptografia-algoritmos-de-cifrado-de-clave-simetrica/

DEUTSCHE BANK & COLABORADORES. (2015). *Cryptool.* Obtenido de https://www.cryptool.org/en/cryptool1-en

DÍAZ, N. (2004). *Artemisa.* Obtenido de http://artemisa.unicauca.edu.co/~nediaz/EDDI/cap01.htm

DMK. (2010). *RedInfoCol.* Obtenido de http://www.redinfocol.org/atacando-por-fuerza-bruta-bruteforce-1/

GABA, J., & KUMAR, M. (2013). Implementation of steganography using CES technique. *IEEE Second International Conference on Image Information Processing (ICIIP)* (pp. 395-399). Shimla: IEEE.

GUO, J., & LE, T. (2010). Secret Communication Using JPEG Double Compression. *IEEE Signal Processing Letters*, 879-882.

GUTIERREZ, P. (2013). *Genbeta: dev*. Obtenido de http://www.genbetadev.com/seguridad-informatica/tipos-de-criptografia-simetrica-asimetrica-e-hibrida

INV SOFTWORKS LLC. (2015). *FlexHEX*. Obtenido de http://www.flexhex.com/

KUMAR, M., HEMRAJANI, N., & KISHORE, A. (2013). Security Improvisation in Image Steganography using DES. *Advance Computing Conference (IACC)* (pp. 1094-1099). Ghaziabad: IEEE.

LÓPEZ, M. (2012). *UnoCero.* Obtenido de http://www.unocero.com/2012/11/28/esteganografia-para-cifrar-mensajes-en-imagenes/

MASON, L. (2014). *eHow en Español.* Obtenido de http://www.ehowenespanol.com/ventajas-desventajas-criptografia-clave-simetrica-info_276624/

MATHUR, M., & KESARWANI, A. (2013). Comparison Between DES, 3DES, RC2, RC6, Blowfish and DES. *Procedings of National Conference of New Horizons in IT*, (pp. 143-148).

MILLER, A. (2012). *AaronMiller*. Obtenido de http://www.aaronmiller.in/thesis/

MORALES, M. (2014). *Gaussianos.* Obtenido de http://gaussianos.com/criptografia-cifrado-de-clave-publica-ii/

MORALES, M. (2014). *Gaussianos.* Obtenido de http://gaussianos.com/criptografia-cifrado-de-clave-publica-i/

MORENO, M. (2010). *Security Artwork*. Obtenido de http://www.securityartwork.es/2010/05/03/introduccion-a-la-esteganografia-ii/

PANTOJA, G. (2015). *E-continua*. Obtenido de http://www.e-continua.com.mx/index.php/eventsoption/13-introcripto

PÉREZ, R. (2012). *Security Artwork.* Obtenido de http://www.securityartwork.es/2012/02/24/ocultando-archivos-en-otros-lsb/

RAMAIYA, M., HEMRAJANI, N., & SAXENA, A. (2013). Security improvisation in image steganography using DES. *Advance Computing Conference (IACC)* (pp. 1094-1099). Ghaziabad: IEEE.

RED ACADÉMICA DE INVESTIGACIÓN ESPAÑOLA. (2013). *Rediris*. Obtenido de http://www.rediris.es/cert/doc/unixsec/node29.html

ROBERTSON, S. (2014). *eHow.* Obtenido de http://www.ehowenespanol.com/ventajas-algoritmos-rijndael-info_290285/

ROJAS, N., & HERNÁNDEZ, H. (2014). *Seguridad en redes.* Obtenido de http://seguridad-en-redes-mimi.blogspot.com/2012/03/algoritmo-3des.html

SAINI, J., & VERMA, H. (2013). A hybrid approach for image security by combining encryption and steganography. *IEEE Second International Conference on Image Information Processing (ICIIP)* (pp. 607-611). Shimla: IEEE.

SEGURA, G., & DÍAZ, A. (2014). *Boletín UPIITA*. Obtenido de http://www.boletin.upiita.ipn.mx/index.php/ciencia/215-cyt-numero-33/109-implementacion-del-algoritmo-esteganografico-lsb-least-significant-bit-estandar-en-archivos-de-audio-mp3

SSL247, E. (2015). *SSL247.* Obtenido de https://www.ssl247.es/certificats-ssl/rsa-dsa-ecc

VÁSQUEZ, M. (2007). *Universidad Pontificia Comillas.* Obtenido de http://www.iit.upcomillas.es/pfc/resumenes/46ea7511774d8.pdf

VIDAL, P. (2015). *Análisis y diseño de un modelo de notaría digital.* Universidad de Cuenca, Cuenca, Ecuador.

YOUTUBE. (2015). *Esteganografía: para cifrar mensajes en imágenes*. Obtenido de https://www.youtube.com/watch?v=qwIvN0fz5WE

ANEXOS

Anexo A: Código fuente principal

Implementación del algoritmo criptográfico base

La parte principal del código para el proceso de cifrado se muestra a continuación:

```
current = 0;
addRoundKey(output, current);

for (current = 1; current < Nr; current++) {
    subBytes(output);
    shiftRows(output);
    mixColumns(output);
    addRoundKey(output, current);
}
subBytes(output);
shiftRows(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso de descifrado se muestra a continuación:

```
current = Nr;
addRoundKey(output, current);

for (current = Nr - 1; current > 0; current--) {
    invShiftRows(output);
    invSubBytes(output);
    addRoundKey(output, current);
    invMixColumns(output);
}
invShiftRows(output);
invSubBytes(output);
addRoundKey(output, current);
return output;
```

El código del botón en la interfaz de usuario para cifrar/descifrar el mensaje se muestra a continuación:

```
private void ejecutejButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    //comprobación de la longitud de la clave
    if (getHexleng() == getHexKey().length()) {

        if (!flagFile) {
            this.setOriginalText(this.sourcejTextPane.getText());
        }
        String cad_val_pT;
        byte[] key = ConvertHex.hexToBytes(getHexKey());
        AES aes = new AES(key);
        if (encriptjComboBox.getSelectedIndex() == 0) {
            if (this.getOriginalText().length() > 0) {
                if (getHexKey() != null) {
                    try {
                        cad_val_pT = aes.validateText((this.getOriginalText()));
                        setCypherText(aes.encryptionAES(cad_val_pT));
                        this.convertjTextPane.setText(getCypherText());
                        exportjButton.setEnabled(true);
                        JOptionPane.showMessageDialog(null, "Encriptación exitosa");
                    } catch (Exception ex) {
                        ex.printStackTrace();
                    }
                }
            } else {
                JOptionPane.showConfirmDialog(this, "No existe texto para procesar");
                return;
            }
        } else {
            if (this.getOriginalText().length() > 0) {
                if (getHexKey() != null) {
                    try {
                        cad_val_pT = aes.validateText((this.getOriginalText()));
                        setCypherText(aes.decryptionAES(cad_val_pT));
                        this.convertjTextPane.setText(getCypherText());
                        exportjButton.setEnabled(true);
                        JOptionPane.showMessageDialog(null, "Desencriptación exitosa");
                    } catch (Exception ex) {
                        ex.printStackTrace();
                    }
                }
            } else {
                JOptionPane.showConfirmDialog(this, "No existe texto para procesar");
                return;
            }
        }
    } else {
        JOptionPane.showMessageDialog(null, "La llave debe de ser de " + typeKey);
    }
}
```

Creación e implementación del nuevo algoritmo criptográfico

La parte principal del código para el proceso de ShiftColumn se muestra a continuación:

```
protected int[][] shiftColumns(int[][] est) {
    int temp1, temp2, temp3, i;
    //columna 1
    temp1 = est[0][1];
    for (i = 0; i < Nb - 1; i++) {
        est[i][1] = est[(i + 1) % Nb][1];
    }
    est[Nb - 1][1] = temp1;
    // columna 2, se desplaza 1-byte
    temp1 = est[0][2];
    temp2 = est[1][2];
    for (i = 0; i < Nb - 2; i++) {
        est[i][2] = est[(i + 2) % Nb][2];
    }
    est[Nb - 2][2] = temp1;
    est[Nb - 1][2] = temp2;
    // columna 3, se desplaza 2-bytes
    temp1 = est[0][3];
    temp2 = est[1][3];
    temp3 = est[2][3];
    for (i = 0; i < Nb - 3; i++) {
        est[i][3] = est[(i + 3) % Nb][3];
    }
    est[Nb - 3][3] = temp1;
    est[Nb - 2][3] = temp2;
    est[Nb - 1][3] = temp3;
    return est;
}
```

La parte principal del código para el proceso de InvShiftColumn se muestra a continuación:

```
protected int[][] invShiftColumns(int[][] est) {
    int temp1, temp2, temp3, i;
    //columna 1;
    temp1 = est[Nb - 1][1];
    for (i = Nb - 1; i > 0; i--) {
        est[i][1] = est[(i - 1) % Nb][1];
    }
    est[0][1] = temp1;
    //columna 2
    temp1 = est[Nb - 1][2];
    temp2 = est[Nb - 2][2];
    for (i = Nb - 1; i > 1; i--) {
        est[i][2] = est[(i - 2) % Nb][2];
    }
    est[1][2] = temp1;
    est[0][2] = temp2;
    //columna 3
    temp1 = est[Nb - 3][3];
    temp2 = est[Nb - 2][3];
    temp3 = est[Nb - 1][3];
    for (i = Nb - 1; i > 2; i--) {
        est[i][3] = est[(i - 3) % Nb][3];
    }
    est[0][3] = temp1;
    est[1][3] = temp2;
    est[2][3] = temp3;
    return est;
}
```

La parte principal del código para el proceso de cifrado se muestra a continuación:

```
current = 0;
addRoundKey(output, current);
shiftColumns(output);

for (current = 1; current < Nr; current++) {
    subBytes(output);
    shiftRows(output);
    shiftColumns(output);
    mixColumns(output);
    addRoundKey(output, current);
}
subBytes(output);
shiftRows(output);
shiftColumns(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso de descifrado se muestra a continuación:

```
current = Nr;
addRoundKey(output, current);
invShiftColumns(output);
for (current = Nr - 1; current > 0; current--) {
    invShiftRows(output);
    invSubBytes(output);
    addRoundKey(output, current);
    invShiftColumns(output);
    invMixColumns(output);
}
invShiftRows(output);
invSubBytes(output);
invShiftColumns(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso para generación de claves se muestra a continuación:

```java
public void setKeyAES(byte[] key, int type) {
    keyAES = new int[key.length];
    int length = key.length;

    if (type == 1) {
        //Clave A
        for (int i = 0; i < length; i++) {
            keyAES[i] = key[i];
        }
    } else {
        //Clave B invertir clave A
        int i = 0;
        for (int j = length - 1; j > -1; j--) {
            keyAES[i] = key[j] + j;
            i++;
        }
        //Clave B rotar a la izquierda 3 posiciones y sumar posición
        int temp1, temp2, temp3;
        temp1 = key[0];
        temp2 = key[1];
        temp3 = key[2];
        for (i = 0; i < length - 3; i++) {
            keyAES[i] = key[i + 3] + i;
        }
        keyAES[length - 3] = temp1 + (length - 3);
        keyAES[length - 2] = temp2 + (length - 2);
        keyAES[length - 1] = temp3 + (length - 1);
    }
}
```

El código del botón en la interfaz de usuario para cifrar/descifrar el mensaje se muestra a continuación:

```java
private void ejecutejButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    //Comprobación de la longitud de la clave
    if (getHexleng() == getHexKey().length()) {
        if (!flagFile) {
            this.setOriginalText(this.sourcejTextPane.getText());
        }
        String cad_val_pT;
        byte[] key = ConvertHex.hexToBytes(getHexKey());
        if (encriptjComboBox.getSelectedIndex() == 0) {
            if (this.getOriginalText().length() > 0) {
                if (getHexKey() != null) {
                    try {
                        cad_val_pT = aes.validateText((this.getOriginalText()));
                        AES aes = new AES();
                        //AES clave A
                        aes.setKeyAES(key, 1);
                        aes.processKey(key.length);
                        String aesEncription1 = aes.encryptionAES(cad_val_pT);
                        //AES clave B
                        aes.setKeyAES(key, 2);
                        aes.processKey(key.length);
                        String aesEncription2 = aes.encryptionAES(aesEncription1);
                        setCypherText(aesEncription2);
                        this.convertjTextPane.setText(getCypherText());
                        exportjButton.setEnabled(true);
                        JOptionPane.showMessageDialog(null, "Encriptación exitosa");
                    } catch (Exception ex) {
                        ex.printStackTrace();
                    }
                }
            } else {
                JOptionPane.showConfirmDialog(this, "No existe texto para procesar");
                return;
            }
        } else {
            if (this.getOriginalText().length() > 0) {
                if (getHexKey() != null) {
                    try {
                        cad_val_pT = aes.validateText((this.getOriginalText()));
                        AES aes = new AES();
                        //AES clave B
                        aes.setKeyAES(key, 2);
                        aes.processKey(key.length);
                        String aesDescription1 = aes.decryptionAES(cad_val_pT);
                        //AES clave A
                        aes.setKeyAES(key, 1);
                        aes.processKey(key.length);
                        String aesDescription2 = aes.decryptionAES(aesDescription1);
                        setCypherText(aesDescription2);
                        this.convertjTextPane.setText(getCypherText());
                        exportjButton.setEnabled(true);
                        JOptionPane.showMessageDialog(null, "Desencriptación exitosa");
                    } catch (Exception ex) {
                        ex.printStackTrace();
                    }
                }
            } else {
                JOptionPane.showConfirmDialog(this, "No existe texto para procesar");
                return;
            }
        }
    } else {
        JOptionPane.showMessageDialog(null, "La llave debe de ser de " + typeKey);
    }
}
```

Implementación del algoritmo criptográfico en imágenes

La parte principal del código para el proceso para armar el mensaje se muestra a continuación:

```
private void setMessage(String message) {
    String bi = "";
    //el tamaño total del mensaje
    lenght = message.length() + flag.length() * 2;
    //transforma el valor ENTERO de la longitud en un valor BINARIO
    for (int i = 15; i >= 0; i--) {
        bi = bi + (((lenght & (1 << i)) > 0) ? "1" : "0");
    }
    //concatena todo el mensaje
    binaryMessage = getMensajeToBinary(flag) + bi + getMensajeToBinary(message);
}
```

La parte principal del código para el proceso para embeber el mensaje se muestra a continuación:

```
public void embebedMessage(BufferedImage bufferedImage, String message) {
    int tmp_count = 0;
    //proceso para concatenar todo el mensaje
    setMessage(message);
    //crea una imagen con la que se trabajara
    image = new BufferedImage(bufferedImage.getWidth(), bufferedImage.getHeight(), BufferedImage.TYPE_INT_RGB);
    //recorre toda la imagen pixel a pixel añadiendo 1 y 0 en los bits LSB
    for (int row = 0; row < image.getHeight(); row++) {
        for (int column = 0; column < image.getWidth(); column++) {
            //se obtiene el color del pixel en coordenadas (i,j)
            color = new Color(bufferedImage.getRGB(column, row));
            //mientras exista un mensaje, se procede a reemplazar los LSB
            if (tmp_count <= this.binaryMessage.length()) {
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String gren = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se reemplaza el ultimo bits
                red = replaceLSB(red);
                gren = replaceLSB(gren);
                blue = replaceLSB(blue);
                //cambia de binario a entero
                r = Integer.parseInt(red, 2);
                g = Integer.parseInt(gren, 2);
                b = Integer.parseInt(blue, 2);
            } else {
                r = color.getRed();
                g = color.getGreen();
                b = color.getBlue();
            }
            //se coloca en la nueva imagen con los valores
            image.setRGB(column, row, new Color(r, g, b).getRGB());
            tmp_count += 3;
        }
    }
}
```

La parte principal del código para el proceso para extraer el mensaje se muestra a continuación:

```
public String retrieveMessage(BufferedImage bufferedImage) {
    originalMessage = "No existe ningún mensaje oculto";
    if (confirmEmbebed(bufferedImage)) {//si se confirme continua
        //determina el tamaño del mensaje
        messageSize(bufferedImage);
        String[] s = new String[this.lenght];
        String tmp = "";
        //recorre toda la imagen pixel x pixel
        for (int row = 0; row < bufferedImage.getHeight(); row++) {
            for (int column = 0; column < bufferedImage.getWidth(); column++) {
                //se obtiene el color del pixel en coordenadas (i,j)
                color = new Color(bufferedImage.getRGB(column, row));
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String green = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se obtiene el bits LSB
                red = getLSB(red);
                green = getLSB(green);
                blue = getLSB(blue);
                //finaliza cuando se termina de leer todo el mensaje
                if (tmp.length() <= (this.lenght * 8)) {
                    tmp = tmp + red + green + blue;
                } else {
                    break;
                }
            }
        }
        //el String obtenido de 1 y 0, se los separa en un array de bytes
        int count_tmp = 0;
        for (int a = 0; a < (this.lenght * 8); a = a + 8) {
            s[count_tmp] = tmp.substring(a, a + 8);
            count_tmp++;
        }
        //convierte en mensaje a string
        originalMessage = getMensajeToString(s);
    }
    return originalMessage;
}
```

El código del botón en la interfaz de usuario para embeber/extraer el mensaje se muestra a continuación:

```
private void executejButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if (optionsjComboBox.getSelectedIndex() == 0) {
        if (!"".equals(originaljTextArea.getText())) {
            //oculta el mensaje en la imagen
            esteganos.embebedMessage(mipaint.getImage(), originaljTextArea.getText());
            //vuelve a pintar la imagen en pantalla
            mipaint.setImage(esteganos.getImage());
            savejButton.setEnabled(true);
            executejButton.setEnabled(false);
            JOptionPane.showMessageDialog(null, "El mensaje embebido exitosamente en la imagen");
        } else {
            JOptionPane.showMessageDialog(null, "Por  favor ingrese un mensaje");
        }
    } else {
        this.originaljTextArea.setText("");
        this.originaljTextArea.setText(esteganos.retrieveMessage(mipaint.getImage()));
        JOptionPane.showMessageDialog(null, "El mensaje ha sido extraido exitosamente de la imagen");
        exportjButton.setEnabled(true);
    }
}
```

Integración de los algoritmos criptográficos y esteganográficos

Prototipo I

La parte principal del código para el proceso para cifrar el mensaje se muestra a continuación:

```
current = 0;
addRoundKey(output, current);

for (current = 1; current < Nr; current++) {
    subBytes(output);
    shiftRows(output);
    mixColumns(output);
    addRoundKey(output, current);
}
subBytes(output);
shiftRows(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso para descifrar el mensaje se muestra a continuación:

```
current = Nr;
addRoundKey(output, current);

for (current = Nr - 1; current > 0; current--) {
    invShiftRows(output);
    invSubBytes(output);
    addRoundKey(output, current);
    invMixColumns(output);
}
invShiftRows(output);
invSubBytes(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso para cifrar y embeber el mensaje por el Prototipo I muestra a continuación:

```
public void encryptEmbebedMessage(BufferedImage bufferedImage, String message, String passwordHex) {
    int tmp_count = 0;
    byte[] key = ConvertHex.hexToBytes(passwordHex);
    AES aes = new AES(key);
    String cad_val_pT = aes.validateText(message);
    String cypherText = aes.encryptionAES(cad_val_pT);
    setEncriptedMessage(cypherText);
    //proceso para concatenar todo el mensaje
    setMessage(cypherText);
    //crea una imagen con la que se trabajara
    image = new BufferedImage(bufferedImage.getWidth(), bufferedImage.getHeight(), BufferedImage.TYPE_INT_RGB);
    //recorre toda la imagen pixel a pixel añadiendo 1 y 0 en los bits LSB
    for (int row = 0; row < image.getHeight(); row++) {
        for (int column = 0; column < image.getWidth(); column++) {
            //se obtiene el color del pixel en coordenadas (i,j)
            color = new Color(bufferedImage.getRGB(column, row));
            //mientras exista un mensaje, se procede a reemplazar los LSB
            if (tmp_count <= this.binaryMessage.length()) {
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String gren = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se reemplaza el ultimo bits
                red = replaceLSB(red);
                gren = replaceLSB(gren);
                blue = replaceLSB(blue);
                //cambia de binario a entero
                r = Integer.parseInt(red, 2);
                g = Integer.parseInt(gren, 2);
                b = Integer.parseInt(blue, 2);
            } else {
                r = color.getRed();
                g = color.getGreen();
                b = color.getBlue();
            }
            //se coloca en la nueva imagen con los valores
            image.setRGB(column, row, new Color(r, g, b).getRGB());
            tmp_count += 3;
        }
    }
}
```

La parte principal del código para el proceso para extraer y descifrar el mensaje por el Prototipo I se muestra a continuación:

```
public String retrieveDecryptMessage(BufferedImage bufferedImage, String passwordHex) {
    originalMessage = "No existe ningún mensaje oculto";
    if (confirmEmbebed(bufferedImage)) {//si se confirme continua
        //determina el tamaño del mensaje
        messageSize(bufferedImage);
        String[] s = new String[this.lenght];
        String tmp = "";
        //recorre toda la imagen pixel x pixel
        for (int row = 0; row < bufferedImage.getHeight(); row++) {
            for (int column = 0; column < bufferedImage.getWidth(); column++) {
                //se obtiene el color del pixel en coordenadas (i,j)
                color = new Color(bufferedImage.getRGB(column, row));
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String green = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se obtiene el bits LSB
                red = getLSB(red);
                green = getLSB(green);
                blue = getLSB(blue);
                //finaliza cuando se termina de leer todo el mensaje
                if (tmp.length() <= (this.lenght * 8)) {
                    tmp = tmp + red + green + blue;
                } else {
                    break;
                }
            }
        }
        //el String obtenido de 1 y 0, se los separa en un array de bytes
        int count_tmp = 0;
        for (int a = 0; a < (this.lenght * 8); a = a + 8) {
            s[count_tmp] = tmp.substring(a, a + 8);
            count_tmp++;
        }
        //convierte en mensaje a string
        String messageEncrypt = getMensajeToString(s);
        setEncriptedMessage(messageEncrypt);
        byte[] key = ConvertHex.hexToBytes(passwordHex);
        AES aes = new AES(key);
        String cad_val_pT = aes.validateText(messageEncrypt);
        setEncriptedMessage(cad_val_pT);
        originalMessage = aes.decryptionAES(cad_val_pT);
    }
    return originalMessage;
}
```

El código del botón en la interfaz de usuario para cifrar/embeber y extraer/descifrar el mensaje con el Prototipo I se muestra a continuación:

```
private void executejButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String hexKey = ConvertHex.asciitoHEX(new String(passwordjPasswordField.getPassword()));
    int hexSize = hexKey.length();
    if (getHexleng() == hexSize) {
        if (optionsjComboBox.getSelectedIndex() == 0) {
            if (!"".equals(originaljTextArea.getText())) {

                //encripta y oculta el mensaje en la imagen
                esteganos.encryptEmbebedMessage(mipaint.getImage(), originaljTextArea.getText(), hexKey);
                //vuelve a pintar la imagen en pantalla
                mipaint.setImage(esteganos.getImage());
                encryptjTextArea.setText(esteganos.getEncriptedMessage());
                savejButton.setEnabled(true);
                executejButton.setEnabled(false);
                JOptionPane.showMessageDialog(null, "Mensaje cifrado y embebido exitosamente");
            } else {
                JOptionPane.showMessageDialog(null, "Por  favor ingrese un mensaje");
            }
        } else {
            this.originaljTextArea.setText("");
            //exrae y desencripta el mensaje
            this.originaljTextArea.setText(esteganos.retrieveDecryptMessage(mipaint.getImage(), hexKey));
            encryptjTextArea.setText(esteganos.getEncriptedMessage());
            JOptionPane.showMessageDialog(null, "Mensaje extraido y descrifrado exitosamente");
            exportjButton.setEnabled(true);
        }
    } else {
        JOptionPane.showMessageDialog(null, "La llave debe de ser de " + typeKey);
    }
}
```

Prototipo II

La parte principal del código para el proceso para cifrar el mensaje se muestra a continuación:

```
current = 0;
addRoundKey(output, current);
shiftColumns(output);

for (current = 1; current < Nr; current++) {
    subBytes(output);
    shiftRows(output);
    shiftColumns(output);
    mixColumns(output);
    addRoundKey(output, current);
}
subBytes(output);
shiftRows(output);
shiftColumns(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso para descifrar el mensaje se muestra a continuación:

```
current = Nr;
addRoundKey(output, current);
invShiftColumns(output);
for (current = Nr - 1; current > 0; current--) {
    invShiftRows(output);
    invSubBytes(output);
    addRoundKey(output, current);
    invShiftColumns(output);
    invMixColumns(output);
}
invShiftRows(output);
invSubBytes(output);
invShiftColumns(output);
addRoundKey(output, current);
return output;
```

La parte principal del código para el proceso para cifrar y embeber el mensaje por el Prototipo II muestra a continuación:

```
public void encryptEmbebedMessaje(BufferedImage bufferedImage, String message, String passwordHex) {
    int tmp_count = 0;
    byte[] key = ConvertHex.hexToBytes(passwordHex);
    AES aes = new AES();
    aes.setKeyAES(key, 1);
    aes.processKey(key.length);
    String cad_val_pT = aes.validateText(message);
    String cypherText1 = aes.encryptionAES(cad_val_pT);
    //AES clave B
    aes.setKeyAES(key, 2);
    aes.processKey(key.length);
    String cypherText2 = aes.encryptionAES(cypherText1);
    setEncriptedMessage(cypherText2);
    //proceso para concatenar todo el mensaje
    setMessage(cypherText2);
    //crea una imagen con la que se trabajara
    image = new BufferedImage(bufferedImage.getWidth(), bufferedImage.getHeight(), BufferedImage.TYPE_INT_RGB);
    //recorre toda la imagen pixel a pixel añadiendo 1 y 0 en los bits LSB
    for (int row = 0; row < image.getHeight(); row++) {
        for (int column = 0; column < image.getWidth(); column++) {
            //se obtiene el color del pixel en coordenadas (i,j)
            color = new Color(bufferedImage.getRGB(column, row));
            //mientras exista un mensaje, se procede a reemplazar los LSB
            if (tmp_count <= this.binaryMessage.length()) {
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String gren = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se reemplaza el ultimo bits
                red = replaceLSB(red);
                gren = replaceLSB(gren);
                blue = replaceLSB(blue);
                //cambia de binario a entero
                r = Integer.parseInt(red, 2);
                g = Integer.parseInt(gren, 2);
                b = Integer.parseInt(blue, 2);
            } else {
                r = color.getRed();
                g = color.getGreen();
                b = color.getBlue();
            }
            //se coloca en la nueva imagen con los valores
            image.setRGB(column, row, new Color(r, g, b).getRGB());
            tmp_count += 3;
        }
    }
}
```

La parte principal del código para el proceso para extraer y descifrar el mensaje por el Prototipo II se muestra a continuación:

```
public String retrieveDecryptMessage(BufferedImage bufferedImage, String passwordHex) {
    originalMessage = "No existe ningún mensaje oculto";
    if (confirmEmbebed(bufferedImage)) {//si se confirme continua
        //determina el tamaño del mensaje
        messageSize(bufferedImage);
        String[] s = new String[this.lenght];
        String tmp = "";
        //recorre toda la imagen pixel x pixel
        for (int row = 0; row < bufferedImage.getHeight(); row++) {
            for (int column = 0; column < bufferedImage.getWidth(); column++) {
                //se obtiene el color del pixel en coordenadas (i,j)
                color = new Color(bufferedImage.getRGB(column, row));
                //se convierten a su equivalente en binario
                String red = toBinary((byte) color.getRed());
                String green = toBinary((byte) color.getGreen());
                String blue = toBinary((byte) color.getBlue());
                //se obtiene el bits LSB
                red = getLSB(red);
                green = getLSB(green);
                blue = getLSB(blue);
                //finaliza cuando se termina de leer todo el mensaje
                if (tmp.length() <= (this.lenght * 8)) {
                    tmp = tmp + red + green + blue;
                } else {
                    break;
                }
            }
        }
        //el String obtenido de 1 y 0, se los separa en un array de bytes
        int count_tmp = 0;
        for (int a = 0; a < (this.lenght * 8); a = a + 8) {
            s[count_tmp] = tmp.substring(a, a + 8);
            count_tmp++;
        }
        //convierte en mensaje a string
        String messageEncrypt = getMensajeToString(s);
        setEncriptedMessage(messageEncrypt);
        byte[] key = ConvertHex.hexToBytes(passwordHex);
        AES aes = new AES();
        String cad_val_pT = aes.validateText(messageEncrypt);
        setEncriptedMessage(cad_val_pT);
        //AES clave A
        aes.setKeyAES(key, 2);
        aes.processKey(key.length);
        String cypherText1 = aes.decryptionAES(cad_val_pT);
        //AES clave B
        aes.setKeyAES(key, 1);
        aes.processKey(key.length);
        originalMessage = aes.decryptionAES(cypherText1);
    }
    return originalMessage;
}
```

El código del botón en la interfaz de usuario para cifrar/embeber y extraer/descifrar el mensaje con el Prototipo II se muestra a continuación:

```
private void executejButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String hexKey = ConvertHex.asscitoHEX(new String(passwordjPasswordField.getPassword()));
    int hexSize = hexKey.length();
    if (getHexleng() == hexSize) {
        if (optionsjComboBox.getSelectedIndex() == 0) {
            if (!"".equals(originaljTextArea.getText())) {

                //oculta el mensaje en la imagen
                esteganos.encryptEmbebedMessaje(mipaint.getImage(), originaljTextArea.getText(), hexKey);
                //vuelve a pintar la imagen en pantalla
                mipaint.setImage(esteganos.getImage());
                encryptjTextArea.setText(esteganos.getEncriptedMessage());
                savejButton.setEnabled(true);
                executejButton.setEnabled(false);
                JOptionPane.showMessageDialog(null, "Mensaje cifrado y embebido exitosamente");
            } else {
                JOptionPane.showMessageDialog(null, "Por  favor ingrese un mensaje");
            }
        } else {
            this.originaljTextArea.setText("");
            this.originaljTextArea.setText(esteganos.retrieveDecryptMessage(mipaint.getImage(), hexKey));
            encryptjTextArea.setText(esteganos.getEncriptedMessage());
            JOptionPane.showMessageDialog(null, "Mensaje extraido y descifrado exitosamente");
            exportjButton.setEnabled(true);
        }
    } else {
        JOptionPane.showMessageDialog(null, "La llave debe de ser de " + typeKey);
    }
}
```