REPÚBLICA DEL ECUADOR

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE POSGRADO

MAESTRÍA EN COMPUTACIÓN MENCIÓN EN SEGURIDAD INFORMÁTICA

TEMA:

TÉCNICAS CRIPTOGRÁFICAS PARA MEJORAR LA SEGURIDAD DE LA TRANSFERENCIA DE ARCHIVOS EN LA EMPRESA DE TELECOMUNICACIONES SERVIMATANGO C.A

Trabajo de Titulación previo a la obtención del Título de Magíster en Computación Mención en Seguridad Informática

AUTORA: Ing. Nelly Alexandra Matango Proaño

DIRECTOR: Ing. Henry Patricio Farinango Endara Msc.

IBARRA - ECUADOR

2025

REPÚBLICA DEL ECUADOR

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1003229646		
APELLIDOS Y NOMBRES:	Matango Proaño Nelly Alexandra		
DIRECCIÓN:	Ibarra, Juan Francisco Cevallos 440		
E-MAIL:	nellymatpro@gmail.com		
TELÉFONO FIJO:	No disponible	TELÉFONO MÓVIL:	0997462285

DATOS DE LA OBRA	
TÍTULO:	Técnicas criptográficas para mejorar la seguridad de la transferencia de archivos en la empresa de telecomunicaciones Servimatango c.a
AUTOR:	Ing. Nelly Alexandra Matango Proaño
FECHA: DD/MM/AA	19/05/2025
SOLO PARA TRABAJO DE GRADO	
PROGRAMA:	[ ] PREGRADO [X] POSGRADO
TÍTULO POR EL QUE OPTA:	Magíster en computación con mención en seguridad informática.
ASESOR/DIRECTOR:	PhD. Iván García, MsC. Henry Farinango

CONSTANCIA

La autora manifiesta que la obra objeto de la presente autorización es original y se la desarrollo, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que se asume la responsabilidad sobre el contenido de la misma y saldrá en defensa la Universidad Técnica del Norte en caso de reclamación por parte de terceros.

Ibarra, a los 19 días del mes de mayo de 2025

LA AUTORA:

Nombre: Nelly Alexandra Matango Proaño

CI: 1003229646

Ibarra, 01 de abril de 2025

Dra.
Lucía Yépez
DECANA FACULTAD DE POSGRADO

ASUNTO: Conformidad con el documento final

Señora Decana:

Nos permitimos informar a usted que revisado el Trabajo final de Grado TÉCNICAS CRIPTOGRÁFICAS PARA MEJORAR LA SEGURIDAD DE LA TRANSFERENCIA DE ARCHIVOS EN LA EMPRESA DE TELECOMUNICACIONES SERVIMATANGO C.A de la maestrante Nelly Alexandra Matango Proaño, de la Maestría en Computación Mención Seguridad Informática, certificamos que han sido acogidas y satisfechas todas las observaciones realizadas.

Atentamente,

	Apellidos y Nombres	**Firma**
Director/a	Msc Farinango Endara Henry Patricio	Firmado electrónicamente por: HENRY PATRICIO FARINANGO ENDARA
Asesor/a	PhD. García Santillán Iván Danilo	IVAN DANILO GARCIA SANTILLAN Digitally signed by IVAN DANILO GARCIA SANTILLAN Date: 2025.04.02 15:13:24 -05'00'

DEDICATORIA

A mi amado esposo,

Por su paciencia inagotable, por su apoyo en cada desafío y por creer en mí incluso en los momentos en los que dudé de mí misma. Su confianza y amor han sido mi mayor fortaleza en este camino.

Con todo mi cariño y gratitud,

Nelly Alexandra Matango Proaño

AGRADECIMIENTO

A mi familia, por su amor incondicional, por ser mi refugio en los momentos difíciles y por brindarme siempre su apoyo y aliento para seguir adelante. Sin ustedes, este logro no sería posible.

A mis amigos, por su compañía, por las palabras de ánimo y por estar a mi lado en este camino, celebrando cada pequeño avance y alentándome en cada obstáculo. Su amistad ha sido un pilar fundamental en esta etapa.

A mi tutor, por su guía, paciencia y valiosas enseñanzas. Su apoyo y confianza en mi trabajo. Gracias por compartir su conocimiento y por ayudarme a dar lo mejor de mí en este proceso.

A todos los que, de una u otra forma, han contribuido a este logro, mi más sincero agradecimiento.

Con gratitud,

Nelly Alexandra Matango Proaño

ÍNDICE

TEMA: i
IDENTIFICACIÓN DE LA OBRA i
CONSTANCIA ii
DEDICATORIA....iv
AGRADECIMIENTO....v
ÍNDICE vi
ÍNDICE DE TABLAS....ix
ÍNDICE DE FIGURAS x
RESUMEN....xi
ABSTRACT xii
CAPITULO I....1
EL PROBLEMA 1
1.1. Problema de Investigación....1
1.2. Interrogantes de la Investigación....2
1.3. Objetivos de la Investigación 2
1.3.1. Objetivo General 2
1.3.2. Objetivos Específicos 2
1.4. Justificación....3
CAPÍTULO II....5
MARCO REFERENCIAL 5
2.1. Antecedentes 5
2.2. Marco Teórico 6
2.2.1. La Seguridad de la Información 6
2.2.2. Transferencia de Archivos....7
2.2.3. Criptografía....8
2.2.3.1. Sistemas Clásicos de Sustitución....8
2.2.3.2. Otros Criptosistemas....11
2.2.3.3. Complejidad, Factorización y Logaritmo Discreto....11
2.2.4. Claves Criptográficas 12
2.2.4.1. Ciclo de Vida de las Claves Criptográficas 13
2.2.4.2. Directrices para la Gestión de Claves Criptográficas: 14
2.2.5. Cifrado....16
2.2.5.1. Tipos de cifrado....18
2.2.6. Funciones Hash 21
2.2.6.1. Aplicaciones de las Funciones Hash en el Cifrado de Archivos.....21
2.2.6.2. Importancia de las Funciones Hash en la Seguridad de Archivos 23
2.2.6.3. Limitaciones de las Funciones Hash.....23
2.2.6.4. Ejemplos de Algoritmos Hash Comunes:....24
2.2.7. Firma Digital 24
2.2.8. VPN....28
2.2.8.1. Características Clave de una VPN:....30
2.2.8.2. ¿Cómo Contribuye una VPN al Cifrado de Archivos?....30
2.2.8.3. Beneficios de Usar una VPN para el Cifrado de Archivos....31
2.2.8.4. Limitaciones de una VPN en el Cifrado de Archivos....32
2.2.8.5. Casos de Uso Comunes 32
2.2.9. Técnicas Modernas de Cifrado de Archivos....33
2.2.9.1. Cifrado por Bloques: 33
2.2.9.2. Cifrado por Flujo:....33
2.2.9.3. Cifrado Híbrido: 33
2.2.10. Errores más frecuentes en el cifrado....33

2.2.10.1. Utilización de protocolos obsoletos 33
2.2.10.2. Utilización de claves muy cortas 35
2.2.11. Errores en los mecanismos de compartición de credenciales 36
2.2.12. ¿Cómo se pueden corregir los errores en la implementación? 38
2.2.13. Casos de Estudio que presentaron fallas en Controles Criptográficos 41
2.3. Marco legal 45
2.3.1. Constitución de la República del Ecuador 45
2.3.2. National Institute of Standards and Technology (NIST) 45
2.3.3. Ley Orgánica de Protección de Datos en el Ecuador 45
2.3.4. Ley Orgánica de Telecomunicaciones 46
2.3.5. Estrategia Nacional de Ciberseguridad 46
2.3.6. Ley Orgánica de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos 47
2.3.7. Normas Técnicas Ecuatorianas NTE INEN-ISO/IEC 27000 48
CAPITULO III 49
MARCO METODOLÓGICO 49
3.1. Descripción del área de estudio / Descripción del grupo de estudio 49
3.2. Enfoque de Investigación 50
3.2.1. Enfoque Mixto 50
3.2.1.1. Características del Enfoque Mixto 50
3.2.1.2. Tipos de Diseño Mixto 50
3.2.1.3. Situación Actual de la Empresa 51
3.2.2. Enfoque Descriptivo 51
3.2.3. Enfoque Aplicado 51
3.3. Tipo de Investigación 52
3.3.1. Investigación Exploratoria 52
3.3.1.1. Características de la Investigación Exploratoria 52
3.3.1.2. Análisis de la encuesta 52
3.3.2. Investigación Experimental 53
3.3.2.1. Características de la investigación experimental 53
3.3.2.2. Ejemplo de uso 53
3.4. Procedimiento de investigación 54
3.4.1. Selección de Técnicas Criptográficas 54
3.4.2. Análisis de Información Recabada 54
3.4.3. Recolección de información 54
3.4.4. Comparación de Metodologías de Auditoría 55
3.4.4.1. OWASP Cryptographic Storage Cheat Sheet 55
3.4.4.2. NIST SP 800-57 (Recomendación para la Gestión de Claves Criptográficas) 55
3.4.4.3. ISO/IEC 27001 (Gestión de la Seguridad de la Información) 55
3.4.5. Implementación 56
3.4.5.1. Verificación de la integridad 56
3.4.5.2. Verificación de la Confidencialidad 59
3.4.5.3. Verificación de la Autenticidad 61
3.4.6. Monitoreo y Evaluación 63
3.4.7. Consideraciones bioéticas 63
CAPITULO IV 65
RESULTADOS 65
4.1. Evaluación del rendimiento de algoritmos de cifrado simétrico. 65
4.2. Evaluación del rendimiento de algoritmos de cifrado asimétrico. 66
4.3. Evaluación del rendimiento de algoritmos de cifrado simétrico y asimétrico 67
4.4. Evaluación del Impacto de diferentes métodos criptográficos en términos de velocidad, eficacia y eficiencia. 68
4.5. Verificación de Integridad, Confidencialidad y Autenticidad de los Archivos 70
4.5.1. Verificación de Integridad y Autenticidad 70

4.5.2. Verificación de la Confidencialidad 71
4.5.2.1. Cifrado de Archivos 71
4.5.2.2. Control de Acceso 72
4.5.2.3. Ejemplo con 7zip 72
4.5.2.4. Pruebas de Penetración 75
4.5.3. Transferencia de archivos seguros entre equipos remotos 75
4.5.4. Pruebas de Python para la verificación 75
4.6. Verificación de Integridad, Confidencialidad y Autenticidad de los Archivos mediante Firma Digital 76
4.7. Discusión 77
CAPITULO V 78
CONCLUSIONES Y RECOMENDACIONES 78
5.1. CONCLUSIONES 78
5.2. RECOMENDACIONES 79
REFERENCIAS 81
ANEXOS 84

ÍNDICE DE TABLAS

Tabla 1 Ramas de la teoría de la información 8
Tabla 2 Cifrado Vigenere 10
Tabla 3 Longitudes de Claves Recomendadas 15
Tabla 4 Propiedades del algoritmo hash seguro 24
Tabla 5 Tabla de Usos de Protocolo VPN 29
Tabla 6 Errores comunes en los mecanismos de compartición de credenciales 37
Tabla 7 Errores y soluciones más frecuentes en los controles criptográficos 40
Tabla 8 Resumen de la comparación 56
Tabla 9 Resultados del Estudio Comparativo de los Algoritmos AES, 3DES y ChaCha20 65
Tabla 10 Resultados de la comparación de algoritmos cifrado asimétrico 66
Tabla 11 Tabla Comparativa entre métodos simétricos y asimétricos 67
Tabla 12 Resultados de la prueba en Python AES 69
Tabla 13 Resultados de la evaluación AES vs. AES+RSA 70

ÍNDICE DE FIGURAS

Figura 1 Principios fundamentales de la seguridad de la información 6
Figura 2. Cifrado César 9
Figura 3. Cifrado César Ejemplo 9
Figura 4 Ciclo de Vida de las Claves Criptográficas 14
Figura 5. Esquema básico de un sistema de cifrado 16
Figura 6. Elementos de un Esquema de Cifrado 17
Figura 7. Cifrado Simétrico 18
Figura 8. Cifrado Asimétrico 19
Figura 9. Ejemplo de Cifrado Asimétrico 20
Figura 10 Generación y verificación de la función hash 23
Figura 11. Firma digital para verificar la integridad de los datos 25
Figura 12. Firma digital para verificar la autenticidad de los datos 26
Figura 13. Ejemplo de arquitectura VPN de pasarela a pasarela 31
Figura 14. Mapa de ubicación de Servimatango C.A 49
Figura 15 Obtener Hash con 7-zip 71
Figura 16 Cifrado con 7-zip paso1 73
Figura 17 archivo comprimido con contraseña 73
Figura 18 Error al colocar contraseña errónea 74
Figura 19 Verificacion CIA con Firma Digital - Kleopatra 76

RESUMEN

Esta investigación aborda el problema de la falta de cifrado adecuado en las transferencias de archivos en Servimatango C.A., una empresa de telecomunicaciones que enfrenta riesgos significativos de seguridad de datos. El objetivo principal fue identificar y aplicar técnicas criptográficas que mejoren de manera efectiva la protección de la información sensible, mediante un enfoque metodológico que incluyó revisión de algoritmos (AES, RSA, SHA-256), pruebas de penetración, y evaluación de integridad, confidencialidad y autenticidad. Los resultados determinaron que la combinación híbrida AES (cifrado rápido de datos) + RSA (intercambio seguro de claves) era la más eficiente, complementada con carpetas seguras en Veracrypt y cifrado de archivos con 7-zip ó Kleopatra. Para conexiones externas, se implementó OpenVPN, asegurando canales remotos, mientras que la integridad y autenticidad se verificaron mediante hashes SHA-256 generados por 7-zip, comparados antes y después de cada transferencia. Las conclusiones destacan que estas técnicas mejoraron significativamente la seguridad, mitigando riesgos de interceptación o manipulación, y se recomendó capacitar al personal, gestionar claves de forma centralizada y realizar auditorías periódicas para mantener la robustez del sistema. El aporte principal fue un modelo accesible y escalable, adaptable a empresas similares con necesidades de protección de datos.

Palabras clave: *cifrado de archivos, transferencia segura de archivos, gestión de claves, métodos criptográficos.*

ABSTRACT

This research addresses the issue of inadequate encryption in file transfers at Servimatango C.A., a telecommunications company facing significant data security risks. The primary objective was to identify and apply cryptographic techniques that effectively enhance the protection of sensitive information, using a methodological approach that included algorithm review (AES, RSA, SHA-256), penetration testing, and assessment of integrity, confidentiality and authenticity. Results determined that the hybrid combination of AES (fast data encryption) + RSA (secure key exchange) was the most efficient, complemented by secure folders in Veracrypt and file encryption with 7-Zip or Kleopatra. For external connections, OpenVPN was implemented, ensuring secure remote channels, while integrity and authenticity were verified through SHA-256 hashes generated by 7-Zip, which were compared before and after each transfer. Conclusions highlight that these techniques significantly improved security, mitigating interception or manipulation risks and it was recommended to train staff, manage keys centrally and perform regular audits to maintain the system's robustness. The main contribution was an accessible and scalable model, adaptable to similar companies with data protection needs.

Keywords: file encryption, secure file transfer, key management, cryptographic methods.

CAPITULO I

EL PROBLEMA

Seguridad limitada en la transferencia de archivos en la empresa de telecomunicaciones Servimatango C.A

1.1. Problema de Investigación

Los problemas de ciberseguridad se han incrementado a nivel mundial, en especial en microempresas que aún no identifican los ataques o no tienen implementado un análisis específico, de los riesgos y peligros que puedan existir en la base de datos o información interna (Sánchez Guzmán, 2021).

El cifrado de archivos según (Veritas Technologies, 2024) como un método de seguridad indispensable para proteger datos confidenciales, ya que los resultados revelaron que el uso de cifrado puede prevenir accesos no autorizados y asegurar la integridad de la información, subrayan que, el cifrado se convierte en una barrera esencial para la protección de datos.

Como se menciona en el análisis de (WeLiveSecurity, 2024) es común que los usuarios asuman que están compartiendo archivos solo con personas de confianza, pero la naturaleza de muchas aplicaciones de intercambio de archivos puede permitir que usuarios desconocidos accedan a datos sin autorización. Esto se agrava por la posibilidad de que los enlaces de descarga sean compartidos inadvertidamente, lo que puede llevar a un acceso no autorizado de información sensible. Por lo tanto, compartir archivos sin cifrado no solo aumenta la vulnerabilidad a ataques externos, sino que también puede resultar una pérdida de control sobre la información compartida, lo que puede tener consecuencias legales y reputacionales significativas para los individuos y las organizaciones.

Entre las múltiples causas de perdida de los atributos de seguridad de la información (confidencialidad, disponibilidad e integridad), se encuentra la transferencia de archivos sin protección criptográfica a través de medios inseguros.

1.2. Interrogantes de la Investigación

Para el presente proyecto se consideran las siguientes interrogantes de investigación:

- ¿Cuáles son los métodos de protección criptográfica más comunes para la transferencia de archivos?
- ¿Qué tipos de errores acontecen con frecuencia en la implementación de controles criptográficos en empresas de telecomunicaciones?
- Que metodología de implementación de controles criptográficos minimiza errores
- ¿Cómo garantizar y medir la autenticidad, velocidad, eficacia y eficiencia en los archivos transferidos con protección criptográfica?

1.3. Objetivos de la Investigación

1.3.1. *Objetivo General*

Analizar técnicas criptográficas efectivas que pueden aplicarse para mejorar la seguridad de la transferencia de archivos en la empresa de telecomunicaciones Servimatango C.A.

1.3.2. *Objetivos Específicos*

- Identificar y describir las características, fortalezas y debilidades de los métodos de protección criptográfica más utilizados, así como su aplicabilidad en diferentes entornos y situaciones.
- Analizar y comprender las posibles fallas en la configuración, gestión y uso de controles criptográficos, así como las consecuencias que pueden tener en la seguridad de la transferencia de archivos.

- Evaluar el impacto de la implementación de métodos criptográficos en la velocidad, eficacia y eficiencia de la transferencia de archivos.
- Verificar la integridad, confidencialidad y autenticidad de los archivos, así como identificar posibles métodos de firma digital o certificados digitales.

1.4. Justificación

Las empresas de telecomunicaciones en Ecuador manejan información confidencial y sensible, incluyendo datos personales de sus clientes, registros financieros, información estratégica y otros datos comerciales valiosos. Ante el aumento constante de las amenazas cibernéticas y la creciente sofisticación de los ataques, resulta fundamental implementar medidas de seguridad sólidas, como la protección criptográfica de archivos en movimiento, para proteger esta información y garantizar la confidencialidad, integridad y disponibilidad de los datos.

Según (Huang et al., 2020) el uso de técnicas criptográficas ofrece una capa adicional de protección contra amenazas como el robo de datos o la interceptación de comunicaciones.

La Seguridad Informática según mencionan en su libro (Samaniego & Ponce, 2021) es la encargada de asegurar que todos los recursos tecnológicos físicos y lógicos de una empresa sean utilizados de manera única, con una previa autenticación y autorización, así como su modificación y actualización solo sea posible a las personas responsables asignadas y dentro de los límites de sus privilegios.

De acuerdo con el Informe Estado Actual de la Ciberseguridad en Ecuador 2024 elaborado por (IT Ahora, 2024), la protección de datos se ha convertido en la habilidad de seguridad más relevante para las organizaciones. Este estudio revela que, para salvaguardar su información, las empresas deben implementar un monitoreo continuo y una gestión de riesgos

efectiva. La inversión en capacitación constante de los colaboradores es crucial para minimizar los riesgos ante amenazas cibernéticas.

Este proyecto se enmarca en el Plan de Creación de Oportunidades 2021-2025 de Ecuador el cual establece en uno de sus ejes la seguridad integral estable y se relaciona con la línea de investigación N.º 10 vigente y aprobado por el Honorable Consejo Universitario de la UTN[1]: Desarrollo, aplicación de software y cyber security (seguridad cibernética) (UTN, 2023).

Por lo tanto, el estudio de "Técnicas criptográficas para mejorar la seguridad de la transferencia de archivos en la empresa de telecomunicaciones Servimatango " se justifica por la necesidad de desarrollar soluciones viables que protejan eficazmente los datos críticos de esta organización, adaptándose a los desafíos específicos del mercado local y las regulaciones aplicables.

[1] UTN Universidad Técnica del Norte. Ibarra – Ecuador.

CAPÍTULO II

MARCO REFERENCIAL

2.1. Antecedentes

En Servimatango se transfieren diariamente varios archivos con sus proveedores y clientes, muchos de esos datos no están cifrados, en ocasiones únicamente están comprimidos, otros archivos tienen contraseña de Word o Excel, pero sin protección criptográfica real, de modo que son susceptibles a la captura, modificación, eliminación y mal uso por parte de atacantes.

En el año 2019 de acuerdo con (vpnMentor, 2019), Ecuador sufrió la filtración de más de 18GB de datos distribuidos en una variedad de archivos, que incluía nombres, información financiera y datos civiles de hasta 20 millones de personas, ocurrió desde un servidor en Miami que no contaba con los requisitos de seguridad establecidos y que era administrado por Novaestrat, una empresa ecuatoriana de marketing y análisis, es así que en el año 2021 se estableció una ley de protección de datos personales que se debía aplicar en todas las empresas para lograr tener un correcto manejo de la información.

En julio de 2021, la Corporación Nacional de Telecomunicaciones (CNT) de Ecuador sufrió un ataque informático de tipo ransomware. Este incidente afectó los sistemas internos de la empresa, incluyendo áreas de facturación, activaciones y recargas. El ataque fue atribuido al ransomware RansomEXX, conocido por bloquear el acceso a sistemas o archivos y exigir un rescate para su liberación. (Primicias EC, 2021)

Los proveedores de telecomunicaciones manejan grandes cantidades de datos personales y empresariales de sus clientes, que pueden ser objeto de robo o filtración por parte de hackers, competidores o actores estatales. Estos datos pueden incluir información sensible como nombres,

direcciones, números de tarjetas de crédito, contraseñas, hábitos de consumo o comunicaciones privadas. (Cyberearmag, 2024)

Implementar un cifrado robusto es esencial para cualquier estrategia de ciberseguridad, ya que minimiza el riesgo de filtraciones de datos y ataques cibernéticos, garantizando así la integridad y confidencialidad de la información.

2.2. Marco Teórico

2.2.1. *La Seguridad de la Información*

Según la norma ISO/IEC 27001:2022, la seguridad de la información se define como la protección de la confidencialidad, integridad y disponibilidad de la información, como podemos apreciar en la *Figura 1* (Peltier, 2020).

La seguridad de la información es fundamental para proteger la información sensible durante su transferencia. Según un estudio realizado por (International IT, 2022), el cifrado asegura que los datos sean ininteligibles para cualquier persona no autorizada, no solo protege los datos en tránsito, sino que también previene accesos no autorizados a la información almacenada.

Figura 1

Principios fundamentales de la seguridad de la información

2.2.2. *Transferencia de Archivos*

La transferencia de archivos es un proceso esencial en el intercambio de datos en entornos digitales, que permite la copia o movimiento de archivos entre sistemas a través de redes. Según (IBM, 2021) este proceso se rige por protocolos de comunicación que definen las reglas para la transmisión de información, siendo el Protocolo de Transferencia de Archivos (FTP) y su variante segura, Protocolo de transferencia segura de archivos (SFTP), los más utilizados. Sin embargo, el uso de FTP presenta riesgos significativos, como la falta de autenticación y el registro inadecuado de violaciones de seguridad, lo que puede exponer datos sensibles a accesos no autorizados. Las organizaciones están comenzando a adoptar soluciones avanzadas como la Transferencia de Archivos Gestionada (MFT[2]), que no solo proporciona un medio seguro para la transferencia de archivos, sino que también mejora la visibilidad operativa y permite detectar problemas en tiempo real (IBM, 2021).

En la transferencia de archivos las principales dificultades que podemos encontrar son: que el archivo no llegue al destinatario, que el archivo sea robado en el proceso de transferencia, que el archivo sea interceptado y modificado por un tercero. En cualquiera de las tres situaciones se estaría afectando los principios fundamentales de la seguridad de la información como se muestra en la *Figura 1* que se expuso anteriormente.

La transferencia de archivos en la nube está respaldada por medidas de seguridad robustas, incluyendo cifrado y controles de acceso. Esto asegura que los datos se mantengan seguros durante la transferencia y almacenamiento en la nube, dejando la vulnerabilidad en el lado del cliente o usuario quien podría descargar esa información y compartirla sin tomar las debidas precauciones. (AWS, 2022)

[2] MFT Managed File Transfer

2.2.3. *Criptografía*

La criptografía es una disciplina interdisciplinaria que combina matemáticas, informática e ingeniería para proteger la información.

La criptografía, junto con la teoría de códigos y la compresión de información, forma parte de la teoría de la información. Su objetivo principal es estructurar la información para que solo el receptor legítimo pueda descifrarla, incluso si terceros interceptan la transmisión. (Plaza, 2021)

Tabla 1

Ramas de la teoría de la información

RAMA	OBJETIVO	TÉCNICAS MATEMÁTICAS	APLICACIONES COMUNES
Teoría de Códigos	Preparar la información para corregir errores durante la transmisión.	Espacios vectoriales sobre cuerpos finitos, curvas algebraicas.	DT, CD, DVD, ADSL, fibra, WIFI.
Criptografía	Encriptar la información para protegerla de accesos no autorizados.	Grupos finitos, curvas elípticas, espacios de Hilbert.	HTTPS, firma digital, autenticación, votación electrónica, DNI electrónico.
Compresión	Reducir el volumen de datos para una transmisión eficiente.	Espacios de Hilbert, wavelets.	MP3, MP4, DivX, H265.

Nota. Adaptado del libro de Plaza, Francisco (2021). Manual de Criptografía, Fundamentos matemáticos

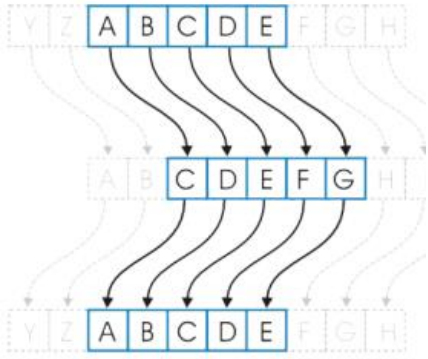
2.2.3.1. Sistemas Clásicos de Sustitución

a) César

El cifrado César es uno de los métodos criptográficos más antiguos y sencillos. Consiste en desplazar cada letra del texto original un número fijo de posiciones en el alfabeto.

Figura 2.

Cifrado César



Nota. Adaptado del libro de Plaza, Francisco (2021). Manual de Criptografía, Fundamentos matemáticos

Si se utiliza un desplazamiento de 3, la letra 'A' se convierte en 'D', la 'B' en 'E', y así sucesivamente

Figura 3.

Cifrado César Ejemplo

Original	a	b	c	d	e	f	g	h	i	j	k	l	m
Cifrado	d	e	f	g	h	i	j	k	l	m	n	o	p

Original	n	o	p	q	r	s	t	u	v	w	x	y	z
Cifrado	q	r	s	t	u	v	w	x	y	z	a	b	c

.

Nota. Adaptado del libro de Plaza, Francisco (2021). Manual de Criptografía, Fundamentos matemáticos

Por ejemplo: Software siguiendo la tabla, podemos codificar el mismo mensaje, dando como resultado al siguiente texto CODIFICADO: Vriwzduh

b) Vigenere

El cifrado de Vigenère es una mejora del cifrado César que utiliza una palabra clave para determinar el desplazamiento de cada letra. Esto hace que sea más difícil de descifrar que el cifrado César.

Tabla 2

Cifrado Vigenere

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Nota. Adaptado del libro de Plaza, Francisco (2021). Manual de Criptografía, Fundamentos matemáticos

Ejemplo: Si el mensaje es "HOLA" y la clave es "ABC", el cifrado sería:

* H + A = H
* O + B = P
* L + C = N
* A + A = A

El texto cifrado sería "HPNA".

2.2.3.2. Otros Criptosistemas

a) ***r-grafos:*** Sistemas que utilizan grafos para la sustitución de caracteres.

b) ***Transformaciones Afines*****:** Utilizan funciones lineales para cifrar.

DES, Triple DES y AES: Algoritmos más complejos utilizados en la actualidad.

2.2.3.3. Complejidad, Factorización y Logaritmo Discreto

a) Operaciones y Tiempos de Cómputo

La eficiencia de un algoritmo criptográfico depende de la complejidad de sus operaciones. Las operaciones básicas deben ser rápidas, mientras que las operaciones que dificultan el criptoanálisis deben ser computacionalmente costosas.

b) Factorización

La factorización de números grandes es un problema fundamental en criptografía. Algunos algoritmos de factorización incluyen:

- Fuerza bruta
- Factorización de Fermat
- Factorización ρ de Pollard

- Algoritmo de Dixon

c) Logaritmo Discreto

El logaritmo discreto es otra operación matemática importante en criptografía. Se utiliza en varios algoritmos de clave pública.

d) Números Primos

Los números primos son esenciales en la criptografía moderna, especialmente en algoritmos como RSA.

e) Pruebas de Primalidad

Existen varias pruebas para determinar si un número es primo:

- Fermat
- Solovay-Strassen
- Miller-Rabin

f) Construcción de Números Primos

Es crucial poder generar números primos grandes de manera eficiente para su uso en criptografía.

g) Criptografía Asimétrica

- ***El Gamal.- se*** *basa en el problema del logaritmo discreto y se utiliza tanto para la comunicación cifrada como para la firma digital*
- ***Diffie-Hellman.-*** permite a dos partes establecer una clave secreta compartida a través de un canal no seguro.
- **RSA.-** es uno de los criptosistemas de clave pública más utilizados. Se basa en la dificultad de factorizar números grandes.

2.2.4. *Claves Criptográficas*

Las claves criptográficas son elementos fundamentales en la seguridad de la información, ya que permiten cifrar y descifrar datos, garantizando su confidencialidad, integridad y autenticidad. (Barker, 2020)

Una clave criptográfica es una cadena de bits utilizada en algoritmos criptográficos para transformar datos legibles (texto plano) en datos ilegibles (texto cifrado) y viceversa. Las claves pueden ser simétricas (una misma clave para cifrar y descifrar) o asimétricas (un par de claves: pública y privada). (NIST, 2020)

2.2.4.1.Ciclo de Vida de las Claves Criptográficas

a) Generación:

- Creación de claves utilizando generadores seguros.
- Uso de algoritmos aprobados por el NIST (por ejemplo, AES, RSA).

b) Distribución:

- Transferencia segura de claves a los usuarios o sistemas autorizados.
- Uso de protocolos como TLS o mecanismos de intercambio de claves (por ejemplo, Diffie-Hellman).

c) Almacenamiento:

- Protección de claves mediante Hardware Security Modules (HSM[3]) o gestores de claves.

d) Uso:

- Aplicación de claves en operaciones de cifrado, descifrado, firma digital o autenticación.

e) Rotación:

[3] HSM Módulo de Seguridad de hardware

- Cambio periódico de claves para reducir el riesgo de compromiso.

***f*)** Revocación:

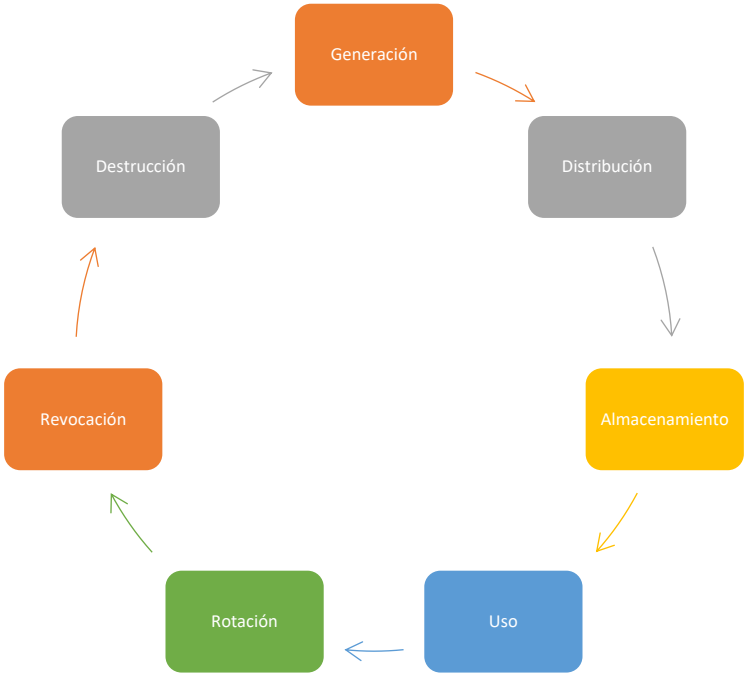
- Inhabilitación de claves comprometidas o no válidas.

g) Destrucción:

- Eliminación segura de claves cuando ya no son necesarias.

Figura 4

Ciclo de Vida de las Claves Criptográficas



Nota. Adaptado de Barker, E. (2020). *Recommendation for key management:* https://doi.org/10.6028/NIST.SP.800-57pt1r5

2.2.4.2. Directrices para la Gestión de Claves Criptográficas:

a) Longitud de las Claves

- La longitud de las claves debe ser adecuada para el nivel de seguridad requerido.

- Ejemplo: Para AES, se recomiendan claves de 128, 192 o 256 bits.

b) Períodos de Validez

- Las claves deben tener un período de validez limitado para reducir el riesgo de compromiso.

- Ejemplo: Claves simétricas para cifrado de datos pueden tener una validez de 1 a 2 años.

c) Almacenamiento Seguro

- Las claves deben almacenarse en dispositivos seguros, como HSM o módulos criptográficos validados por el NIST.

d) Gestión de Claves Comprometidas

- Las claves comprometidas deben revocarse inmediatamente y reemplazarse por nuevas claves.

Tabla 3

Longitudes de Claves Recomendadas

Algoritmo	**Longitud de Clave**	**Nivel de Seguridad**
AES	128 bits	Alto
AES	192 bits	Muy Alto
AES	256 bits	Extremadamente Alto
RSA	2048 bits	Alto
RSA	3072 bits	Muy Alto
ECC	256 bits	Alto
ECC	384 bits	Muy Alto

Nota. Adaptado de Barker, E. (2020). *Recommendation for key management:* https://doi.org/10.6028/NIST.SP.800-57pt1r5

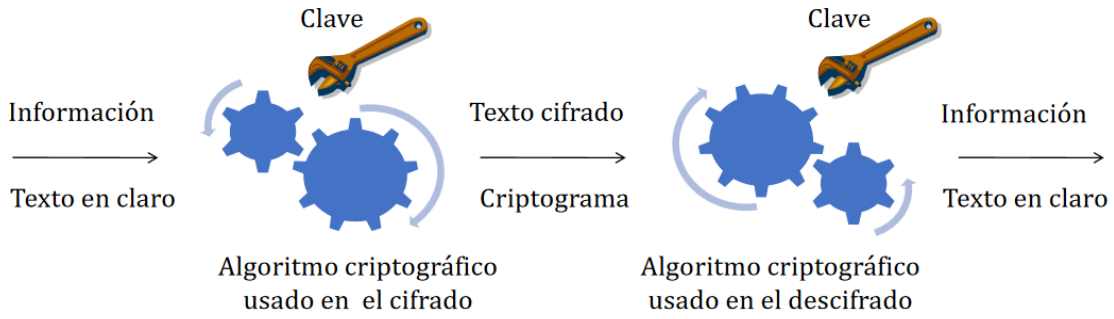
2.2.5. *Cifrado*

El cifrado es un proceso fundamental en la seguridad de la información que convierte datos legibles (texto plano) en un formato ilegible (texto cifrado) utilizando algoritmos matemáticos y claves criptográficas. El cifrado permite que solo las partes autorizadas puedan acceder a la información original, protegiéndola de accesos no autorizados. Este proceso es esencial tanto para la protección de datos en reposo como en tránsito, asegurando que la información sensible, como datos personales y financieros, permanezcan confidenciales. El uso de claves criptográficas es crucial, ya que la seguridad del cifrado depende de la complejidad de estas claves y de los algoritmos utilizados. Un cifrado seguro debe ser resistente a ataques, como el de fuerza bruta, donde un atacante intenta adivinar la clave. (Simplilearn.com, 2020)

El principal objetivo del cifrado de datos es evitar que los fisgones lean datos confidenciales en tránsito, el cifrado se puede utilizar para cualquier forma de comunicación en línea.

Figura 5.

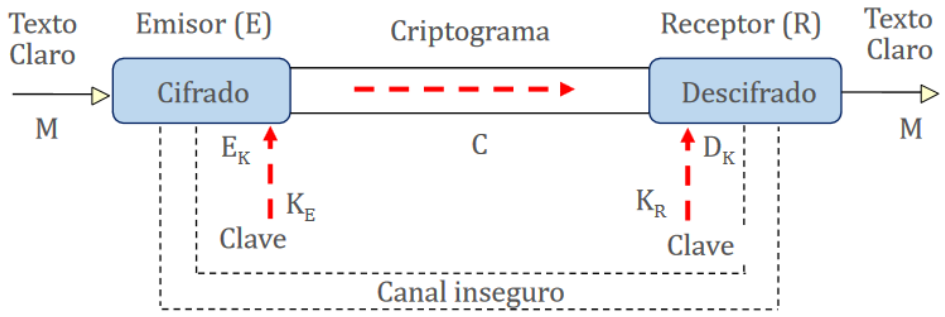
Esquema básico de un sistema de cifrado



Nota. Adaptado de Criptografía para Ingenieros (p. 404), por J. Ramió , 2022.

Figura 6.

Elementos de un Esquema de Cifrado



Texto en claro o texto base	Algoritmo de descifrado
Criptograma o texto cifrado	Clave usada en emisión
Canal o medio de transmisión	Clave usada en recepción
Algoritmo de cifrado	Alfabeto usado en la cifra

Nota. Adaptado de Criptografía para Ingenieros (p. 405), por J. Ramió , 2022.

Los elementos que forman parte de un criptosistema son:

M: que representa el mensaje o texto en claro, que por sí solo es legible o interpretable.

C: que representa al criptograma, esto es el texto cifrado que se trasmitirá por el canal de comunicación, el cual por definición es inseguro y por ello es necesario cifrar la información.

E: que representa la función de cifrado que se aplica al texto en claro. La letra E proviene del inglés Encrypt.

D: que representa la función de descifrado que se aplica al texto cifrado para recuperar el texto en claro. La letra D proviene del inglés Decrypt.

K: que representa la clave empleada para cifrar el texto en claro M, o bien para descifrar el criptograma

Las claves de cifrado KE y de descifrado KD operarán de forma diferente si se trata de una cifra simétrica o de una cifra asimétrica.

2.2.5.1. Tipos de cifrado

Existen dos tipos principales de cifrado:

• Cifrado simétrico o de clave secreta: utiliza la misma clave para cifrar y descifrar la información, la clave debe ser conocida por el emisor y el receptor

• Cifrado asimétrico o de clave pública: utiliza dos claves, una para cifrar y otra para descifrar la información, cuando se cifra con la clave pública se puede descifrar con la privada y viceversa

a) Cifrado simétrico

El cifrado simétrico es más eficiente que el cifrado asimétrico, pero requiere que las dos partes que se comunican compartan la misma clave. La clave se puede compartir de forma segura a través de canales seguros, como un canal de comunicación encriptado o una reunión en persona.

Algunos algoritmos de cifrado simétrico populares incluyen:

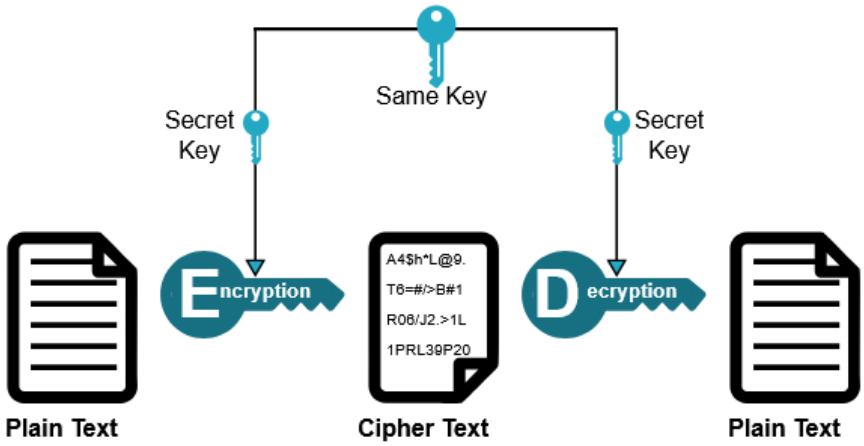
• AES: Advanced Encryption Standard, un algoritmo de cifrado de bloque adoptado por el gobierno de los Estados Unidos.

• DES: Data Encryption Standard, un algoritmo de cifrado de bloque que fue el estándar de facto para el cifrado de datos durante muchos años.

• Triple DES: una versión más segura de DES que utiliza tres rondas de cifrado.

Figura 7.

Cifrado Simétrico



Nota. Adaptado de AboutSSL, 2022, symmetric encryption vs asymmetric encryption (https://aboutssl.org/symmetric-encryption-vs-asymmetric-encryption/).

b) Cifrado Asimétrico

El cifrado asimétrico no requiere que las dos partes que se comunican compartan la misma clave. Una clave se utiliza para cifrar la información y la otra clave se utiliza para descifrarla. La clave pública se puede compartir de forma segura con cualquier persona, mientras que la clave privada se mantiene en secreto. Algunos algoritmos de cifrado asimétrico populares incluyen:

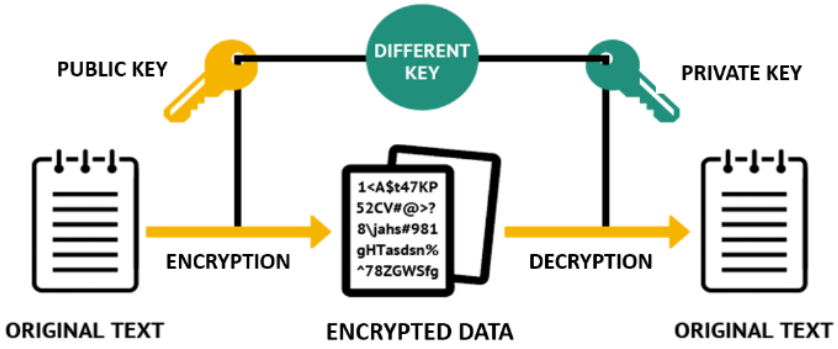
• RSA: Rivest-Shamir-Adleman, un algoritmo de cifrado de clave pública que es uno de los más utilizados en la actualidad.

• ElGamal: un algoritmo de cifrado de clave pública que es similar a RSA.

• Diffie-Hellman: un protocolo de intercambio de claves que se utiliza para establecer una clave de sesión segura entre dos partes.

Figura 8.

Cifrado Asimétrico

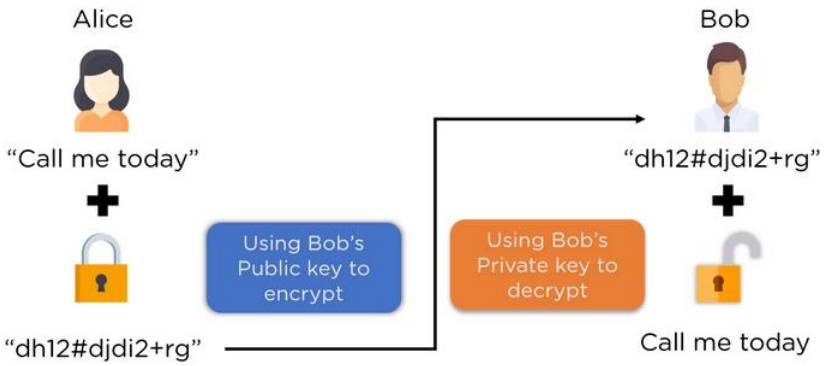


Nota. Adaptado de Digital Signature Algorithm (DSA) in Cryptography, por Baivab Kumar Jena, 2023, simplilearn (https://www.simplilearn.com/tutorials/cryptography-tutorial/digital-signature-algorithm). CC BY 2.0

Por ejemplo, si Alice necesita enviar un mensaje a Bob, tanto la clave pública como la privada deben pertenecer a Bob.

Figura 9.

Ejemplo de Cifrado Asimétrico



Nota. Adaptado de Digital Signature Algorithm (DSA) in Cryptography, por Baivab Kumar Jena, 2023, simplilearn (https://www.simplilearn.com/tutorials/cryptography-tutorial/digital-signature-algorithm).

El proceso para la imagen de arriba es el siguiente:

Paso 1: Alice usa primero la clave pública de Bob para cifrar el mensaje

Paso 2: El mensaje cifrado llega a Bob

Paso 3: Bob descifra el mensaje con su clave secreta

Esto elimina el requisito de que el remitente y el destinatario intercambien claves secretas, minimizando la ventana de oportunidad para la explotación.

2.2.6. *Funciones Hash*

Las funciones hash son herramientas fundamentales en la criptografía y la seguridad de la información. Aunque no se utilizan directamente para cifrar archivos, desempeñan un papel crucial en la verificación de la integridad de los datos, la autenticación y la protección de contraseñas. (NIST, 2015)

Una función hash es un algoritmo matemático que toma una entrada (como un archivo o un mensaje) y produce una cadena de caracteres de longitud fija, conocida como valor hash o digest. (ISO/IEC, 2018)

Las funciones hash tienen las siguientes características clave:

Deterministas: La misma entrada siempre produce el mismo hash.

Unidireccionales: Es computacionalmente inviable revertir el proceso (es decir, obtener la entrada original a partir del hash).

Resistentes a colisiones: Es extremadamente difícil encontrar dos entradas diferentes que produzcan el mismo hash.

2.2.6.1. Aplicaciones de las Funciones Hash en el Cifrado de Archivos.

Aunque las funciones hash no cifran archivos directamente, son esenciales en los siguientes aspectos relacionados con el cifrado:

a) Verificación de la Integridad de los Archivos

Uso: Las funciones hash se utilizan para generar un valor único (hash) de un archivo. Si el archivo se modifica, el hash cambia.

Ejemplo: Al descargar un archivo, puedes comparar su hash con el proporcionado por el proveedor para asegurarte de que no ha sido alterado.

Herramientas: sha256sum (Linux/Mac), CertUtil (Windows).

b) Autenticación de Mensajes (HMAC)

Uso: Las funciones hash se combinan con una clave secreta para crear un código de autenticación de mensajes (HMAC). Esto garantiza que el mensaje no ha sido alterado y proviene de una fuente confiable.

Ejemplo: En transferencias de archivos, el HMAC verifica la autenticidad del archivo.

c) Protección de Contraseñas

Uso: Las contraseñas se almacenan como hashes en lugar de texto plano. Cuando un usuario ingresa su contraseña, se compara con el hash almacenado.

Ejemplo: Sistemas de autenticación como LDAP[4] o bases de datos de usuarios.

d) Firma Digital

Uso: Las funciones hash se utilizan para generar un resumen único de un archivo, que luego se cifra con una clave privada para crear una firma digital.

Ejemplo: Verificación de la autenticidad de software o documentos.

e) Derivación de Claves

Uso: Las funciones hash se utilizan en algoritmos como PBKDF2 o bcrypt para derivar claves criptográficas a partir de contraseñas.

[4] LDAP Protocolo Ligero de Acceso a Directorios

Ejemplo: Cifrado de archivos con contraseñas.

2.2.6.2.Importancia de las Funciones Hash en la Seguridad de Archivos

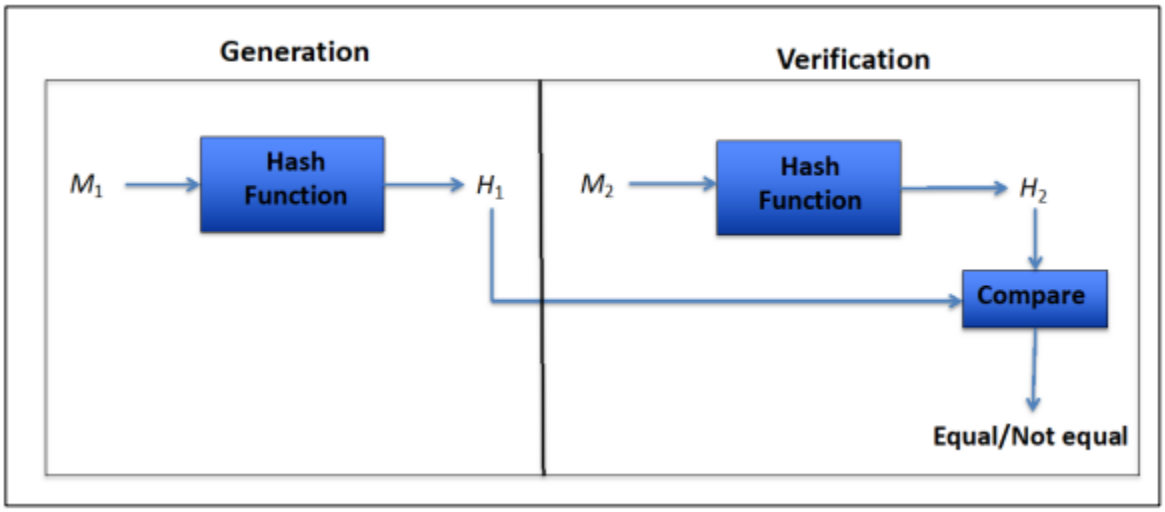
Garantizan la Integridad: Permiten detectar alteraciones no autorizadas en los archivos.

Protegen la Autenticidad: Aseguran que los archivos provienen de una fuente confiable.

Mejoran la Seguridad: Almacenar hashes en lugar de datos sensibles reduce el riesgo de exposición.

Figura 10

Generación y verificación de la función hash



Nota. Adaptado de NIST800-175B, 2020, Guideline for Using Cryptographic Standards in the Federal Government (https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-175Br1.pdf/).

2.2.6.3.Limitaciones de las Funciones Hash.

No son Cifrado: Los hashes no pueden utilizarse para cifrar archivos, ya que no permiten recuperar la información original.

Vulnerabilidades en Algoritmos Antiguos: Algoritmos como MD5 y SHA-1 son vulnerables a colisiones y no deben usarse en aplicaciones críticas.

Dependencia de la Seguridad del Hash: Si un atacante obtiene el hash, puede intentar ataques de fuerza bruta o de diccionario para encontrar la entrada original.

2.2.6.4.Ejemplos de Algoritmos Hash Comunes:

MD5: Aunque ampliamente utilizado, ya no se considera seguro debido a vulnerabilidades conocidas.

SHA-1: Similar a MD5, ha sido descontinuado para usos críticos.

SHA-256 (parte de la familia SHA-2): Ampliamente utilizado y considerado seguro.

SHA-3: La última generación de algoritmos hash, diseñada para ser resistente a ataques futuros.

Tabla 4

Propiedades del algoritmo hash seguro

Algoritmo	Tamaño del Mensaje (bits)	Tamaño del Bloque (bits)	Tamaño de la Palabra (bits)	Tamaño del resumen del mensaje (bits)
SHA-1	**$< 2^{64}$**	**512**	**32**	**160**
SHA-224	**$< 2^{64}$**	**512**	**32**	**224**
SHA-256	**$< 2^{64}$**	**512**	**32**	**256**
SHA-384	**$< 2^{64}$**	**1024**	**64**	**384**
SHA-512	**$< 2^{64}$**	**1024**	**64**	**512**
SHA-512/224	**$< 2^{64}$**	**1024**	**64**	**224**
SHA-512/256	**$< 2^{64}$**	**1024**	**64**	**256**

Nota. Adaptado de Federal Information Processing Standards Publication 180-4 (p. 3), por NIST, 2015, Secure Hash Standard (https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf).

2.2.7. *Firma Digital*

Según señala (Jena, 2023) que la firma digital se utiliza para autenticar y verificar documentos y datos. Esto es necesario para evitar manipulaciones, modificaciones digitales o falsificaciones documentales durante la transmisión de documentos, mantenido la confidencialidad e integridad de estos.

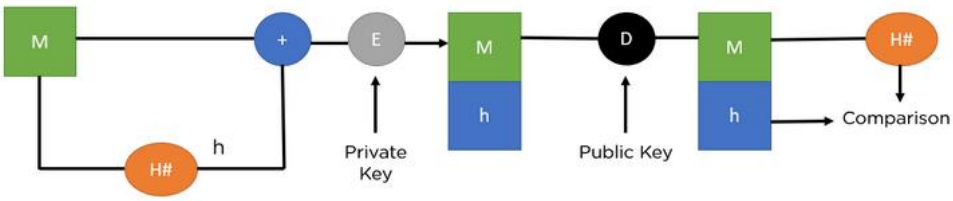
A diferencia del cifrado asimétrico, la firma se cifra con la clave privada y se descifra con la clave pública. Debido a los vínculos entre las claves, descifrar el archivo con la clave pública demuestra que se utilizó la clave privada correcta para firmar el archivo, demostrando la autenticidad de la firma.

Una firma digital es un tipo de firma electrónica basada en criptografía y se utiliza para autenticar la identidad del remitente de un mensaje o del firmante de un documento y para garantizar que el contenido original del mensaje o documento no se modifique. (Gupta & Goyal, 2020)

La firma digital combina hash y cifrado asimétrico para autenticar la procedencia y verificar la integridad de los datos.

Figura 11.

Firma digital para verificar la integridad de los datos



Nota. Adaptado de Digital Signature Algorithm (DSA) in Cryptography, por Baivab Kumar Jena, 2023, simplilearn (https://www.simplilearn.com/tutorials/cryptography-tutorial/digital-signature-algorithm).

M - Texto sin formato

H - Función hash

h - resumen de hash

'+': agrupa texto sin formato y resumen

E-cifrado

D - Descifrado

La imagen anterior muestra todo el proceso, desde la firma de la clave hasta su verificación.

Paso 1: M, el mensaje original se pasa primero a una función hash indicada por H# para crear un resumen.

Paso 2: A continuación, agrupa el mensaje junto con el resumen hash h y lo cifra utilizando la clave privada del remitente.

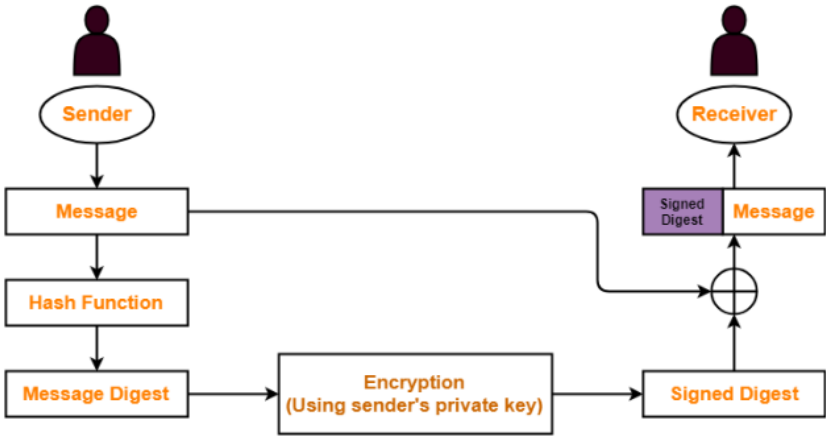
Paso 3: envía el paquete cifrado al receptor, quien puede descifrarlo utilizando la clave pública del remitente.

Paso 4: Una vez que descifra el mensaje, pasa por la misma función hash (H#) para generar un resumen similar.

Paso 5: Compara el hash recién generado con el valor hash incluido recibido junto con el mensaje. Si coinciden, verifica la integridad de los datos.

Figura 12.

Firma digital para verificar la autenticidad de los datos



Nota. Adaptado de Digital Signature Algorithm (DSA) in Cryptography, por Baivab Kumar Jena, 2023, simplilearn (https://www.simplilearn.com/tutorials/cryptography-tutorial/digital-signature-algorithm).

El diagrama de bloques del proceso de firma digital es el siguiente:

Mensaje o Documento: Es el dato o documento que se debe firmar.

Hashing: el documento o mensaje se convierte en un hash de longitud fija mediante una función hash criptográfica.

Cifrado: el hash se cifra utilizando la clave privada del remitente.

Firma: El hash cifrado es la firma digital del documento.

Verificación: Luego, la firma se verifica utilizando la clave pública del remitente para garantizar la validez de la firma y la autenticidad del documento.

El proceso comienza con el envío de un mensaje de un lugar a otro. Después de eso, se aplica un algoritmo *hash* criptográfico a este mensaje para crear un hash de este. Luego, el *hash* se cifra utilizando la clave privada del remitente. Para crear la firma digital, el hash cifrado se adjunta a continuación al mensaje original.

Una vez que el *hash* se ha descifrado utilizando la clave pública del remitente, el destinatario lo compara con un *hash* nuevo creado a partir del mensaje original para determinar

si la firma digital es genuina. Este proceso de descifrado garantiza que el mensaje no haya sido alterado durante el tránsito y que se pueda confiar en que proviene del remitente. Una vez más, la firma digital demuestra que los datos son originales, inalterados y provienen de una fuente confiable.

2.2.8. *VPN*

Una VPN (Red Privada Virtual) es una tecnología utilizada para garantizar la seguridad y privacidad de las comunicaciones en internet. Aunque su principal función es cifrar el tráfico de red, también puede desempeñar un papel importante en la protección de archivos durante su transferencia. (NIST, 2020)

La investigación sobre la transferencia de archivos a través de redes VPN es crucial especialmente en empresas, donde el teletrabajo se ha vuelto común y necesario. Según Morales García (2022), las VPN permiten la transferencia segura de datos a través de redes públicas, garantizando la confidencialidad y la integridad de la información. Este tipo de tecnología es esencial para las empresas que necesitan proteger datos sensibles mientras permiten el acceso remoto a sus empleados. Sin embargo, la implementación de VPN también presenta desafíos, como la necesidad de una infraestructura adecuada y el manejo de protocolos de transferencia que pueden afectar el rendimiento.

El uso de protocolos de transferencia de archivos como SMB/CIFS en entornos VPN puede limitar la eficiencia de las transferencias, ya que estos protocolos no están optimizados para la transmisión de datos en alta demanda (Triofox, 2023). Esto sugiere que, aunque las VPN son efectivas para la seguridad, su diseño y la elección de protocolos son factores críticos que deben considerarse para mejorar la experiencia del usuario y la velocidad de transferencia.

Además, el uso de tecnologías de almacenamiento en caché y la implementación de protocolos de streaming, como HTTP, pueden mejorar significativamente el rendimiento de las transferencias de archivos en entornos VPN (Triofox, 2023). Esto implica que las soluciones de acceso remoto deben evolucionar para integrar estas tecnologías, permitiendo a los trabajadores remotos acceder a los archivos de manera más eficiente, sin comprometer la seguridad.

La encriptación de datos es un componente fundamental en la transferencia de archivos a través de VPN. PrivadoVPN (2023) señala que la encriptación AES de 256 bits es el estándar de oro para proteger la información durante su transmisión. Esto no solo asegura que los datos permanezcan confidenciales, sino que también protege a los usuarios de posibles ataques cibernéticos. La combinación de estas tecnologías y prácticas es esencial para garantizar la seguridad y eficiencia en la transferencia de archivos en un entorno de trabajo remoto.

La elección del protocolo de cifrado es crucial. Protocolos como OpenVPN e IKEv2/IPsec son preferidos por su capacidad de ofrecer un cifrado sólido y un rendimiento óptimo. El uso de estos protocolos asegura que la información se transmita de manera segura a través de un túnel cifrado, protegiendo así la privacidad del usuario y la integridad de los datos (Surfshark, 2020). Los protocolos VPN de renombre utiliza el cifrado AES-256. Se recomienda no usar PPTP ya que utiliza únicamente 128 bits.

Tabla 5

Tabla de Usos de Protocolo VPN

PROTOCOLO	**CIFRADO**	**USO**
OpenVPN	AES-256	Uso diario; poner una VPN en un router
IKEv2/IPsec	AES-256	Dispositivos móviles, conexiones de corto alcance, uso diario
WireGuard	ChaCha20	La mejor novedad para uso diario
SoftEther	AES-256	Uso diario

PPTP	128 bits	Un protocolo de túnel; es inútil utilizarlo si no es con tecnología anticuada
SSTP	AES-256	Un protocolo de túnel de Microsoft; para conectar dispositivos Windows
L2TP/IPsec	AES-256	No tiene sentido utilizarlo, ya que IKEv2 es mejor en todos los sentidos

Nota. Adaptado de Antanas Rimeikis. (2023, June 12). *¿Qué es el cifrado de una VPN y cómo funciona?* Surfshark. https://surfshark.com/es/blog/vpn-cifrado

2.2.8.1. Características Clave de una VPN:

Cifrado de Datos: Utiliza protocolos como OpenVPN, IPSec o WireGuard para cifrar el tráfico.

Privacidad: Oculta la dirección IP del usuario, protegiendo su identidad.

Seguridad: Protege los datos de ataques como el "*man-in-the-middle*".

2.2.8.2. ¿Cómo Contribuye una VPN al Cifrado de Archivos?

Aunque una VPN no cifra archivos directamente, protege los archivos durante su transferencia a través de internet. Esto es especialmente útil en los siguientes escenarios:

a) Transferencia Segura de Archivos

Cuando se envían archivos a través de internet (por ejemplo, mediante FTP, correo electrónico o servicios en la nube), una VPN cifra el tráfico, evitando que los archivos sean interceptados.

b) Acceso Remoto a Archivos

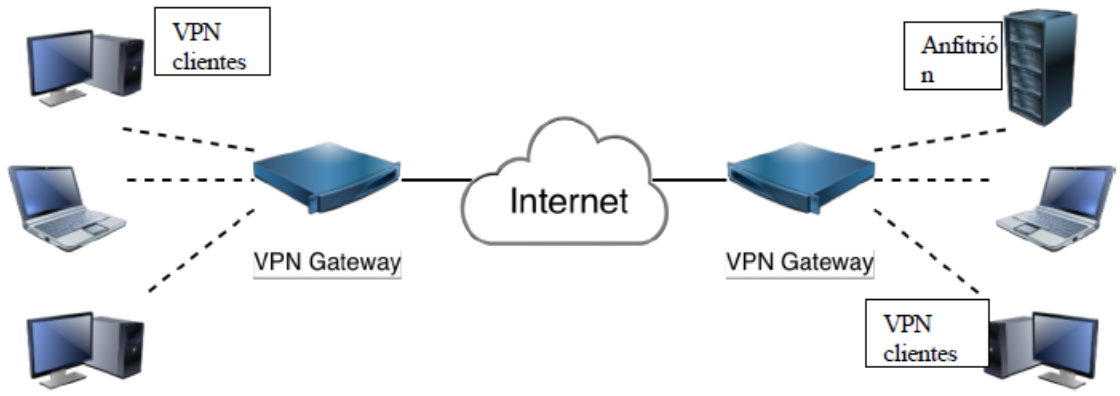
Los empleados que acceden a archivos almacenados en servidores corporativos desde ubicaciones remotas pueden usar una VPN para garantizar que la conexión sea segura.

c) Uso en Redes Públicas

En redes Wi-Fi públicas (como cafeterías o aeropuertos), una VPN protege los archivos transferidos de posibles ataques.

Figura 13.

Ejemplo de arquitectura VPN de pasarela a pasarela



Nota. Adaptado de Guide to IPsec VPNs, por NIST Special Publication 800-77, 2020, (p. 12) (https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-77r1.pdf).

2.2.8.3.Beneficios de Usar una VPN para el Cifrado de Archivos

Confidencialidad:

Garantiza que los archivos transferidos no puedan ser leídos por terceros.

Integridad:

Protege los archivos de modificaciones no autorizadas durante la transferencia.

Privacidad:

Oculta la identidad del usuario y la ubicación del servidor.

Cumplimiento Normativo:

Ayuda a cumplir con regulaciones como GDPR[5], HIPAA[6] o PCI-DSS[7] al proteger los datos en tránsito.

2.2.8.4. Limitaciones de una VPN en el Cifrado de Archivos

No Cifra Archivos en Reposo:

Una VPN solo protege los archivos durante su transferencia. No cifra archivos almacenados en dispositivos o servidores.

Dependencia del Proveedor de VPN:

La seguridad de la VPN depende de la confiabilidad del proveedor. Algunos proveedores pueden registrar el tráfico.

Rendimiento:

El cifrado y descifrado de datos pueden ralentizar la velocidad de transferencia.

Configuración Compleja:

Requiere configuración y mantenimiento, especialmente en entornos corporativos.

2.2.8.5. Casos de Uso Comunes

Teletrabajo:

Los empleados remotos usan VPN para acceder de manera segura a archivos corporativos.

Transferencia de Datos Sensibles:

Organizaciones que transfieren archivos confidenciales (por ejemplo, en el sector financiero o sanitario).

Uso en Redes Públicas:

---

[5] GDPR Reglamento General de Protección de Datos

[6] HIPAA Ley de Portabilidad y Responsabilidad de Seguros de Salud

[7] PCI-DSS Estándar de seguridad de datos de la industria de tarjetas de pago

Personas que necesitan enviar archivos desde redes Wi-Fi no seguras.

Evitar Censura:

Usuarios en países con restricciones de internet usan VPN para acceder y transferir archivos de manera segura.

2.2.9. *Técnicas Modernas de Cifrado de Archivos*

Según (IETF8446, 2018) existen técnicas modernas de cifrado de archivos entre ellas tenemos:

2.2.9.1. Cifrado por Bloques:

Divide los datos en bloques de tamaño fijo para cifrarlos. AES es un ejemplo destacado.

2.2.9.2. Cifrado por Flujo:

Cifra los datos bit a bit, generando un flujo de claves pseudoaleatorias. ChaCha20 es un algoritmo popular en este ámbito.

2.2.9.3. Cifrado Híbrido:

Combina cifrado simétrico y asimétrico para aprovechar las ventajas de ambos. Por ejemplo, se usa RSA para intercambiar una clave AES que cifra el archivo.

2.2.10. *Errores más frecuentes en el cifrado*

2.2.10.1. Utilización de protocolos obsoletos

Se considera que un protocolo de cifrado es obsoleto cuando la protección es vulnerable, un protocolo puede ser obsoleto si el mismo tienen errores. Según (IETF, 2015) RC4 también conocido como ARC4 o ARCFOUR tiene vulnerabilidades, especialmente cuando se utiliza en

determinadas configuraciones. Puede ser susceptible a ataques como el ataque Fluhrer-Mantin-Shamir (FMS)[8] y el ataque Klein.

En el cifrado de flujo, el término "flujo de claves" se define como una secuencia pseudoaleatoria de bits que se cifran con la secuencia de bits del texto plano mediante la operación XOR. El texto cifrado producido a partir del cifrado de flujo comprende una secuencia de bits con una longitud igual a la del texto plano. Sin embargo, puede haber varios cifradores de flujo.

RC4. **-** es la corriente más popular que depende de la operación clásica (XOR) con dos estados: simplemente (0, 1) que tiene varios puntos débiles, como ser simple, donde puede ser descifrado fácilmente por los atacantes. Además, existe un solapamiento problemático entre las salidas del estado interno conocidas públicamente. Estos puntos débiles han sido explotados por los atacantes para descubrir la clave y recuperar los datos.

> Considerado inseguro en la negociación de las claves de sesión, lo que podría llevar a un ataque "man-in-the-middle"(IETF, 2015).

IDEA. - El algoritmo internacional de cifrado de datos Este es un algoritmo de clave simétrica, utiliza la misma clave para cifrar y descifrar. El proceso de descifrado es el mismo que el de cifrado, salvo que las subclaves se obtienen utilizando un algoritmo diferente. El tamaño de la clave de cifrado es de 128 bits. En todo el proceso de cifrado utilizamos un total de 52 claves (ronda1 a ronda8 y fase de transformación de salida); generadas a partir de una clave de cifrado de 128 bits. En cada ronda (ronda1 a ronda8) utilizamos seis subclaves. Cada subclave consta de 16 bits. Y la transformación de salida utiliza 4 subclaves. (Forprodat, 2020)

> El hecho de limitarse a 128 bits lo vuelve vulnerable al hackeo.

[8] FMS permiten la obtención de la clave de cifrado utilizando para ello ataques estadísticos

TLS. - Transport Layer Security cifra y descifra datos utilizando una clave secreta que tanto el remitente como el destinatario conocen; generalmente tiene una longitud de 128 bits, pero preferiblemente 256 bits, ya que cualquier cosa menor a 80 bits ahora se considera insegura. En términos de cálculo, TLS funciona bien, pero compartir una clave secreta común significa que debe hacerse de forma segura, presentando ahí la debilidad de TLS (Ramió Jorge, 2020). Desde febrero de 2015, su uso está prohibido en TLS (IETF, 2015).

Kerberos v.4.- Errores en el manejo de identidades durante las sesiones de usuario, esto se debe a que, en lugar de usar contraseñas, utiliza los *hashes* de estas para validar la identidad en los tickets de sesión. El uso incorrecto de mecanismos de verificación utilizando *hashes* de sesión, como el ataque "Pass the Hash", puede llevar a la suplantación de identidades. (Avelino, 2021)

2.2.10.2. Utilización de claves muy cortas

La utilización de claves cortas se debe a que el algoritmo de encriptación es robusto, si la clave es demasiado pequeña la misma se vuelve vulnerable a los ataques de búsqueda (motores de búsqueda). Cuando se utilizan claves criptográficas predeterminadas, se generan o reutilizan claves criptográficas débiles, o no hay gestión de cambio de claves con determinada frecuencia, es probable que el algoritmo no tenga errores, pero al ser fácilmente descifrables se vuelve vulnerable. (OWASP, 2021)

DES. - Es un cifrado simétrico, lo que significa que utiliza la misma clave secreta para encriptar y desencriptar datos. La misma clave se puede usar una y otra vez. Esto no es factible en otros algoritmos más elementales, como el de la "libreta de un solo uso", es posible usar claves pequeñas. Por otro lado, otros algoritmos requieren que la clave sea del mismo tamaño

que el mensaje, lo que resulta inútil para transmitir información, es fácil de ejecutar en computadoras. (Tornil, 2018)

RSA tamaño mínimo a octubre 2023 es de 3095

2.2.11. *Errores en los mecanismos de compartición de credenciales*

Muchos de los datos son revelados a cualquier persona que tenga acceso al canal de comunicación al momento de aplicar directamente protocolos de revelación de atributos, esto se da a razón que los verificadores no están intrínsecamente autenticados, se pueden revelar atributos a la parte equivocada de manera accidental. Si las credenciales no son compartidas de forma criptográfica podrían caer en manos de un tercero (*man in the midle*). Además, las funciones de olvido de contraseña o recuperación de credenciales son débiles o ineficaces y no se pueden usar de manera segura, como se resume en la ***Tabla 6***

- ***Error asociado a la implementación parcial***

Se refiere a los riesgos y vulnerabilidades que surgen cuando solo se cifra una parte de los datos sensibles, dejando otra parte sin protección. Este enfoque incompleto puede comprometer la seguridad general del sistema y exponer la información a posibles ataques.

- ***No garantizar la autenticidad de los usuarios externos.***

Estamos conectados con un tercero, pero no estamos totalmente seguros si es la persona que dice ser o con quien necesitamos establecer conexión, no tener autenticados los extremos

- ***No proteger criptográficamente los datos en reposo.***

En la mayoría de las ocasiones los datos en las estaciones de trabajo no tienen protección criptográfica, violando el principio de cero confianza, cero *trust*

- ***Utilización de aplicaciones criptográficas con vulnerabilidades.***

Aplicaciones vulnerables

Solución. Mantener actualizadas las aplicaciones con los últimos parches disponibles

- ***Utilización de configuraciones débiles (no optimas)***

Utilizar claves compartidas, preconfiguradas con *psk* compartido lo que lo vuelve débil, especialmente en entornos empresariales

- ***No proteger correctamente las claves de cifrado***

Las claves de cifrado deben gestionarse, almacenarse y protegerse de manera adecuada, o algún atacante podría acceder a ellas y acceder a la información.

- ***Ocupar VPN ssl en lugar de ocupar VPN Ipsec***

Una VPN que empieza a cifrar en la capa de la aplicación únicamente cifra los datos más no las cabeceras que se van agregando en cada una de las fases del modelo OSI no son suficientes en entornos donde la seguridad es muy crítica

Solución VPN Ipsec es más complejo, pero a su vez más segura.

Tabla 6

Errores comunes en los mecanismos de compartición de credenciales

Errores	**Descripción del Error**	**Soluciones Propuestas**
Implementación parcial	Puede dejar brechas de seguridad al no cubrir todos los aspectos necesarios	Realizar una implementación completa y probada, siguiendo estándares de seguridad reconocidos
No garantizar la autenticidad de usuarios externos	Riesgo de acceso no autorizado o suplantación de identidad.	Implementar autenticación multifactor (MFA) y verificación de identidad robusta
No proteger criptográficamente los datos en reposo	Exposición de datos sensibles en caso de acceso no autorizado.	Utilizar algoritmos de cifrado fuertes (AES-256) para

		proteger los datos almacenados.
Utilización de aplicaciones criptográficas con vulnerabilidades	Explotación de vulnerabilidades conocidas, comprometiendo la seguridad.	Actualizar y parchear aplicaciones criptográficas regularmente, y usar solo software confiable.
Utilización de configuraciones débiles (no óptimas)	Mayor susceptibilidad a ataques y brechas de seguridad.	Aplicar configuraciones robustas basadas en mejores prácticas y estándares de seguridad.
No proteger correctamente las claves de cifrado	Pérdida o robo de claves, lo que compromete toda la seguridad del sistema.	Almacenar claves en hardware seguro (HSM) y gestionarlas con políticas estrictas.
Ocupar VPN SSL en lugar de VPN IPsec	Menor seguridad y rendimiento en comparación con IPsec	Migrar a VPN IPsec, que ofrece mayor seguridad y eficiencia en la transmisión de datos

Nota. Adaptación de errores y soluciones sugeridas por la autora

2.2.12. *¿Cómo se pueden corregir los errores en la implementación?*

Para corregir los errores en la implementación de los métodos de controles criptográficos, se pueden seguir diversas estrategias y prácticas recomendadas que abordan las vulnerabilidades más comunes, como se observa en la ***Tabla 7***, Por ejemplo:

- ***Uso de Algoritmos Criptográficos Actualizados***

Se debe utilizar algoritmos que sean reconocidos por su seguridad y que estén actualizados. Por ejemplo, se debe evitar el uso de algoritmos obsoletos como MD5 y SHA-1, y

optar por estándares más robustos como AES (*Advanced Encryption Standard*) y SHA-256. Esto garantiza que la criptografía utilizada sea resistente a ataques modernos.

- ***Gestión Segura de Claves***

Implementar prácticas estrictas para la generación, almacenamiento y rotación de claves criptográficas. Las claves deben ser generadas utilizando fuentes de entropía adecuadas y almacenadas en módulos de seguridad de *hardware* (HSM) o sistemas seguros para prevenir el acceso no autorizado

- ***Validación Rigurosa de Certificados***

Es crucial validar correctamente los certificados digitales y las cadenas de confianza antes de establecer conexiones seguras. Esto ayuda a prevenir ataques de suplantación (*man-in-the-middle*) que pueden comprometer la confidencialidad e integridad de los datos transmitidos

- ***Implementación Correcta de Protocolos***

Los protocolos criptográficos deben ser implementados exactamente como se especifica en sus estándares. Cualquier error en la implementación puede introducir vulnerabilidades significativas, como se evidenció en incidentes como Heartbleed (filtración de memoria) [9] y POODLE (*downgrade attack en SSL* 3.0) [10]

- ***Capacitación Continua del Personal***

Capacitar frecuentemente a los desarrolladores sobre las mejores prácticas en criptografía es vital para reducir errores en la implementación. Esto incluye educación sobre riesgos asociados y actualizaciones sobre nuevas vulnerabilidades y soluciones

[9] Heartbleed: Una vulnerabilidad en la implementación de OpenSSL que permitió a los atacantes leer la memoria del servidor, exponiendo datos sensibles.

[10] POODLE: Un ataque que explota una vulnerabilidad en SSL 3.0, permitiendo a los atacantes descifrar información sensible.

- ***Pruebas de Seguridad Periódicas***

Realizar auditorías regulares y pruebas de penetración ayuda a identificar y corregir fallos en los sistemas criptográficos antes de que puedan ser explotados por atacantes. Estas pruebas deben incluir evaluaciones del código fuente y configuraciones del sistema

- ***Uso de Librerías Criptográficas Confiables***

Utilizar librerías bien documentadas y ampliamente adoptadas puede minimizar errores en la implementación. Estas librerías suelen incluir funciones predefinidas que han sido probadas y auditadas por la comunidad.

Al seguir estas estrategias, las organizaciones pueden mejorar significativamente la seguridad de sus controles criptográficos y reducir el riesgo de brechas de seguridad causadas por errores en la implementación.

Tabla 7

Errores y soluciones más frecuentes en los controles criptográficos

Errores Comunes en Controles Criptográficos	**Descripción del Error**	**Soluciones Propuestas**
Uso de algoritmos débiles u obsoletos	La implementación de algoritmos criptográficos que han sido comprometidos o que son considerados inseguros, como MD5 o SHA1.	Adoptar algoritmos modernos y seguros, como AES o SHA-256, que sean reconocidos por su robustez y resistencia a ataques.
Mala gestión de claves criptográficas	Como la reutilización de claves, claves débiles, falta de rotación de claves y almacenamiento inseguro (por ejemplo, en el código fuente).	Implementar políticas estrictas para la generación, almacenamiento y rotación de claves. Utilizar un gestor de claves seguro.

Transmisión de datos en texto plano	Enviar datos sensibles sin cifrado a través de protocolos inseguros (por ejemplo, HTTP en lugar de HTTPS).	Utilizar siempre protocolos seguros como HTTPS, SFTP o FTPS para la transmisión de datos.
Fallo en la validación de certificados	No validar correctamente los certificados digitales puede permitir ataques de suplantación.	Implementar validaciones rigurosas para los certificados y las cadenas de confianza antes de establecer conexiones seguras.
Ausencia de cifrado autenticado	Utilizar solo cifrado sin autenticación puede comprometer la integridad y autenticidad de los datos.	Adoptar métodos de cifrado autenticado que proporcionen tanto confidencialidad como integridad, como AES-GCM.
Uso inadecuado del modo de operación	Aplicar modos inseguros como ECB que no ocultan patrones en los datos cifrados.	Utilizar modos seguros como CBC (Cipher Block Chaining) o GCM (Galois/Counter Mode) que ofrecen mejor seguridad.
No realizar pruebas de seguridad periódicas	La falta de pruebas regulares puede dejar vulnerabilidades sin detectar.	Realizar pruebas de penetración y auditorías regulares para identificar y corregir debilidades en el sistema criptográfico.

Nota. Adaptación de errores y soluciones sugeridas por la autora

2.2.13. *Casos de Estudio que presentaron fallas en Controles Criptográficos*

a) Vulnerabilidad en la Implementación de TLS en una Operadora Europea (2021)

En 2021, una operadora de telecomunicaciones europea experimentó una brecha de seguridad debido a una implementación deficiente del protocolo TLS (Transport Layer Security). La empresa utilizaba versiones obsoletas de TLS (TLS 1.0 y 1.1), que ya no se consideraban seguras. Además, se descubrió que los certificados digitales no se renovaron a tiempo, lo que permitió a los atacantes interceptar comunicaciones sensibles entre los usuarios y la red. (ENISA, 2021)

Consecuencias:

Interceptación de Datos: Los atacantes pudieron interceptar llamadas VoIP y mensajes de texto.

Pérdida de Confianza: La empresa enfrentó una caída significativa en la confianza de sus clientes.

Multas Regulatorias: La violación del Reglamento General de Protección de Datos (GDPR) resultó en una multa de 2 millones de euros.

b) Ataque a una Empresa de Telecomunicaciones en Asia (2022)

Una empresa de telecomunicaciones asiática sufrió un ataque de ransomware en 2022 debido a una mala gestión de las claves criptográficas. Los atacantes explotaron una vulnerabilidad en el sistema de gestión de claves, lo que les permitió cifrar los datos críticos de la empresa y exigir un rescate. (Verizon, 2023)

Consecuencias:

Interrupción del Servicio: Más de 10 millones de usuarios se vieron afectados por la interrupción del servicio durante varios días.

Pérdidas Financieras: La empresa pagó un rescate de 5 millones de dólares y enfrentó pérdidas adicionales por la interrupción del negocio.

Daño Reputacional: La empresa perdió varios contratos gubernamentales debido a la falta de confianza en su seguridad.

c) Falla en la Encriptación de Datos en una Empresa Latinoamericana (2023)

En 2023, una empresa de telecomunicaciones latinoamericana reportó una brecha de seguridad debido a una falla en la encriptación de datos almacenados en sus servidores. La empresa utilizaba un algoritmo de encriptación débil (DES) para proteger los datos de los usuarios, lo que permitió a los atacantes descifrar la información fácilmente. (ITU, 2024)

Consecuencias:

***Exposición de Datos*:** Los datos personales de más de 1 millón de usuarios fueron expuestos, incluyendo nombres, direcciones y números de teléfono.

***Demandas Legales*:** La empresa enfrentó múltiples demandas colectivas por violación de la privacidad.

***Sanciones Regulatorias*:** La autoridad de protección de datos impuso una multa de 1.5 millones de dólares.

d) Vulnerabilidad en Apache Log4j (2021)

Falla identificada: Uso de controles criptográficos insuficientes en la validación de entradas.

Consecuencias: La vulnerabilidad *Log4Shell* permitió a los atacantes ejecutar código remoto en sistemas vulnerables. Miles de aplicaciones y servicios en todo el mundo se vieron comprometidos, incluyendo servicios críticos en la nube.

Lección aprendida: La falta de validación criptográfica de las entradas permitió la explotación de la vulnerabilidad. Se recomienda implementar validación robusta y actualizaciones regulares. (Apache Software Foundation, 2021)

e) Ataque a Kaseya VSA (2021)

Falla identificada: Uso de cifrado débil en la comunicación entre servidores y clientes.

Consecuencias: Los atacantes explotaron una vulnerabilidad en el software de gestión remota *Kaseya VSA* para desplegar ransomware en más de 1,500 empresas. El ataque afectó a cadenas de suministro en todo el mundo.

Lección aprendida: La falta de cifrado fuerte en las comunicaciones permitió la interceptación y explotación de datos. Se recomienda utilizar protocolos de comunicación seguros como *TLS 1.3. (Kaseya, 2021)*

f) Vulnerabilidad en OpenSSL (2022)

Falla identificada: Uso de una implementación criptográfica vulnerable en la biblioteca OpenSSL.

Consecuencias: La vulnerabilidad *CVE-2022-3602* permitió a los atacantes ejecutar código arbitrario en sistemas que utilizaban versiones vulnerables de OpenSSL. Aunque no se reportaron explotaciones masivas, el riesgo potencial fue significativo.

Lección aprendida: La falta de actualización de bibliotecas criptográficas puede exponer a los sistemas a vulnerabilidades críticas. Se recomienda mantener las bibliotecas actualizadas y monitorear alertas de seguridad. (OpenSSL, 2022)

2.3. Marco legal

2.3.1. *Constitución de la República del Ecuador*

La Constitución de la República del Ecuador, es la Norma Suprema, a la que está sometida toda la legislación ecuatoriana, donde se establecen las normas fundamentales que amparan los derechos, libertades y obligaciones de todos los ciudadanos, así como las del estado y las instituciones que la conforman.

La Constitución del Ecuador reconoce y garantiza en el artículo 66 numeral 19 a las personas: "El derecho a la protección de datos carácter personal, que incluye el acceso y la decisión sobre información y datos de este carácter, así como su correspondiente protección. La recolección, archivo, procesamiento, distribución o difusión de estos datos personales requerirán la autorización del titular o el mandato de ley" (Constitución de la República del Ecuador, 2008).

2.3.2. *National Institute of Standards and Technology (NIST)*

Es una autoridad reconocida a nivel internacional en el desarrollo de estándares y guías para la ciberseguridad. En el contexto del cifrado de archivos, el NIST ha establecido estándares fundamentales como el Advanced Encryption Standard (AES) y el Secure Hash Algorithm (SHA), que garantizan la confidencialidad, integridad y autenticidad de los datos. Además, publicaciones como el NIST SP 800-57 y el NIST SP 800-175B proporcionan recomendaciones detalladas para la gestión de claves criptográficas y la implementación de estándares criptográficos. Estas guías son esenciales para el diseño de sistemas seguros y el cumplimiento de normativas de protección de datos.

2.3.3. *Ley Orgánica de Protección de Datos en el Ecuador*

Promulgada en el año 2021, esta ley establece un marco legal para la protección de datos personales en Ecuador. Las empresas deben obtener el consentimiento explícito de los usuarios

para el tratamiento de sus datos, implementar medidas de seguridad adecuadas y notificar brechas de seguridad a la autoridad correspondiente y a los afectados. La LOPD también exige la designación de un delegado de protección de datos (DPD) en ciertas circunstancias, así como la realización de evaluaciones de impacto sobre la privacidad (Russell Bedford, 2021).

2.3.4. *Ley Orgánica de Telecomunicaciones*

La protección de datos personales según (Copo, 2022), es una oportunidad para lograr mejores resultados en las organizaciones, mejorando la calidad de la información y estrategias de fidelización de los clientes. El uso adecuado y legítimo de los datos personales se traducirá en beneficios para la organización.

En el ámbito internacional la ley de protección de datos permite la realización de transferencia de datos a países, organizaciones y personas jurídicas de ámbito internacional siempre que se cumpla niveles correctos de protección de datos y que se ajusten a la obligación con cumplimiento de garantías y estándares reconocidos a nivel internacional. De igual manera en la ley se establecerán acciones conjuntas entre las autoridades de los países para prevenir, mitigar o suspender el uso indebido de los datos entre ambos países (Espada, 2021).

Esta ley permite la transferencia de datos siempre y cuando se cumplan con las garantías adecuadas como:

- Garantizar el cumplimiento de principios, derechos y obligaciones en el tratamiento de datos personales en un estándar igual o mayor a la normativa.
- Derecho integral en reparación y,
- Efectiva tutela del derecho a la protección de datos.

2.3.5. *Estrategia Nacional de Ciberseguridad*

Desde este 3 de agosto de 2022, Ecuador cuenta, por primera vez, con su Estrategia Nacional de Ciberseguridad (ENC), que permitirá a los ciudadanos acceder a servicios digitales con mayor seguridad y fortalecer la protección de sus datos personales. Además, abre nuevas opciones para generar regulación a fin de proteger a todos los actores de la sociedad de la ciberdelincuencia y fortalece las infraestructuras tecnológicas de las entidades públicas y privadas.

2.3.6. *Ley Orgánica de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos*

Esta ley reconoce los mensajes de datos como equivalentes a documentos escritos, lo que proporciona un respaldo jurídico a las comunicaciones digitales y a la transferencia de archivos, esto significa que los archivos transferidos electrónicamente tienen el mismo valor legal que aquellos en formato físico, lo que es crucial para las empresas que operan en el ámbito digital.

Uno de los aspectos más relevantes de esta ley es la regulación de las firmas electrónicas, que son fundamentales para autenticar la identidad del remitente y asegurar la integridad del mensaje. Al utilizar firmas electrónicas, las empresas pueden garantizar que los archivos no han sido alterados durante su transferencia, lo cual es esencial para mantener la confianza en las transacciones comerciales.

Además, la ley establece principios de confidencialidad y reserva, obligando a las entidades a proteger la información durante su transmisión y a evitar la divulgación no autorizada. Esta ley también impone responsabilidades a las entidades de certificación, asegurando que se mantenga la protección de datos personales y se cumpla con los estándares de seguridad en el manejo de información sensible. Esto incluye la obligación de notificar a los usuarios sobre cualquier riesgo asociado al uso indebido de firmas electrónicas o certificados, lo que refuerza aún más la seguridad en la transferencia de archivos.

En resumen, la Ley Orgánica de Comercio Electrónico establece un marco normativo que no solo valida las transacciones electrónicas, sino que también promueve prácticas seguras en la transferencia de archivos, protegiendo tanto a las empresas como a los consumidores en el entorno digital.

2.3.7. *Normas Técnicas Ecuatorianas NTE INEN-ISO/IEC 27000*

Estas normas proporcionan un marco para la gestión de la seguridad de la información, donde incluyen directrices sobre cómo establecer, implementar, mantener y mejorar un Sistema de Gestión de Seguridad de la Información (SGSI). Las empresas deben seguir estas normas para proteger sus activos de información y asegurar su disponibilidad, integridad y confidencialidad (CELEC EP, 2014).

CAPITULO III

MARCO METODOLÓGICO

3.1. Descripción del área de estudio / Descripción del grupo de estudio

El presente proyecto de investigación se ejecutó en la empresa de telecomunicaciones Servimatango C.A ubicada en Ecuador, provincia de Imbabura, cantón Ibarra, barrio Yacucalle, se encuentra ubicada en la calle Zenon Villacis 4-42

Figura 14.

Mapa de ubicación de Servimatango C.A

Nota. Tomado de *Google Maps*. (2025). https://goo.gl/maps/dQk42WVXMLD5hEyF7

3.2. Enfoque de Investigación

La investigación en Servimatango seguirá un enfoque mixto, combinando métodos cualitativos y cuantitativos para comprender la situación actual, identificar riesgos y proponer soluciones efectivas. El enfoque es descriptivo y aplicado, ya que busca describir la situación actual de la empresa y aplicar soluciones prácticas para mejorar la seguridad de los archivos.

3.2.1. *Enfoque Mixto*

El enfoque mixto en la investigación combina estrategias de los enfoques cualitativo y cuantitativo para obtener una comprensión más completa de un fenómeno. Esta integración permite complementar las fortalezas de cada método y mitigar sus debilidades. (Vizcaíno Paulina et al., 2023)

3.2.1.1. Características del Enfoque Mixto

- Complementariedad: Usa métodos cualitativos para profundizar en experiencias o significados y métodos cuantitativos para medir y analizar datos numéricos.

- Flexibilidad: Permite ajustar el diseño de la investigación según las necesidades del estudio.

- ***Triangulación****:* Mejora la validez del estudio al cruzar información obtenida por distintos métodos.

3.2.1.2. Tipos de Diseño Mixto

- ***Diseño Convergente Paralelo:*** Se recogen y analizan datos cualitativos y cuantitativos al mismo tiempo, luego se integran los resultados.
- ***Diseño Secuencial Explicativo:*** Primero se recogen datos cuantitativos, luego se amplían con información cualitativa.

- ***Diseño Secuencial Exploratorio:*** Primero se recopilan datos cualitativos para explorar el fenómeno y luego se complementan con análisis cuantitativos.

3.2.1.3. Situación Actual de la Empresa

Se utilizó la entrevista con los diferentes miembros de la empresa para recopilar datos cuantitativos sobre el cifrado y compartición de archivos, así como obtener un panorama inicial del problema, con estos datos se propone la solución más adecuada. *Anexo 1*

3.2.2. *Enfoque Descriptivo*

Según el artículo de (Ochoa-Pachas, 2020) la investigación descriptiva:

- Caracteriza un fenómeno o situación concreta, identificando sus rasgos distintivos.

- Responde preguntas como "¿Qué es?", "¿Cómo es?", "¿Dónde está?" y "¿Cuánto?".

- Permite conocer situaciones, costumbres y actitudes predominantes.

- Se basa en la recolección de datos que pueden expresarse de forma cualitativa (símbolos verbales) o cuantitativa (símbolos matemáticos).

- Puede incluir encuestas, estudios de interrelaciones (casos, comparativos y correlacionales) y estudios de desarrollo (crecimiento y tendencia).

3.2.3. *Enfoque Aplicado*

En el mismo estudio de (Ochoa-Pachas, 2020) se menciona que este enfoque:

- Busca resolver un problema específico: la falta de cifrado en la transferencia de archivos en Servimatango.

- Propone la implementación de herramientas como VeraCrypt y 7-Zip para cifrar archivos antes de enviarlos a proveedores y clientes. Igualmente usar OpenVPN en caso de conexiones remotas entre personal de la empresa.

3.3. Tipo de Investigación

La investigación es de tipo exploratoria y experimental

3.3.1. *Investigación Exploratoria*

Se realiza para comprender el problema y explorar posibles soluciones. Incluye encuestas a los empleados, entrevistas, revisión de políticas de seguridad y análisis de los flujos de trabajo actuales. (Reyes, 2022)

La investigación exploratoria es aquella que se realiza cuando hay poca información sobre un tema. Su propósito es proporcionar un panorama inicial del problema, identificar patrones y generar hipótesis para estudios más detallados. (Vizcaíno Paulina et al., 2023)

3.3.1.1. Características de la Investigación Exploratoria

- ***No busca establecer causalidades:*** Solo pretende conocer mejor el problema o fenómeno. Anexo 2

- ***Utiliza fuentes diversas****:* Revisión de literatura, entrevistas, estudios de casos y observaciones.

- ***Es flexible y abierta:*** Puede ajustar su dirección conforme se obtiene nueva información.

3.3.1.2. Análisis de la encuesta

a) Cuando el personal recibe archivos de proveedores los almacena en el computador asignado por la empresa, mismo que cuenta únicamente con la protección de contraseña de acceso.

b) La información de los clientes igualmente es almacenada en los computadores personales de los empleados.

c) Los medios más frecuentes para enviar información a clientes y proveedores son los emails y Whatsapp.

d) Únicamente el 30% de los empleados comprime los archivos o carpetas antes de enviarlos, el 70% restante envía los archivos tal como los tienen. Únicamente dos empleados han escuchado del cifrado de archivos.

e) La información que los empleados tienen en sus equipos es vulnerable a cualquier persona que tenga acceso a un computador desbloqueado o tenga su contraseña, igualmente si se coloca el disco de un empleado como disco esclavo en otro equipo, se podría acceder a esa información sin problemas.

f) Actualmente la empresa no tiene una política que exija el cifrado en los archivos que se envían y reciben.

3.3.2. *Investigación Experimental*

La investigación experimental es un método riguroso en el que se manipulan variables independientes para observar su efecto en una variable dependiente. Su principal objetivo es establecer relaciones de causa y efecto. (Vizcaíno Paulina et al., 2023)

3.3.2.1. Características de la investigación experimental

- Control de variables: Se aíslan factores externos para garantizar que los cambios en la variable dependiente sean causados por la manipulación de la variable independiente.

- Uso de grupos de estudio: Se divide a los participantes en grupo experimental (recibe el tratamiento) y grupo de control (no recibe tratamiento).

- Reproducibilidad: Los experimentos pueden replicarse para comprobar la validez de los resultados.

3.3.2.2. Ejemplo de uso

Luego de optar por encriptar los discos con VeraCrypt únicamente a la mitad de los miembros de la empresa se les pidió lo usarán siempre y la otra mitad no era necesario. Luego se analizó el resultado de ambos grupos para determinar si la técnica de encriptado tuvo un impacto significativo.

3.4. Procedimiento de investigación

3.4.1. *Selección de Técnicas Criptográficas*

Evaluar y seleccionar las técnicas criptográficas más adecuadas para implementar en el contexto específico de la empresa. Esto puede incluir algoritmos como AES, RSA, y protocolos como SFTP y FTPS, analizando sus ventajas y desventajas en términos de seguridad y rendimiento.

3.4.2. *Análisis de Información Recabada*

En Servimatango, luego de analizar la encuesta realizada al personal de la empresa como se muestra en el *Anexo 1* se identificó que los archivos sensibles, como contratos, datos de clientes y especificaciones técnicas, se transfieren a proveedores y clientes sin cifrado, la información se guarda en los computadores sin protección, esto representa un riesgo significativo para la confidencialidad e integridad de la información, ya que los archivos podrían ser interceptados o modificados durante la transferencia. Además, la falta de cifrado incumple normativas de protección de datos como indica LPDP y expone a la empresa a posibles sanciones y daños reputacionales. *Anexo 2*

3.4.3. *Recolección de información*

Con la información obtenida en la situación actual se procedió a recolectar información acerca de técnicas criptográficas, errores y debilidades de estas, así como métodos de implementación y verificación de integridad, confidencialidad y autenticidad.

3.4.4. *Comparación de Metodologías de Auditoría*

A continuación, se comparan tres metodologías ampliamente utilizadas para auditar controles criptográficos:

3.4.4.1. OWASP Cryptographic Storage Cheat Sheet

Enfoque: Seguridad en aplicaciones web.

Ventajas:

Proporciona directrices específicas para el almacenamiento seguro de datos.

Fácil de implementar en entornos de desarrollo.

Desventajas:

Limitado a aplicaciones web y no cubre otros sistemas.

Adecuado para: Empresas con aplicaciones web que manejan datos sensibles.

3.4.4.2. NIST SP 800-57 (Recomendación para la Gestión de Claves Criptográficas)

Enfoque: Gestión de claves criptográficas.

Ventajas:

Estándar reconocido internacionalmente.

Cubre todo el ciclo de vida de las claves (generación, almacenamiento, uso, rotación y destrucción).

Desventajas:

Puede ser complejo de implementar en organizaciones pequeñas.

Adecuado para: Empresas que requieren cumplimiento con normativas gubernamentales o de alto nivel de seguridad.

3.4.4.3. ISO/IEC 27001 (Gestión de la Seguridad de la Información)

Enfoque: Gestión integral de la seguridad de la información.

Ventajas:

Enfoque holístico que incluye controles criptográficos como parte de un sistema de gestión de seguridad.

Certificación reconocida a nivel mundial.

Desventajas:

Requiere una implementación extensa y costosa.

Adecuado para: Organizaciones que buscan una certificación internacional y un enfoque integral de seguridad.

Tabla 8

Resumen de la comparación

Metodología	Enfoque	Ventajas	Desventajas	Adecuado para
OWASP	Aplicaciones web	Directrices específicas, fácil de implementar	Limitado a aplicaciones web	Empresas con aplicaciones web que manejan datos sensibles
NIST SP 800-57	Gestión de claves criptográficas	Estándar reconocido, cubre todo el ciclo de vida de las claves	Complejo para organizaciones pequeñas	Empresas que requieren cumplimiento con normativas gubernamentales
ISO/IEC 27001	Gestión integral de seguridad	Enfoque holístico, certificación reconocida	Implementación extensa y costosa	Organizaciones que buscan una certificación internacional y un enfoque integral

Nota. Adaptación de la comparación de las metodologías de auditoría

3.4.5. *Implementación*

3.4.5.1. Verificación de la integridad

Verificar la integridad de los archivos es un proceso crucial para asegurar que no han sido alterados, corrompidos o modificados de manera no autorizada. A continuación, se detallan los métodos más comunes para realizar esta verificación:

a) Uso de Funciones Hash

Las funciones hash son algoritmos que generan una cadena de caracteres única (hash) a partir de un archivo. Si el archivo se modifica, el hash cambia, lo que permite detectar alteraciones.

- ***Pasos para Verificar la Integridad con Hash:***
  - Generar el Hash Original:
  - Almacenar el Hash de Manera Segura:
  - Verificar el Hash Posteriormente:
  - Otras Herramientas Recomendadas:

OpenSSL: Para generar hashes en múltiples sistemas operativos.

HashCalc: Herramienta gráfica para calcular hashes en Windows.

Integrity Checker: Extensiones o software específico para verificar hashes en tiempo real.

b) Uso de Firmas Digitales

Las firmas digitales no solo verifican la integridad, sino también la autenticidad del archivo (es decir, quién lo firmó).

- ***Pasos para Verificar la Integridad con Firmas Digitales:***
  - Firmar el Archivo:
  - Distribuir la Clave Pública:
  - Verificar la Firma:
  - Herramientas Recomendadas:

GnuPG (GPG): Herramienta de código abierto para firmar y verificar archivos.

Adobe Sign: Para documentos PDF.

Microsoft Office: Incluye funcionalidades de firma digital.

Gpg4win: Kleopatra

a) Uso de Checksums

Los checksums son valores numéricos que se calculan a partir de los datos de un archivo. Son similares a los hashes, pero suelen ser más simples y menos seguros.

- ***Pasos para Verificar la Integridad con Checksums:***
  - Generar el Checksum:
  - Comparar el Checksum:
  - Herramientas Recomendadas:

FCIV (File Checksum Integrity Verifier): Herramienta de Microsoft para Windows.

cksum: Comando integrado en sistemas Unix/Linux.

b) Uso de Herramientas de Auditoría de Integridad de Archivos

Existen herramientas especializadas que monitorean y verifican la integridad de los archivos en tiempo real.

- ***Herramientas Recomendadas:***

Tripwire: Herramienta de código abierto que monitorea cambios en archivos y directorios.

AIDE (Advanced Intrusion Detection Environment): Similar a Tripwire, pero con funcionalidades adicionales.

OSSEC: Sistema de detección de intrusiones que incluye verificación de integridad de archivos.

c) Comparación de Archivos

Si tienes una copia original del archivo, puedes compararla directamente con la copia que deseas verificar.

- ***Pasos para Comparar Archivos:***
  - Usar Herramientas de Comparación:
  - Verificar Diferencias

3.4.5.2. Verificación de la Confidencialidad

Verificar la confidencialidad de los archivos implica asegurar de que solo las personas o sistemas autorizados pueden acceder al contenido. A continuación, se detallan los métodos más comunes para realizar esta verificación:

a) Cifrado de Archivos

El cifrado es la técnica más efectiva para garantizar la confidencialidad. Si un archivo está cifrado, solo quienes tengan la clave correcta podrán acceder a su contenido.

- ***Pasos para Verificar la Confidencialidad mediante Cifrado:***
  - Cifrar el Archivo:
  - Verificar el Cifrado:
  - Gestionar las Claves de Cifrado:
  - Herramientas Recomendadas:

VeraCrypt: Para cifrado de archivos y volúmenes.

7-Zip: Permite cifrar archivos comprimidos con AES-256.

OpenSSL: Para cifrado de archivos mediante comandos.

b) Control de Acceso

El control de acceso garantiza que solo los usuarios o sistemas autorizados puedan acceder a los archivos.

- ***Pasos para Verificar la Confidencialidad mediante Control de Acceso:***
    - Verificar permisos
    - Auditar los Permisos:
    - Implementar Autenticación y Autorización:

Herramientas Recomendadas:

Active Directory (Windows): Para gestión centralizada de permisos.

SELinux (Linux): Para políticas de acceso avanzadas.

AWS IAM o Azure AD: Para control de acceso en la nube.

Splunk: Para análisis de registros y monitoreo.

Wazuh: Para detección de intrusiones y monitoreo de archivos.

OSSEC: Para auditoría de integridad y detección de cambios.

c) Pruebas de Penetración

Realizamos pruebas de penetración para identificar vulnerabilidades que puedan comprometer la confidencialidad.

- ***Pasos para Verificar la Confidencialidad mediante Pentesting:***
    - Identificar Puntos Débiles:
    - Simular Ataques:
    - Corregir Vulnerabilidades:
    - Herramientas Recomendadas:

Nmap: Para escaneo de redes.

Metasploit: Para pruebas de penetración.

Burp Suite: Para pruebas en aplicaciones web.

d) Uso de Firmas Digitales

Las firmas digitales no solo verifican la confidencialidad, sino también la autenticidad del archivo (es decir, quién lo firmó), su fortaleza en la confidencialidad está en la distribución de la clave publica y manejo de la clave privada

- ***Pasos para Verificar la Confidencialidad con Firmas Digitales:***
  - Firmar el Archivo:
  - Distribuir la Clave Pública:
  - Verificar la Firma:
  - Herramientas Recomendadas:

GnuPG (GPG): Herramienta de código abierto para firmar y verificar archivos.

Adobe Sign: Para documentos PDF.

Microsoft Office: Incluye funcionalidades de firma digital.

Gpg4win: Kleopatra

3.4.5.3. Verificación de la Autenticidad

La autenticidad de los archivos se refiere a la capacidad de asegurar que un archivo proviene de una fuente confiable y que no ha sido alterado desde su creación.

a) Autenticidad con Uso de Firmas Digitales

Pasos para Verificar la Autenticidad con Firmas Digitales:

- Firmar el Archivo:
- Distribuir la Clave Pública:
- Verificar la Firma:
- Herramientas Recomendadas:

- **GnuPG (GPG):** Herramienta de código abierto para firmar y verificar archivos.

- **Adobe Sign:** Para documentos PDF.

- **Microsoft Office:** Incluye funcionalidades de firma digital.

- **Gpg4win:** Kleopatra

b) **Autenticidad con Uso de Funciones Hash**

Las funciones hash permiten verificar la integridad de un archivo, lo que indirectamente ayuda a confirmar su autenticidad. Si el hash del archivo coincide con el hash proporcionado por la fuente confiable, el archivo no ha sido alterado.

- ***Pasos para Verificar la Autenticidad con Hash:***

  - Generar el Hash Original:

  - Comparar el Hash:

  - Herramientas Recomendadas:

- OpenSSL: Para generar hashes en múltiples sistemas operativos.

- HashCalc: Herramienta gráfica para calcular hashes en Windows.

- Integrity Checker: Extensiones o software específico para verificar hashes en tiempo real.

c) ***Autenticidad con Uso de Certificados Digitales***

Los certificados digitales son emitidos por autoridades de certificación (CA) y vinculan una clave pública con una identidad. Pueden usarse para verificar la autenticidad de archivos firmados digitalmente.

- ***Pasos para Verificar la Autenticidad con Certificados Digitales:***

  - Firmar el Archivo con un Certificado:

  - Verificar el Certificado:

- Verificar la Firma:
- Herramientas Recomendadas:

- Adobe Acrobat: Para verificar firmas en documentos PDF.

- Microsoft Office: Para verificar firmas en documentos de Office.

- OpenSSL: Para verificar certificados y firmas en archivos.

d) Autenticidad con Uso de Blockchain

La tecnología blockchain puede utilizarse para verificar la autenticidad de archivos almacenando hashes o firmas digitales en una cadena de bloques inmutable.

- ***Pasos para Verificar la Autenticidad con Blockchain:***
  - Registrar el Hash en Blockchain:
  - Verificar el Hash:
  - Herramientas Recomendadas:

- Stampery: Servicio para registrar hashes en blockchain.

- Blockchain Notary: Herramientas para notarización de archivos en blockchain.

3.4.6. *Monitoreo y Evaluación*

Cada dos meses se realiza pruebas de penetración para identificar vulnerabilidades que puedan comprometer la confidencialidad.

Con el Hash se verifica la integridad y autenticidad

3.4.7. *Consideraciones bioéticas*

La investigación se llevó a cabo con los permisos respectivos de la empresa Servimatango. Asimismo, se informó oportunamente a los empleados sobre los aspectos clave del estudio, permitiéndoles comprender la relevancia de su participación, así como sus derechos,

deberes y beneficios asociados. Además, se garantizó y respetó el anonimato de todos los participantes e involucrados.

CAPITULO IV

RESULTADOS

4.1. Evaluación del rendimiento de algoritmos de cifrado simétrico.

En un estudio comparativo según (Centellas et al., 2022), de los algoritmos de cifrado simétrico AES, 3DES y ChaCha20 revela resultados significativos sobre su rendimiento en términos de tiempo de ejecución, consumo de CPU y eficiencia energética. Se determinó que ChaCha20 es el algoritmo más rápido, logrando un rendimiento superior en comparación con AES y 3DES. En particular, ChaCha20 mostró una mejora del 300% en velocidad respecto a 3DES, que es considerablemente más lento en la ejecución de operaciones de cifrado y descifrado. Por otro lado, AES se destacó por su bajo consumo de CPU, siendo el algoritmo que menos recursos computacionales requiere durante su funcionamiento. En términos porcentuales, AES consumió aproximadamente un 25% menos CPU que ChaCha20 en los escenarios evaluados. Sin embargo, no se encontraron diferencias significativas en el consumo energético entre los tres algoritmos, lo que sugiere que, aunque ChaCha20 es más rápido, su eficiencia energética es comparable a la de sus contrapartes. A continuación, se presenta un resumen visual de los resultados obtenidos:

Tabla 9

Resultados del Estudio Comparativo de los Algoritmos AES, 3DES y ChaCha20

Algoritmo	**Tiempo de Cifrado (ms)**	**Consumo de CPU (%)**	**Consumo Energético (J)**
AES	100	30	0.5
3DES	300	40	0.5
ChaCha20	75	35	0.5

Nota. Adaptado del estudio de (Centellas et al., 2022)

4.2. Evaluación del rendimiento de algoritmos de cifrado asimétrico.

El objetivo principal del estudio realizado por (Yifeng et al., 2021) fue evaluar el rendimiento de tres algoritmos de cifrado asimétrico: RSA, ECC. DSA. Se llevaron a cabo simulaciones en Java para medir la velocidad de cifrado, velocidad de cifrado, el rendimiento y la utilización de CPU con archivos de diferentes tamaños (1 MB a 1 GB).

Los resultados mostraron variaciones significativas en el rendimiento entre los algoritmos analizados.

* **RSA**: El tiempo de cifrado y descifrado aumenta con el tamaño de la clave. Por ejemplo, para una clave de 2048 bits, el tiempo de cifrado (firma) es de aproximadamente 5.69 ms, y el de descifrado (verificación) es de 0.18 ms.

* **DSA:** El tiempo de firma con una clave de 2048 bits es de aproximadamente 4.6 ms.

* **ECC:** Ofrece tiempos más rápidos; por ejemplo, con una clave de 128 bits, el tiempo de firma es de aproximadamente 0.47 ms y el de verificación es de 0.79 ms.

Tabla 10

Resultados de la comparación de algoritmos cifrado asimétrico

Algoritmo	Tamaño de Clave	Tiempo de Cifrado	Tiempo de Descifrado	Velocidad (MB/s)	Uso de CPU (%)
RSA	1024 bits	~1.08 ms (firma)	~0.09 ms (verificación)	N/A	Alto
RSA	2048 bits	~5.69 ms (firma)	~0.18 ms (verificación)	N/A	Alto
RSA	3192 bits	~19.67 ms (firma)	~0.38 ms (verificación)	N/A	Alto
DSA	2048 bits	~4.6 ms (firma)	N/A	N/A	Medio

ECC	128 bits	~0.47 ms (firma)	~0.79 ms (verificación)	N/A	Bajo

Nota. Adaptado del estudio de (Yifeng et al., 2021)

4.3. Evaluación del rendimiento de algoritmos de cifrado simétrico y asimétrico

Estudios de benchmarks criptográficos de supercop (Bench, 2024) entre algoritmos simétricos y asimétricos demuestran que a medida que aumenta el tamaño de la clave, generalmente se incrementa la seguridad, pero también se requieren más recursos computacionales. ECC ofrece una mejor relación entre tamaño de clave y seguridad en comparación con RSA y ChaCha20.

Uso de CPU y consumo energético: Algoritmos con claves más largas y operaciones matemáticas más complejas (como RSA y AES) tienden a consumir más CPU y energía. ECC, con claves más cortas y operaciones eficientes, es más adecuado para entornos con recursos limitados.

Aplicaciones prácticas: ECC se está convirtiendo en el estándar para muchas aplicaciones modernas debido a su eficiencia y seguridad. RSA sigue siendo ampliamente utilizado, pero su eficiencia disminuye con claves más largas. DSA se utiliza principalmente para firmas digitales.

Tabla 11

Tabla Comparativa entre métodos simétricos y asimétricos

Algoritmo	**Tipo**	**Velocidad (MB/s)**	**Tamaño de Clave**	**Uso de CPU**	**Seguridad**
AES-256	Simétrico	500	256 bits	Alto	Alta
ChaCha20	Simétrico	700	256 bits	Medio	Alta
RSA-2048	Asimétrico	10	2048 bits	Alto	Alta
ECC	Asimétrico	50	256 bits	Medio	Alta

SHA-256	Hash	1000	256 bits	Bajo	Resistente a colisiones

Nota: Adatado de los estudios de benchmarks criptográficos de supercop, (https://bench.cr.yp.to/supercop.html)(Bench, 2024)

4.4. Evaluación del Impacto de diferentes métodos criptográficos en términos de velocidad, eficacia y eficiencia.

Para realizar esta evaluación se realizaron pruebas en Python, se utilizó la biblioteca cryptography para implementar los algoritmos y se procedió a medir los tiempos de ejecución para cifrar y descifrar archivos de diferentes tamaños.

- Se instaló las librerías necesarias: cryptography, psutil, pycryptodome
- Generación de archivos de prueba de diferentes tamaños (1MB, 10MB, 100MB)

criptografia_test.py 1

.vscode > criptografia_test.py > ...

```python
def create_test_file(file_path, size_mb):
    with open(file_path, "wb") as f:
        f.write(os.urandom(size_mb * 1024 * 1024))  # Genera datos aleatorios

# Crear archivos de prueba
create_test_file("test_file_1MB.bin", 1)
create_test_file("test_file_10MB.bin", 10)
create_test_file("test_file_100MB.bin", 100)
```

- Se implementaron funciones para cifrar y descifrar usando AES (simétrico) y RSA (asimétrico). Código en *Anexo 4*

```python
# Generar una clave y un IV para AES
def generate_aes_key():
    return os.urandom(32)  # Clave de 256 bits

def generate_aes_iv():
    return os.urandom(16)  # IV de 128 bits

```

- Luego se probó los algoritmos de prueba y se midió el tiempo de ejecución, así como el uso de recursos. Código en Anexo 4

```python
if __name__ == "__main__":
    # Generar claves
    aes_key = generate_aes_key()
    aes_iv = generate_aes_iv()
    rsa_private_key, rsa_public_key = generate_rsa_keys()

    # Archivos de prueba
    test_files = ["test_file_1MB.bin", "test_file_10MB.bin", "test_file_100MB.bin"]

    # Probar cada archivo
    for file_path in test_files:
        print(f"\nProbando archivo: {file_path}")
```

- Se obtuvo los siguientes resultados.

Tabla 12

Resultados de la prueba en Python AES

	1MB	**10MB**	**100MB**
Tiempo de cifrado	0,0180 s	0,1589 s	1,3102 s
Tiempo de descifrado	0,0260 s	0,1189 s	1,2182 s
Uso del CPU	16,1 %	12.9 %	3,5 %
Uso de Memoria	7039,23 MB	7063,41 MB	7208,57 MB

Nota. Adaptado de los resultados obtenidos en la prueba

- El cifrado con RSA directo para archivos con un tamaño superior a 1MB tomó demasiado tiempo y se detuvo; por lo cual RSA no es recomendado para cifrar grandes volúmenes de datos.
- Para AES + RSA se utilizó el código Anexo 5

  AES - Cifrado: 0,0310 s

Descifrado: 0,0310 s

RSA - Cifrado Clave AES: 0,0030 s

Descifrado Clave AES: 0,0120 s

- Se realizo un análisis del enfoque híbrido (AES + RSA), que es la mejor solución en sistemas reales.

Tabla 13

Resultados de la evaluación AES vs. AES+RSA

Tamaño	**Cifrado AES (s)**	**Descifrado AES (s)**	**Cifrado AES + RSA (s)**	**Descifrado AES + RSA (s)**
1MB	0,0055	0,0049	0,0008	0,0021
10MB	0,0521	0,0582	0,0009	0,0018
100MB	0,4856	0,4982	0,0010	0,0029

Nota. Adaptado de las pruebas realizadas en Python

De la Evaluación podemos concluir que RSA directo es inviable para archivos grandes, AES es el más veloz.

El enfoque híbrido (AES + RSA) es eficiente, ya que solo cifra la clave AES con RSA, lo que toma menos de 0,003s en todos los casos.

4.5. Verificación de Integridad, Confidencialidad y Autenticidad de los Archivos

Luego de los estudios y pruebas realizadas se optó por usar el hash como método de verificación de integridad y autenticidad

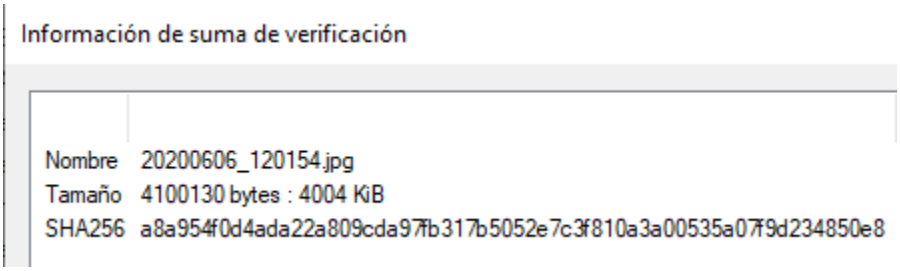
4.5.1. *Verificación de Integridad y Autenticidad*

a) Generar el Hash Original:

Cuando el archivo está en su estado original, generamos el hash con la herramienta CRC SHA de 7-zip eligiendo el SHA-256

Figura 15

Obtener Hash con 7-zip



- Obtenemos el hash: SHA256:

a8a954f0d4ada22a809cda97fb317b5052e7c3f810a3a00535a07f9d234850e8

b) Almacenar el Hash de Manera Segura:

Guardamos el hash en un lugar seguro, separado del archivo (en un documento protegido o en un sistema de gestión de claves).

c) Verificar el Hash Posteriormente:

Cuando necesitamos verificar la integridad, se vuelve a generar el hash del archivo y se compara con el hash original.

Si los hashes coinciden, el archivo no ha sido modificado. Si no coinciden, el archivo ha sido alterado.

4.5.2. *Verificación de la Confidencialidad*

Verificar la confidencialidad de los archivos implica asegurar de que solo las personas o sistemas autorizados pueden acceder al contenido.

4.5.2.1. Cifrado de Archivos

El cifrado es la técnica más efectiva para garantizar la confidencialidad. Si un archivo está cifrado, solo quienes tengan la clave correcta podrán acceder a su contenido.

Pasos para Verificar la Confidencialidad mediante Cifrado:

a) Cifrar el Archivo:

Usamos algoritmos de cifrado robustos como AES-256 de la herramienta 7-zip

b) Verificar el Cifrado:

Nos aseguramos de que el archivo no pueda ser leído sin la clave correcta.

Intentamos abrir el archivo sin la clave para confirmar que está cifrado, de igual forma se realizaron pruebas de penetración.

c) Gestionar las Claves de Cifrado:

Almacenamos las claves en un lugar seguro, como un HSM (Hardware Security Module) o un gestor de contraseñas.

d) Herramientas Recomendadas:

VeraCrypt: Para cifrado de archivos y volúmenes.

7-Zip: Permite cifrar archivos comprimidos con AES-256.

Kleopatra: permite cifrar archivos con firma digital.

4.5.2.2.Control de Acceso

El control de acceso garantiza que solo los usuarios o sistemas autorizados puedan acceder a los archivos.

Se usa VeraCrypt para cifrar los discos del personal en la empresa

4.5.2.3.Ejemplo con 7zip

* Damos clic derecho sobre el archivo a cifrar y elegimos la opción de 7 zip agregar al archivo comprimido.

* En la parte derecha de cifrado colocamos una contraseña de por lo menos 8 caracteres, de preferencia que tenga números, letras, símbolos.

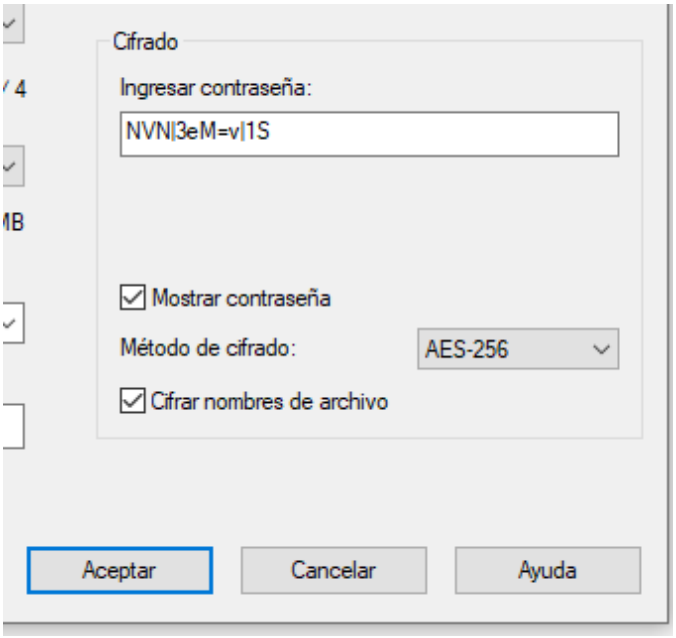
* mostrar contraseña es solo por ejemplo

* clic en cifrar nombres de archivo

* elegir el cifrado AES-256

Figura 16

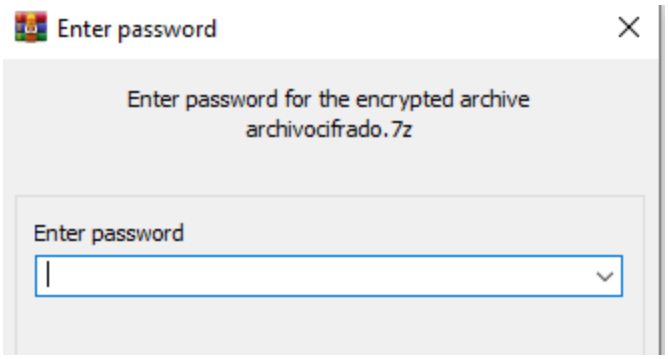
Cifrado con 7-zip paso1



Solicita contraseña para acceder al archivo

Figura 17

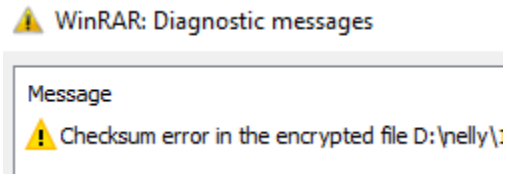
archivo comprimido con contraseña



Si colocamos la contraseña incorrecta nos indica un error de chucksum

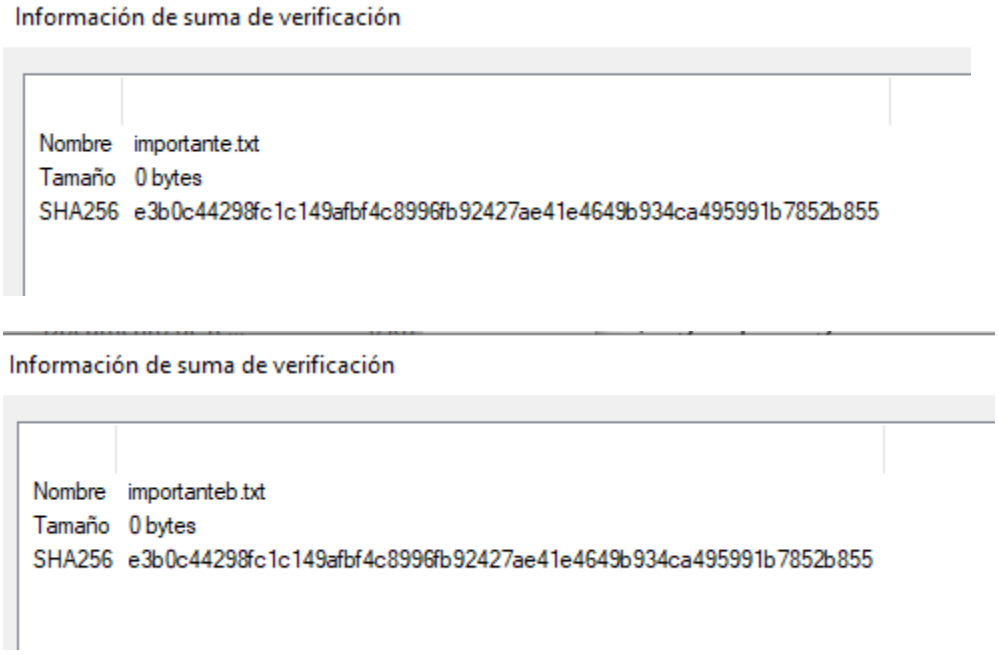
Figura 18

Error al colocar contraseña errónea



Si, colocamos la contraseña adecuada se descargará el archivo

Verificamos que el sha sea el mismo del archivo original con el que resulta luego de descomprimir



El archivo original importante como el archivo descomprimido importanteb tienen el mismo SHA256

4.5.2.4.Pruebas de Penetración

Realizamos pruebas de penetración para identificar vulnerabilidades que puedan comprometer la confidencialidad.

Pasos para Verificar la Confidencialidad mediante Pentesting:

Identificar Puntos Débiles:

Usa herramientas como Nmap o Metasploit para escanear sistemas y archivos.

Simular Ataques:

Intenta acceder a archivos sin autorización para probar las defensas.

Corregir Vulnerabilidades:

Implementa medidas correctivas basadas en los hallazgos.

Herramientas Recomendadas:

Nmap: Para escaneo de redes.

Metasploit: Para pruebas de penetración.

Burp Suite: Para pruebas en aplicaciones web.

4.5.3. *Transferencia de archivos seguros entre equipos remotos*

Para la transferencia en forma remota se optó por usar OpenVPN

4.5.4. *Pruebas de Python para la verificación*

Para verificar la integridad, confidencialidad y autenticidad de los archivos se lo realizó mediante pruebas en Python, revisar código en *Anexo 7*

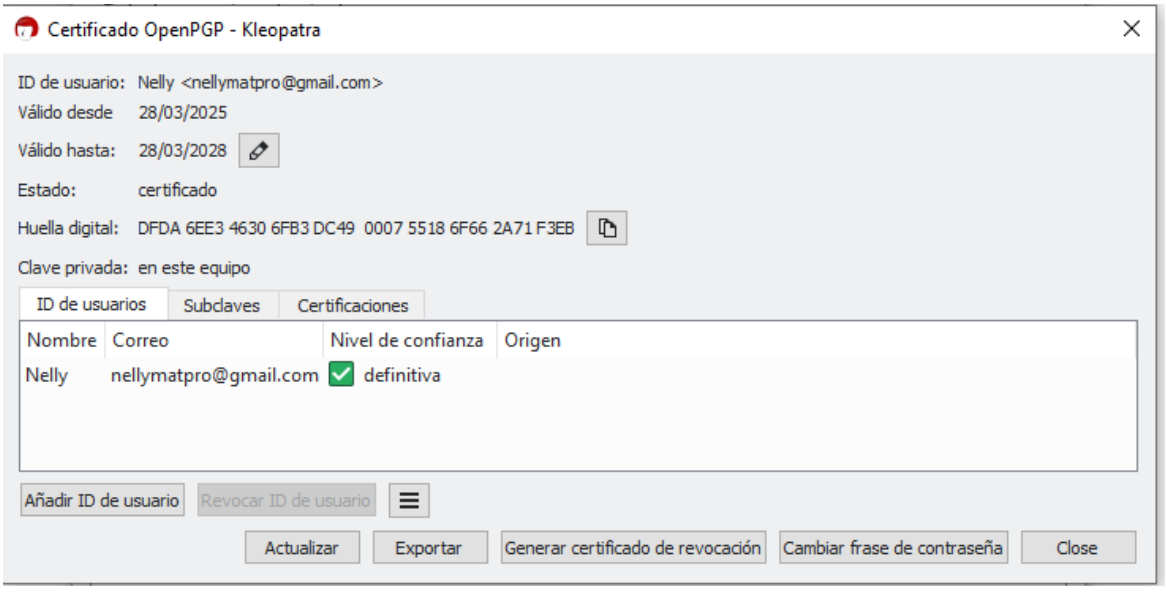
4.6. Verificación de Integridad, Confidencialidad y Autenticidad de los Archivos mediante Firma Digital

Para la verificación con firma digital se tomó como base la “Guía de Instalación, Configuración y uso del Software Kleopatra para el Cifrado y Descifrado Asimétrico De Directorios y Archivos” de (Quintero & Pacheco, 2021) la cual indica los siguientes pasos:

- Descargar e instala Gpg4win (Kleopatra)
- Genera un par de claves (pública y privada)
- Cifrar un archivo, generando un archivo (`.gpg`)
- Firmar digitalmente un archivo, se generará un archivo `.sig` o un `.asc` (firma separada) o un archivo firmado (`.gpg`).
- Verificar y descifrar un archivo, si se tienes la clave privada correcta, te pedirá la contraseña y descifrará el contenido.
- Para verificar una firma, seleccionamos el archivo firmado y hacemos clic en "Verificar".

Figura 19

Verificacion CIA con Firma Digital - Kleopatra



4.7. Discusión

Una vez analizadas varias técnicas criptográficas, métodos de cifrado y descifrado se recomienda a la empresa, crear una política de cifrado de archivos la cual incluya el almacenar la información en volúmenes cifrados con VeraCrypt, de necesitar información en el escritorio la misma puede estar cifrada con Kleopatra, toda la información tanto de clientes, proveedores y de giro del negocio,

Por rapidez y facilidad se recomienda usar el 7-zip para cifrar los archivos antes de enviarlos a proveedores y clientes, adjuntando la tabla de coordenadas del *Anexo 3*

El cifrado de archivos es vital en las empresas, en especial las de telecomunicaciones que manejan datos sensibles.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

Luego del análisis de la presente investigación se llegó a la conclusión de lo siguiente:

a) La gestión adecuada de las claves criptográficas es esencial para garantizar la seguridad de los datos.

b) Las funciones hash son una herramienta esencial en la seguridad de la información, especialmente en la verificación de la integridad y autenticidad de los archivos. Aunque no cifran datos directamente, su uso complementario con técnicas de cifrado garantiza una protección robusta contra alteraciones y accesos no autorizados.

c) Se puede decir que los algoritmos cifrados en el tiempo llegan a funcionar con eficacia, pero se debe mantener una investigación continua de nuevos algoritmos que brinden una mayor seguridad en el intento de descifrado por fuerza bruta, ya que todo esto depende de la capacidad de cómputo existente en la actualidad.

d) Así como va avanzando la tecnología, los ataques también se van fortaleciendo, es así que en la actualidad no existen algoritmos criptográficos que tengan una máxima seguridad, pero la aplicación del criptográfico AES su nivel de seguridad es 1.8 veces mejor en seguridad que los demás algoritmos implementados en el software.

e) Es fundamental la utilización de herramientas que monitoreen las redes contra ataques a sistemas informáticos, estos ayudan a encontrar vulnerabilidades, estándares de ataque e información de los mismos, logrando así establecer políticas de seguridad que permitan minimizar el hurto de información o ataques a cualquier red de la empresa.

5.2. RECOMENDACIONES

Como recomendaciones luego de realizar la presente investigación se presentan las siguientes:

a) Se recomienda reforzar la gestión de claves, asegurándose de que las claves utilizadas en AES y RSA se almacenen de manera segura, preferiblemente en un HSM (Hardware Security Module) o un gestor de claves como Hashicorp Vault o AWS KMS.

b) Capacitar al personal sobre el uso correcto de Veracrypt, 7-zip, Kleopatra y OpenVPN, considerando que muchas brechas de seguridad ocurren por errores humanos, como el uso incorrecto de herramientas o la exposición accidental de claves.

c) Fomentar una cultura de seguridad en la empresa, destacando la importancia de cifrar la información sensible.

d) Realizar auditorías periódicas para verificar que las herramientas de cifrado se estén utilizando correctamente y que no haya vulnerabilidades en su implementación.

e) Implementa un sistema de monitoreo para detectar actividades sospechosas en la red, especialmente en conexiones externas a través de OpenVPN.

f) Si la empresa utiliza dispositivos móviles o portátiles, asegúrate de que también estén cifrados (por ejemplo, con BitLocker para Windows o FileVault para macOS).

g) Considerar implementar soluciones de DLP (Data Loss Prevention) para evitar fugas de información.

h) Es fundamental considerar a la criptografía como una herramienta eficiente y a su vez importante para asegurar la información, por lo que se deben realizar estudios e investigaciones para poder elegir un algoritmo adecuado para la empresa de acuerdo a su funcionamiento, estructura y características.

i) Documentar las políticas de cifrado y seguridad de la información, incluyendo cómo y cuándo se deben utilizar Veracrypt, 7-zip, Kleopatra y OpenVPN, estableciendo protocolos claros para el intercambio seguro de información con clientes y proveedores.

j) Establecer protocolos de seguridad como el de respaldos diarios de la información más importante, ya sea a través de procesos automatizados o a su vez el de capacitar al personal sobre la importancia de respaldar la información para garantizar una seguridad proactiva.

k) Realizar pruebas periódicas de recuperación de datos cifrados para asegurar de que, en caso de emergencia, la información pueda ser descifrada y accesible sin problemas.

REFERENCIAS

Apache Software Foundation. (2021). *Apache Logging Services*. https://logging.apache.org/log4j

Avelino, M. (2021). *Buenos algoritmos con malas implementaciones*.

AWS. (2022). *¿Qué es el almacenamiento en la nube?* . https://aws.amazon.com/es/what-is/cloud-storage/

Barker, E. (2020). *Recommendation for key management:* https://doi.org/10.6028/NIST.SP.800-57pt1r5

Bench. (2024). *Introduction*. https://bench.cr.yp.to/

CELEC EP. (2014). *Normativa Técnica Ecuatoriana NTE INEN-ISO/IEC 27000*. https://www.celec.gob.ec/images/lotaip/2015/Comun/a3)SIC_NTE_001_2014.pdf

Centellas, L. S., Blanco, L., & Sandoval, J. P. (2022). Estudio comparativo de los algoritmos de criptografía simétrica AES, 3DES y ChaCha20 Comparative study of the symmetric cryptography algorithms AES, 3DES and ChaCha20. In *ACTA NOVA* (Vol. 10). http://www.scielo.org.bo/pdf/ran/v10n3/1683-0789-ran-10-03-283.pdf

Cyberearmag. (2024). *Principales riesgos de ciberseguridad en la industria de las telecomunicaciones - Cyber War Mag*. https://cyberwarmag.com/principales-riesgos-ciberseguridad-industria-telecomunicaciones/

ENISA. (2021). *ENISA Report - NIS Investments 2021*. https://doi.org/10.2824/77127

Forprodat. (2020). *Cifrado de archivos y protección de datos - FORPRODAT*. https://forprodatcyl.es/cifrado-de-archivos-y-proteccion-de-datos/

Gupta, C. P., & Goyal, K. K. (2020). *Cybersecurity : a self-teaching introduction*.

Huang, H., Zhu, P., Xiao, F., Sun, X., & Huang, Q. (2020). A blockchain-based scheme for privacy-preserving and secure sharing of medical data. *Computers & Security*, *99*, 102010. https://doi.org/10.1016/J.COSE.2020.102010

IBM. (2021). *¿Qué es la transferencia de archivos? | IBM*. https://www.ibm.com/es-es/topics/file-transfer

IETF. (2015). *RFC 7465 - Prohibiting RC4 Cipher Suites*. Https://Datatracker.Ietf.Org/Doc/Rfc7465/.

IETF8446. (2018). *RFC 8446 - The Transport Layer Security (TLS) Protocol Version 1.3*. https://datatracker.ietf.org/doc/html/rfc8446

International IT. (2022, September 8). *¿Cuáles son los principales tipos de cifrado para la transferencia segura de archivos?* https://www.internationalit.com/post/cu%C3%A1les-son-los-principales-tipos-de-cifrado-para-la-transferencia-segura-de-archivos?lang=es

ISO/IEC. (2018). *ISO/IEC 10118-3:2018(en), IT Security techniques — Hash-functions — Part 3: Dedicated hash-functions*. https://www.iso.org/obp/ui/en/#iso:std:iso-iec:10118:-3:ed-4:v1:en

IT Ahora, D. (2024). *Presentación 5to. Informe Estado Actual de la Ciberseguridad Ecuador 2024: Sector privado.* https://itahora.com/2024/06/04/presentacion-5to-informe-estado-actual-de-la-ciberseguridad-ecuador-2024-sector-privado/

ITU. (2024). *ITUPublications International Telecommunication Union Global Cybersecurity Index 2024 5 th Edition Global Cybersecurity Index 2024 5th Edition Acknowledgements*.

Jena, B. (2023). *Digital Signature Algorithm (DSA) in Cryptography: How It Works & More*.

Kaseya. (2021). *Kaseya Responds Swiftly to Sophisticated Cyberattack - Kaseya*. https://www.kaseya.com/press-release/kaseya-responds-swiftly-to-sophisticated-cyberattack-mitigating-global-disruption-to-customers/

NIST. (2015). *Secure hash standard (SHS)*. https://doi.org/10.6028/NIST.FIPS.180-4

NIST. (2020). *Guide to IPsec VPNs*. https://doi.org/10.6028/NIST.SP.800-77r1

Ochoa-Pachas, J. (2020, October 26). *Vista de El estudio descriptivo en la investigación científica*. http://revistas.autonoma.edu.pe/index.php/AJP/article/view/224/191

OpenSSL. (2022). *OpenSSL Security Advisory*. https://openssl-library.org/news/secadv/20221101.txt

OWASP. (2021). *A02 Cryptographic Failures - OWASP Top 10:2021*. https://owasp.org/Top10/A02_2021-Cryptographic_Failures/

Plaza, F. (2021). *Manual de Criptografía. Fundamentos matemáticos de la Criptografía para un estudiante de Grado*.

Primicias EC. (2021, July 30). *Los misterios del ataque que dejó a CNT sumida en la "emergencia."* https://www.primicias.ec/noticias/tecnologia/los-misterios-del-ataque-que-dejo-a-cnt-sumida-en-emergencia/?utm_source=chatgpt.com

Quintero, M., & Pacheco, J. (2021). *GUÍA DE INSTALACIÓN, CONFIGURACIÓN Y USO DEL SOFTWARE KLEOPATRA PARA EL CIFRADO Y DESCIFRADO ASIMÉTRICO DE DIRECTORIOS Y ARCHIVOS*. https://www.seguridad.unam.mx/sites/default/files/guia-cifrado-asimetrico-con-kleopatra.pdf

Ramió Jorge. (2020). *Curso Criptografía Aplicada*.

Reyes, E. (2022). *Metodología de la Investigación Científica* (primera).

Russell Bedford. (2021). *Ley de Protección de Datos en Ecuador | Guía para Empresas - Russell Bedford EC*. https://russellbedford.com.ec/ley-de-proteccion-de-datos-en-ecuador-guia-para-empresas/

Samaniego, E. A., & Ponce, J. A. (2021). *Fundamentos de seguridad informática*.

Simplilearn.com. (2020, March 26). *What Is Data Encryption: Types, Algorithms, Techniques and Methods*. Https://Www.Simplilearn.Com/Data-Encryption-Methods-Article?Tag=encode.

Tornil, X. P. (2018). *Seguridad en redes WLAN*.

Veritas Technologies. (2024). *Safeguarding Your Sensitive Data.* Https://Www.Veritas.Com/Es/Mx/Information-Center/File-Encryption.

Verizon. (2023). *Data breach investigations report 2022*. https://espanol.verizon.com/business/resources/reports/2022-dbir-data-breach-investigations-report.pdf

Vizcaíno Paulina, Cedeño, R., & Maldonado, I. (2023). *Metodología de la investigación científica: guía práctica*. https://ciencialatina.org/index.php/cienciala/article/view/7658/11619

vpnMentor. (2019, September 17). *Base de datos expuso información personal de más de 20 millones de ecuatorianos*. https://www.vpnmentor.com/la-es/2019/09/17/base-datos-expuso-informacion-personal-millones-ecuatorianos/

WeLiveSecurity. (2024). *Cómo compartir archivos confidenciales en línea de forma segura*. https://www.welivesecurity.com/es/consejos-seguridad/como-compartir-archivos-forma-segura/

Yifeng, H., Nan, Y., & Rui, Z. (2021). Analysis of Data Encryption Algorithms for Telecommunication Network-Computer Network Communication Security. *Wireless Communications and Mobile Computing, 2021*. https://doi.org/10.1155/2021/2295130

ANEXOS

Anexo 1 Encuesta situación Actual

Por favor responda las siguientes preguntas de manera honesta,

*1. Cuando recibe un Archivo de un **proveedor** que contiene información personal o sensible ¿en qué lugar lo guarda?*

[ ] En documentos

[ ] En descargas

[ ] En el escritorio

[ ] En el disco alterno

[ ] En una carpeta segura

[ ] No sé en qué lugar se guarda el archivo

*2. Cuando recibe un Archivo de un **cliente** que contiene información personal o sensible ¿en qué lugar lo guarda?*

[ ] En documentos

[ ] En descargas

[ ] En el escritorio

[ ] En el disco alterno

[ ] En una carpeta segura

[ ] No sé en qué lugar se guarda el archivo

3. *¿Qué medios ocupa para enviar archivos a clientes y proveedores?*

[ ] Correo Electrónico (e-mail)

[ ] FTP

[ ] WhatsApp

[ ] Telegram

[ ] otro ----------

4. *¿Cifra de alguna forma los archivos qué comparte?*

[ ] Si

[ ] No

[ ] comprimo con contraseña

[ ] No sé qué es cifrar

5. *Si alguna persona accede a su estación de trabajo, ¿Podría acceder a toda la información que usted dispone en la misma?*

[ ] Si

[ ] No

6. ¿Los archivos recibidos por la empresa suelen venir cifrados por parte de terceros?

[ ] Sí, en todos los casos

[ ] Sí, en algunos casos

[ ] No

[ ] No lo sé

7. ¿Se utilizan canales de comunicación seguros para el intercambio de archivos (Ej. SFTP, VPN, TLS, HTTPS)?

[ ] Sí, siempre

[ ] Sí, pero no en todos los casos

[ ] No

[ ] No lo sé

8.¿Existe una política en la empresa que exija el uso de cifrado en los archivos que se envían y reciben?

[ ] Sí, y se cumple estrictamente

[ ] Sí, pero no se aplica en todos los casos

[ ] No, pero está en proceso de implementación

[ ] No

Anexo 2 Entrevista Administrador de TI

Entrevista con Administrador de Sistemas

1. Pregunta: *¿Consideras que las medidas actuales son efectivas para proteger la seguridad en la transferencia de archivos?*

R. la verdad no, creo que la información está expuesta a un ataque físico y los archivos que enviamos igualmente son vulnerables pues no se cifran.

2. Pregunta: *¿Qué recomendarías para proteger la seguridad de los archivos?*

R. que cifremos los archivos de información sensible de la empresa, y busquemos la forma de cifrar los archivos que recibimos y enviamos.

3. Pregunta: *¿Sabes si la empresa piensa implementar algún sistema por la aplicación de la LPDP?*

R. Si, se espera implementar las políticas correspondientes para salvaguardar datos sensibles que maneja la empresa

4. Pregunta: *¿Los empleados han recibido capacitación sobre la importancia del cifrado de archivos?*

No, se ha conversado levemente, pero nada en profundidad

4. Pregunta: *¿Se utilizan herramientas de cifrado recomendadas por la empresa o cada empleado elige su método?*

No, aun no se estableces las herramientas a utilizar

Anexo 3 Ejemplo de Coordenadas

D001

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	N	W	F	M	M	K	F	P	F	R	Z	I	S	L	I
2	3	5	6	8	1	0	3	8	4	1	6	3	8	4	4
3	e	s	h	k	f	s	j	t	x	s	s	l	e	j	i
4	?	&	-	#	[	]	=	.	]	.#	?	_	[	_	
5	Z	Z	K	T	P	G	Q	H	L	F	I	L	H	Q	T
6	h	h	o	m	c	c	p	q	i	h	x	t	s	k	q
7	A	H	4	p	I	&	$	#	#	0	5	#	_	P	W
8	1	9	1	1	9	5	8	0	6	4	7	0	9	8	5
9	h	h	k	f	I	o	q	h	c	g	g	k	w	I	z
10	W	P	E	U	E	Y	M	A	D	S	Q	Q	F	U	A
11	.	[	_	*	^	(	=	#	|	}	*	]	)	:	^
12	A	H	4	p	I	&	$	#	#	0	5	#	_	P	W
13	8	0	6	3	6	6	3	2	8	1	1	5	4	7	1
14	O	M	P	B	R	L	S	Q	X	Y	O	P	C	J	T
15	w	k	d	o	y	c	l	t	n	v	l	u	l	r	e

Anexo 4 tiempos de ejecución para cifrar y descifrar archivos de diferentes tamaños.

Código Fuente de Python

```
import os
import time
import psutil
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms,
modes
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.backends import default_backend

# Generar una clave y un IV para AES
def generate_aes_key():
    return os.urandom(32)  # Clave de 256 bits

def generate_aes_iv():
    return os.urandom(16)  # IV de 128 bits

# Generar un par de claves para RSA
def generate_rsa_keys():
```

```
            private_key = rsa.generate_private_key(
                public_exponent=65537,
                key_size=2048,
                backend=default_backend()
            )
            public_key = private_key.public_key()
            return private_key, public_key

        # Cifrar un archivo con AES-GCM
        def encrypt_file_aes(file_path, key, iv):
            cipher = Cipher(algorithms.AES(key), modes.GCM(iv),
backend=default_backend())
            encryptor = cipher.encryptor()
            with open(file_path, "rb") as f:
                data = f.read()
            start_time = time.time()
            encrypted_data = encryptor.update(data) + encryptor.finalize()
            tag = encryptor.tag  # Obtener el tag de autenticación
            end_time = time.time()
            return encrypted_data, tag, end_time - start_time

        # Descifrar un archivo con AES-GCM
        def decrypt_file_aes(encrypted_data, key, iv, tag):
            cipher = Cipher(algorithms.AES(key), modes.GCM(iv, tag),
backend=default_backend())
            decryptor = cipher.decryptor()
            start_time = time.time()
            decrypted_data = decryptor.update(encrypted_data) +
decryptor.finalize()
            end_time = time.time()
            return decrypted_data, end_time - start_time

        # Cifrar un archivo con RSA
        def encrypt_file_rsa(file_path, public_key):
            with open(file_path, "rb") as f:
                data = f.read()
            start_time = time.time()
            encrypted_data = public_key.encrypt(
```

```
            data,
            padding.OAEP(
                mgf=padding.MGF1(algorithm=hashes.SHA256()),
                algorithm=hashes.SHA256(),
                label=None
            )
        )
        end_time = time.time()
        return encrypted_data, end_time - start_time

    # Descifrar un archivo con RSA
    def decrypt_file_rsa(encrypted_data, private_key):
        start_time = time.time()
        decrypted_data = private_key.decrypt(
            encrypted_data,
            padding.OAEP(
                mgf=padding.MGF1(algorithm=hashes.SHA256()),
                algorithm=hashes.SHA256(),
                label=None
            )
        )
        end_time = time.time()
        return decrypted_data, end_time - start_time

    # Medir el uso de recursos
    def measure_resource_usage():
        cpu_usage = psutil.cpu_percent(interval=1)
        memory_usage = psutil.virtual_memory().used / (1024 * 1024)  # En
MB
        return cpu_usage, memory_usage

    # Crear archivos de prueba
    def create_test_file(file_path, size_mb):
        with open(file_path, "wb") as f:
            f.write(os.urandom(size_mb * 1024 * 1024))  # Genera datos
aleatorios

    # Programa principal
```

```
    if __name__ == "__main__":
        # Crear archivos de prueba
        create_test_file("test_file_1MB.bin", 1)
        create_test_file("test_file_10MB.bin", 10)
        create_test_file("test_file_100MB.bin", 100)

        # Generar claves
        aes_key = generate_aes_key()
        aes_iv = generate_aes_iv()
        rsa_private_key, rsa_public_key = generate_rsa_keys()

        # Archivos de prueba
        test_files = ["test_file_1MB.bin", "test_file_10MB.bin",
"test_file_100MB.bin"]

        # Probar cada archivo
        for file_path in test_files:
            print(f"\nProbando archivo: {file_path}")

            # AES
            encrypted_data, tag, aes_encrypt_time =
encrypt_file_aes(file_path, aes_key, aes_iv)
            decrypted_data, aes_decrypt_time =
decrypt_file_aes(encrypted_data, aes_key, aes_iv, tag)
            cpu_usage, memory_usage = measure_resource_usage()
            print(f"AES - Tiempo de cifrado: {aes_encrypt_time:.4f}
segundos")
            print(f"AES - Tiempo de descifrado: {aes_decrypt_time:.4f}
segundos")
            print(f"AES - Uso de CPU: {cpu_usage}%")
            print(f"AES - Uso de memoria: {memory_usage:.2f} MB")

            # RSA
            encrypted_data, rsa_encrypt_time = encrypt_file_rsa(file_path,
rsa_public_key)
            decrypted_data, rsa_decrypt_time =
decrypt_file_rsa(encrypted_data, rsa_private_key)
            cpu_usage, memory_usage = measure_resource_usage()
```

```
            print(f"RSA - Tiempo de cifrado: {rsa_encrypt_time:.4f}
segundos")
            print(f"RSA - Tiempo de descifrado: {rsa_decrypt_time:.4f}
segundos")
            print(f"RSA - Uso de CPU: {cpu_usage}%")
            print(f"RSA - Uso de memoria: {memory_usage:.2f} MB")
```

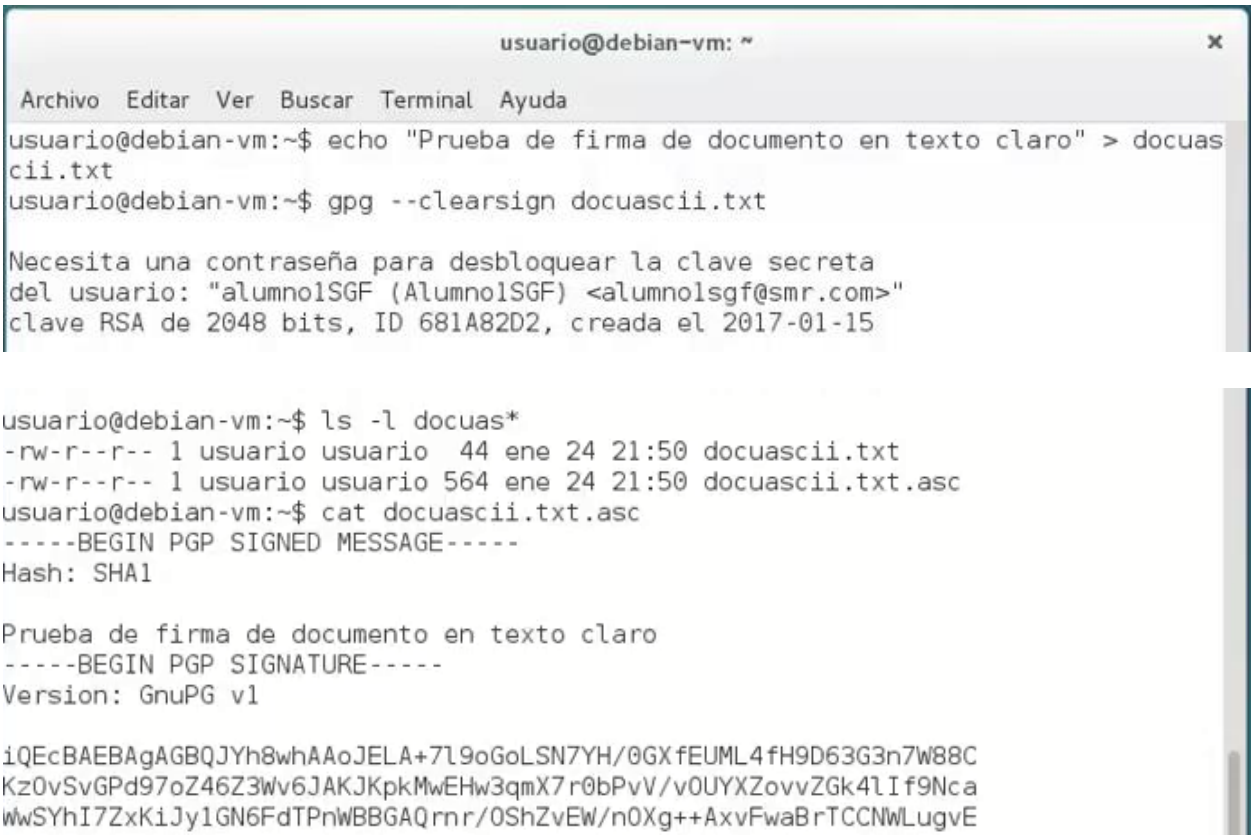
Anexo 5 Código Fuente de Python AES +RSA

```
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP, AES
from Crypto.Random import get_random_bytes
import time

def generate_rsa_keys():
    key = RSA.generate(2048)
    return key, key.publickey()

def encrypt_aes(data, aes_key):
    cipher = AES.new(aes_key, AES.MODE_GCM)
    ciphertext, tag = cipher.encrypt_and_digest(data)
    return cipher.nonce, ciphertext, tag

def decrypt_aes(nonce, ciphertext, tag, aes_key):
    cipher = AES.new(aes_key, AES.MODE_GCM, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext, tag)

def encrypt_aes_key(aes_key, rsa_public_key):
    cipher_rsa = PKCS1_OAEP.new(rsa_public_key)
    return cipher_rsa.encrypt(aes_key)

def decrypt_aes_key(encrypted_aes_key, rsa_private_key):
    cipher_rsa = PKCS1_OAEP.new(rsa_private_key)
    return cipher_rsa.decrypt(encrypted_aes_key)

def hybrid_encryption(data):
    rsa_private, rsa_public = generate_rsa_keys()
    aes_key = get_random_bytes(32)  # AES-256
```

```
            # Cifrado AES
            start = time.time()
            nonce, ciphertext, tag = encrypt_aes(data, aes_key)
            aes_enc_time = time.time() - start

            # Cifrado de clave AES con RSA
            start = time.time()
            encrypted_aes_key = encrypt_aes_key(aes_key, rsa_public)
            rsa_enc_time = time.time() - start

            return (rsa_private, nonce, ciphertext, tag, encrypted_aes_key,
aes_enc_time, rsa_enc_time)

      def hybrid_decryption(rsa_private, nonce, ciphertext, tag,
encrypted_aes_key):
            # Descifrado de clave AES con RSA
            start = time.time()
            aes_key = decrypt_aes_key(encrypted_aes_key, rsa_private)
            rsa_dec_time = time.time() - start

            # Descifrado AES
            start = time.time()
            decrypted_data = decrypt_aes(nonce, ciphertext, tag, aes_key)
            aes_dec_time = time.time() - start

            return decrypted_data, aes_dec_time, rsa_dec_time

      # --- Ejemplo de uso ---
      data = b"Mensaje de prueba" * 100000  # Simula un archivo grande
      rsa_private, nonce, ciphertext, tag, encrypted_aes_key, aes_enc_time,
rsa_enc_time = hybrid_encryption(data)
      decrypted_data, aes_dec_time, rsa_dec_time =
hybrid_decryption(rsa_private, nonce, ciphertext, tag, encrypted_aes_key)

      print(f"AES - Cifrado: {aes_enc_time:.4f} s | Descifrado:
{aes_dec_time:.4f} s")
```

```
      print(f"RSA - Cifrado Clave AES: {rsa_enc_time:.4f} s | Descifrado
Clave AES: {rsa_dec_time:.4f} s")
      print("Datos coinciden:", decrypted_data == data)
```

Anexo 6 Ejemplo con GnuPG



```
xMEUTzSI05ymojRxiFY/UuJ9riYcCRPOSvIwSdb7bBzvONhHLvvYtUTWAYuWRzRo
xr6vpCVq/axjp7bbOFWgmBYj/WZBMkdX++ZeYUSzPwSl06iEUVrrg+GFytR5VeaK
ZdeQY0zsSe7RDqJo1YjopalEx+8slgiesj/GHqNi8858b3IL0NGfhJ8H/ky3aZI=
=MXps
-----END PGP SIGNATURE-----
```

Canal CIFP Villa de Agüimes Agüimes. (20 de marzo de 2017). Firma digital de archivos con gpg [Archivo de Vídeo]. Youtube. https://www.youtube.com/watch?v=9gdXvAT2q-4

Anexo 7 Código Fuente de Python de verificación de integridad, confidencial y autenticidad.

```
import os
import hashlib
from cryptography.hazmat.primitives import hashes, serialization
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms,
modes
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives.asymmetric import utils
from cryptography.hazmat.primitives import hmac

# Generar una clave simétrica para cifrado AES
def generate_aes_key(password: bytes, salt: bytes) -> bytes:
    kdf = PBKDF2HMAC(
        algorithm=hashes.SHA256(),
        length=32,  # Clave de 256 bits
        salt=salt,
        iterations=100000,
        backend=default_backend()
    )
    return kdf.derive(password)

# Cifrar un archivo con AES
def encrypt_file(file_path: str, key: bytes, iv: bytes) -> bytes:
    cipher = Cipher(algorithms.AES(key), modes.CFB(iv),
backend=default_backend())
    encryptor = cipher.encryptor()
    with open(file_path, "rb") as f:
        data = f.read()
    encrypted_data = encryptor.update(data) + encryptor.finalize()
    return encrypted_data

# Descifrar un archivo con AES
def decrypt_file(encrypted_data: bytes, key: bytes, iv: bytes) ->
bytes:
```

```
        cipher = Cipher(algorithms.AES(key), modes.CFB(iv),
backend=default_backend())
        decryptor = cipher.decryptor()
        decrypted_data = decryptor.update(encrypted_data) +
decryptor.finalize()
        return decrypted_data

    # Verificar la integridad de un archivo usando SHA-256
    def verify_integrity(file_path: str, original_hash: str) -> bool:
        sha256_hash = hashlib.sha256()
        with open(file_path, "rb") as f:
            for chunk in iter(lambda: f.read(4096), b""):
                sha256_hash.update(chunk)
        return sha256_hash.hexdigest() == original_hash

    # Generar un par de claves RSA para firmas digitales
    def generate_rsa_keys():
        private_key = rsa.generate_private_key(
            public_exponent=65537,
            key_size=2048,
            backend=default_backend()
        )
        public_key = private_key.public_key()
        return private_key, public_key

    # Firmar un archivo con RSA
    def sign_file(file_path: str, private_key) -> bytes:
        with open(file_path, "rb") as f:
            data = f.read()
        signature = private_key.sign(
            data,
            padding.PSS(
                mgf=padding.MGF1(hashes.SHA256()),
                salt_length=padding.PSS.MAX_LENGTH
            ),
            hashes.SHA256()
        )
        return signature
```

```
    # Verificar la firma de un archivo con RSA
    def verify_signature(file_path: str, signature: bytes, public_key) ->
bool:
        with open(file_path, "rb") as f:
            data = f.read()
        try:
            public_key.verify(
                signature,
                data,
                padding.PSS(
                    mgf=padding.MGF1(hashes.SHA256()),
                    salt_length=padding.PSS.MAX_LENGTH
                ),
                hashes.SHA256()
            )
            return True
        except Exception:
            return False

    # Programa principal
    if __name__ == "__main__":
        # Archivo de prueba
        file_path = "archivo_vica.txt"
        with open(file_path, "w") as f:
            f.write("Este es un archivo de prueba para verificar
integridad, confidencialidad y autenticidad.")

        # 1. Verificar la integridad del archivo
        print("Verificando integridad del archivo...")
        original_hash = hashlib.sha256(open(file_path,
"rb").read()).hexdigest()
        print(f"Hash original: {original_hash}")
        is_integrity_ok = verify_integrity(file_path, original_hash)
        print(f"Integridad verificada: {'Éxito' if is_integrity_ok else
'Fallo'}")

        # 2. Cifrar y descifrar el archivo para confidencialidad
```

```
        print("\nCifrando archivo para confidencialidad...")
        password = b"my_password"
        salt = os.urandom(16)
        key = generate_aes_key(password, salt)
        iv = os.urandom(16)
        encrypted_data = encrypt_file(file_path, key, iv)
        print("Archivo cifrado.")

        print("Descifrando archivo...")
        decrypted_data = decrypt_file(encrypted_data, key, iv)
        with open("decrypted_file.txt", "wb") as f:
            f.write(decrypted_data)
        print("Archivo descifrado y guardado como 'decrypted_file.txt'.")

        # 3. Firmar y verificar el archivo para autenticidad
        print("\nFirmando archivo para autenticidad...")
        private_key, public_key = generate_rsa_keys()
        signature = sign_file(file_path, private_key)
        print("Archivo firmado.")

        print("Verificando firma del archivo...")
        is_signature_valid = verify_signature(file_path, signature,
public_key)
        print(f"Autenticidad verificada: {'Éxito' if is_signature_valid
else 'Fallo'}")
```