



Hybrid Encryption Prototype Combining AES and RSA Encryption Methods

Prototipo de Cifrado Híbrido Combinando los Métodos de Encriptación AES y RSA

Andrés Mauricio Clavijo ¹, Jhon Alexander Chacón ²,

Para citar este artículo: A. M. Clavijo, J. A. Chacón, "Hybrid Encryption Prototype Combining AES and RSA Encryption Methods", Revista Vínculos, vol 20, no. 1, p-p 63-74, 2023.
<https://doi.org/10.14483/2322939X.17668>

Recibido: 19-12-2021 / Aprobado: 10-03-2023

Resumen: Este trabajo tiene como finalidad presentar un prototipo de cifrado híbrido implementado una red virtual basada en grid, con el propósito de aprovechar el nivel de seguridad alcanzado mediante esta forma de cifrado. Por medio de una aplicación de mensajería, se analizará el método de cifrado propuesto con y sin la implementación del mismo, verificando los resultados obtenidos y validando posibles puntos débiles en el proceso de cifrado y transmisión de la información. El prototipo fue desarrollado basado en una arquitectura que le permite compartir recursos y

funcionalidades, con el fin de proveer la capacidad de operar en cualquier sistema que requiera el uso de ellos, por lo tanto, la aplicación de mensajería es un objeto de prueba que brinda información sobre la funcionalidad del prototipo de cifrado y el nivel de seguridad alcanzado para ser implementado en otros sistemas de comunicación.

Palabras clave: RSA, AES, cifrado simétrico, cifrado asimétrico, criptosistema híbrido.

1 Ingeniero en Telemática, Universidad Distrital Francisco José de Caldas, Facultad Tecnológica, Bogotá Colombia. E-mail: amclavijom@correo.udistrital.edu.co.

2 Ingeniero en Telemática, Universidad Distrital Francisco José de Caldas, Facultad Tecnológica, Bogotá- Colombia. E-mail: jhachacont@correo.udistrital.edu.co.

Abstract: The purpose of this work is to present a prototype of hybrid encryption implemented in a grid based virtual network, in order to take advantage of the security level achieved by this form of encryption. By means of a messaging application, the proposed encryption method will be analyzed with and without the implementation of the same, verifying the results obtained and validating possible weak points in the encryption and information transmission process. The prototype was developed based on an architecture that allows it to share resources and functionalities, in order to provide the ability to operate in any system that require the use of them, therefore, the messaging application itself is only a test object that provides information on the encryption prototype and the level of security achieved to be implemented in other communication systems.

Keywords: RSA, AES, symmetric encryption, asymmetric encryption, hybrid cryptosystem.

1. Introducción

Las técnicas destinadas a proteger la información en entornos de comunicación no seguros, como internet, han evolucionado significativamente sus niveles de complejidad y seguridad para la transferencia de información, sin embargo, se han encontrado falencias, ya sea a nivel del algoritmo criptográfico, en el canal de

comunicaciones o en la implementación de dicho algoritmo, lo que exige el avance o surgimiento de nuevos métodos de cifrado seguros.

El desarrollo del prototipo de cifrado se fundamenta en el uso del cifrado simétrico y asimétrico, con el objetivo de aprovechar las ventajas y disminuir las desventajas y/o vulnerabilidades de ambos en una solución integrada; este proceso se conoce como cifrado híbrido.

Este método consiste en proteger la llave privada del método de cifrado AES [1], aprovechando la ventaja del algoritmo AES en términos de velocidad de cifrado para grandes cadenas de datos, y las ventajas del cifrado asimétrico en la distribución de llaves. Es importante tener en cuenta que este enfoque requiere más capacidad computacional y/o de tiempo de procesamiento para llevar a cabo las tareas de cifrado con el fin de aumentar la complejidad y la seguridad en la información.

Se busca disminuir las debilidades del cifrado simétrico, mediante la encapsulación por parte del cifrado asimétrico [2]. Este método de encapsulamiento permite reducir las restricciones relacionadas con la longitud de los mensajes, sin afectar la seguridad del algoritmo, además, facilita la implementación de sistemas que aprovechen las ventajas del funcionamiento del cifrado híbrido, haciendo posible la sustitución de las llaves usadas en la comunicación, por

unas nuevas para el cifrado de la información, al realizar el proceso de transmisión de la información.

Cifrado simétrico: Este tipo de cifrado tiene como propiedad fundamental el uso de una clave privada, que se genera para realizar el proceso de cifrar y descifrar la información o datos que se transmiten. Los algoritmos criptográficos simétricos tienen dos versiones, cifrador en bloque y cifrador en flujo. Los cifradores en bloque codifican datos en bloques pequeños de longitud fija de 64 bits de longitud; hay distintos tipos de cifradores en bloque, que se utilizan por los algoritmos de cifrado DES, 3-DES, RC2, RC5, RC6 y Rijndael (AES) [3]. Este tipo de cifrado es el utilizado en el desarrollo del prototipo.

Cifrado asimétrico: Este tipo de cifrado es mucho más complejo que el simétrico, ambos tienen la funcionalidad de tratar la información para que no pueda ser interceptada por terceros, pero a diferencia estos tipos de sistemas utilizan dos claves, una privada y una pública. Ambas pueden ser usadas para cifrar y descifrar información; dichas claves están matemáticamente relacionadas entre sí, la clave pública está disponible para todos y la clave privada es conocida solo por el emisor. Existen varios algoritmos de cifrado asimétrico muy utilizados como Diffie-Hellman, RSA, DSA [4].

Cifrado híbrido: Es un algoritmo que combina dos o más algoritmos que resuelven el mismo problema, ya sea eligiendo uno

(según los datos o información a cifrar) o cambiando entre ellos a lo largo de la implementación del algoritmo. Esto generalmente se hace para combinar las características deseadas de cada uno, de modo que el algoritmo general sea mejor que los componentes individuales [5].

Computación en grid: La computación en grid (o en malla), ha surgido como un importante campo de la computación; a diferencia de otras arquitecturas, cuyo objetivo principal es desarrollar computadoras de gran rendimiento, medido en número de cálculos de punto flotante por unidad de tiempo, la importancia de la computación grid radica en la cantidad de trabajo que es capaz de gestionar y despachar en una unidad de tiempo. Esta tecnología no es revolucionaria, se puede decir que ha evolucionado de otras tecnologías como los sistemas distribuidos, la virtualización, los servicios web, el internet y la criptografía [6].

Servicios REST: Los servicios REST (Representational State Transfer) constituyen un estilo de arquitectura para desarrollar servicios; los servicios web basados en este estilo de arquitectura cumplen con las siguientes premisas: Cliente/Servidor: definen una interfaz de comunicación entre el cliente y el servidor, separando de forma clara las responsabilidades entre ambas partes. Sin estado: Son servicios web que no mantienen un estado asociado al cliente; cada petición que se realiza es completamente independiente de la

siguiente, lo que garantiza que todas las llamadas al mismo servicio sean idénticas e independientes entre sí [6].

WebSockets: Los WebSockets introducidos con la llegada de HTML5, representan una de las mejoras más destacadas y ampliamente utilizadas en el desarrollo web. Esta tecnología permite una comunicación bidireccional entre el navegador y el servidor, posibilitando el envío y la recepción de mensajes de forma simultánea (Full Duplex), además mantienen una conexión activa con el servidor utilizando Sockets TCP [7].

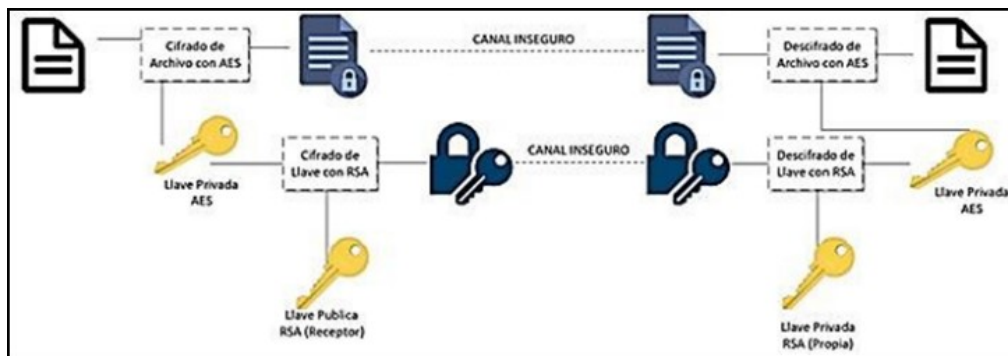
Los autores Avinash, R., Potnis, A., Kumar, S., Dwivedy, P., y Soofi, S., presentan un modelo de encriptación híbrido que combina diferentes técnicas, este modelo comienza con el cifrado mediante el algoritmo RSA, seguido del uso del algoritmo AES y finaliza con la operación XOR. En el proceso, los datos se cifran utilizando el algoritmo AES con una llave de 128 bits (10 rondas) [8].

Los resultados experimentales del modelo propuesto por los autores muestran que el cifrado tiene un rendimiento variable, esto debido a los resultados en la medición de parámetros relacionados con su nivel de complejidad. Así mismo, Anane, M., y Nadja, A, proponen un modelo basado en componentes electrónicos para optimizar el cifrado mediante el algoritmo AES, ya que, según los autores, el proceso de cifrado necesita ser acelerado por hardware a fin de

brindar mejoras en el desempeño. También se plantea el desarrollo de un software para realizar la encriptación de la llave del algoritmo AES sobre RSA, adicional a la generación de RoundKeys del algoritmo AES, la cual es realizada solo al inicio de las tareas de cifrado y descifrado [9]. Los resultados de la aplicación de este sistema muestran que la optimización realizada a nivel de hardware permite mejorar la eficiencia, cuando se trata de ejecutar procesos de cifrado, proporcionando tiempos de respuesta óptimos, que permiten un aumento en el rendimiento de la ejecución del algoritmo de cifrado AES.

2. Metodología

El objetivo de este trabajo es desarrollar un prototipo de cifrado híbrido, mediante la combinación de los algoritmos de encriptación AES y RSA, aprovechando los altos niveles de seguridad que ofrece cada uno de ellos, la implementación se realiza a través de una aplicación de mensajería en una virtualización de un sistema de red en grid. Se plantea el uso de los algoritmos AES y RSA para generar un método de cifrado híbrido, capaz de proveer un nivel de seguridad adecuado en la transmisión de información a través de internet. La Figura 1 presenta la implementación del modelo de cifrado híbrido.

Figura 1. Modelo de cifrado híbrido.**Fuente:** Elaboración propia.

La complejidad Computacional en los métodos de cifrado híbrido radica en el proceso de generación de las llaves, debido a que se requiere ejecutar métodos matemáticos que cumplan las condiciones del cifrado simétrico y asimétrico, estos métodos deben garantizar que, incluso ante ataques de criptoanálisis, como el método de fuerza bruta, los datos sean casi imposibles de descifrar, protegerlos contra intentos de alteración o robos de información, y mantener la seguridad aun cuando se encuentran las llaves de descifrado.

Para este prototipo, se realiza la implementación de una aplicación de escritorio y de un entorno virtual en grid. Con el fin de validar su funcionalidad, se plantea el uso de llaves de longitud adecuada para AES y RSA. Para el caso del algoritmo AES, se establece una longitud de llave privada de 256 bits, 14 rondas, de acuerdo a lo planteado en [10]; esta longitud provee

una adecuada protección, como se presenta en la Tabla 1.

Tabla 1. Niveles de seguridad para algoritmos comunes de cifrados en bloque.

Nombre	Longitud de la clave	Longitud del bloque	Nivel de seguridad
DES	56	64	56
triple DES de dos llaves	112	64	95-100
triple DES de tres llaves	168	64	112-116
DESX	120	64	120
IDEA	128	64	128
AES-128	128	128	128
AES-192	192	128	192
AES-256	256	128	256

Fuente:

<https://infoscience.epfl.ch/record/164539/files/NPDF-32.pdf>.

Para el algoritmo RSA, se establece que una longitud de llaves de 2048 bits, y de acuerdo a la tabla 2, es suficiente para brindar una adecuada protección [11].

Tabla 2. Proyección de seguridad en el tiempo según la longitud de las llaves en bits.

Año	Longitud de las llaves en bits (optimista)	Longitud de las llaves en bits (conservadora)
2010	1112	1153
2020	1387	1569
2030	1608	2064
2040	2048	2645
2050	2439	3314

Fuente:

<https://infoscience.epfl.ch/record/164539/files/NPDF-32.pdf>.

Para los procesos de firmado y verificación de la información, se propone el uso de las llaves generadas para el cifrado del algoritmo RSA, con el fin de aprovechar los recursos computacionales usados en el proceso de generación de llaves y cifrado, a pesar de no ser el algoritmo de firmas más rápido y sencillo [12]; además, se implementa el algoritmo hash SHA- 512, para alcanzar un mensaje suficientemente corto y fácil de procesar por el emisor y receptor, sin comprometer la seguridad que este proporciona. Teniendo en cuenta las diferentes librerías utilizadas para el desarrollo del prototipo en lenguaje .Net, se elige la versión de núcleo 4.7.2. La librería AesCryptoServiceProvider, permite realizar el uso del algoritmo AES de manera correcta y la generación del Vector de Inicialización (IV) de forma rápida y segura, los parámetros utilizados para la implementación del algoritmo AES en el prototipo se detallan en la Tabla 3. Para la implementación del algoritmo RSA, se utiliza la librería RSACryptoServiceProvider, la cual, con una longitud de bits particular, genera un par de llaves, la llave pública y la llave privada,

para su uso en los procesos de cifrado, descifrado y firmado.

Tabla 3. Parámetros establecidos para el proceso de cifrado con el algoritmo AES.

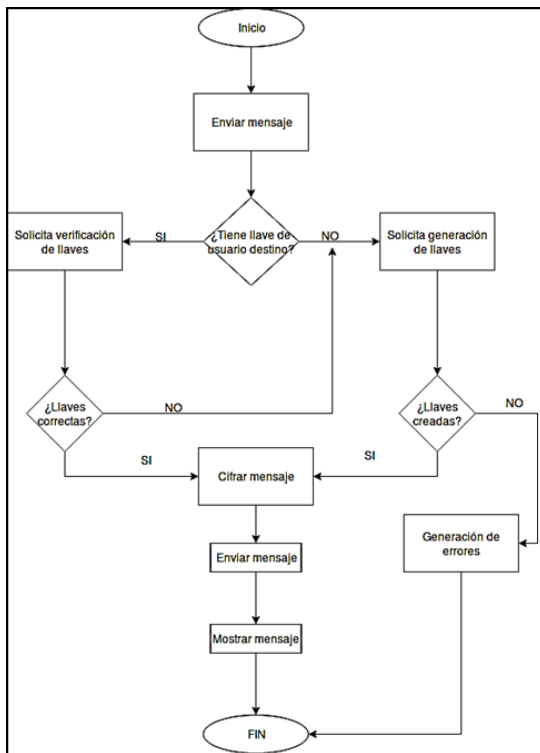
Tamaño de la llave privada en bits	Tipo de Relleno	Modo de cifrado en bloque	Vector de inicialización
256	PKCS7	CBC	Aleatorio

Fuente:

<https://infoscience.epfl.ch/record/164539/files/NPDF-32.pdf>.

3. Arquitectura Planteada

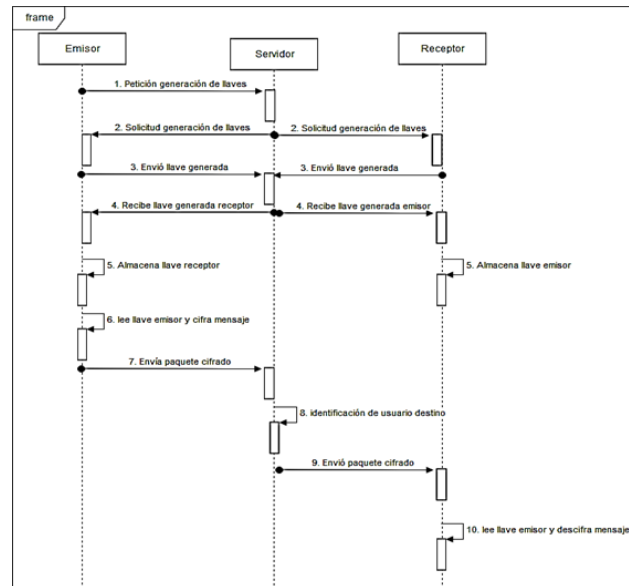
Para el desarrollo del servicio de cifrado y descifrado, se tienen en cuenta las acciones que los usuarios pueden realizar en el prototipo, con el objetivo de establecer una comunicación segura. Es importante segmentar las operaciones en diferentes servicios, dado que la información no puede emitirse desde el origen sin ser procesada. En este caso, el envío de un texto cifrado inicia un procedimiento que inculca tanto servicios locales como remotos. La Figura 2 ilustra el proceso planteado y los pasos a realizar en el prototipo para cifrar, enviar y capturar (mostrar) un mensaje de manera segura y adecuada.

Figura 2. Diagrama de flujo del mensaje.

Fuente: Elaboración propia.

Después de comprender el funcionamiento lógico del prototipo, se plantea un escenario de comunicación, representado por el diagrama de secuencia mostrado en la Figura 3 en este escenario intervienen dos usuarios, el emisor y el receptor, quienes iniciaran una comunicación, y un servidor encargado de gestionar los respectivos procesos de direccionamiento y transmisión de la información. El emisor es quien realiza la petición al servidor para iniciar una conversación con el receptor, una vez ocurre la primera llamada, el servidor comienza a gestionar las peticiones entre ambos usuarios, llevando a cabo los procesos de generación e intercambio de llaves, permitiendo que los usuarios puedan enviar

mensajes de un punto al otro de manera cifrada y segura.

Figura 3. Diagrama de secuencia del prototipo.

Fuente: Elaboración propia.

Desarrollo del Prototipo: Los diferentes componentes para el desarrollo del entorno y aplicación, se realizan sobre un ambiente virtual con dos hosts, con el fin de simular la intercomunicación real y la transmisión de información.

Entorno virtual: Para la implementación del ambiente virtual se utiliza el software de virtualización VMware Player Workstation; en este entorno se crean tres servidores virtuales: dos servidores de aplicaciones y un servidor BOINC, con el fin de simular el ambiente en grid entre dos terminales que harán las veces de host, creando una máquina virtual (VM) del servidor de

aplicaciones en cada uno de ellos, como se observa en la Figura 4.

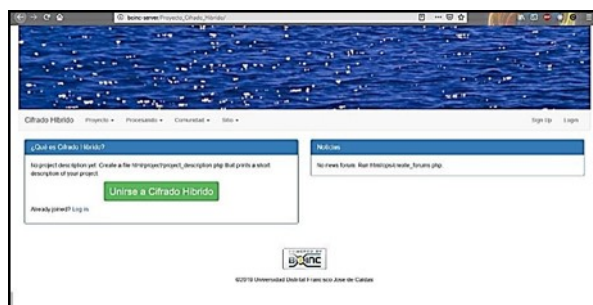
Figura 4. Máquinas virtuales creadas en un host.



Fuente: Elaboración propia.

Implementación de la red en malla: Para la creación del prototipo de red en malla, se hace uso del software con licencia de código abierto BOINC, el cual permite la creación y unión a entornos de red en malla, con el fin de hacer uso de los recursos de cada máquina unida al proyecto, y la ejecución de determinadas tareas, como se observa en la Figura 5.

Figura 5. Proyecto Cifrado Híbrido en BOINC.

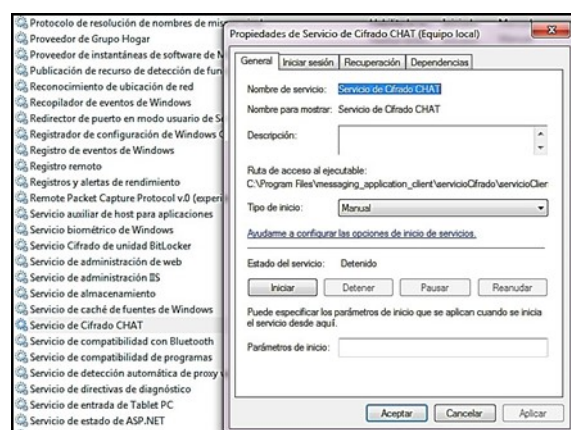


Fuente: Elaboración propia.

Servicio de cifrado híbrido: Los servicios de cifrado y descifrado se desarrollan en el lenguaje

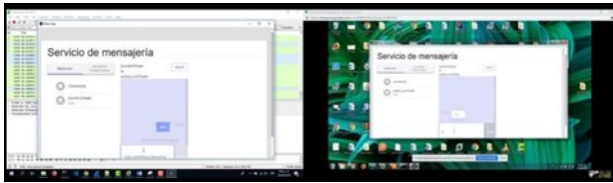
.Net core y Java con servicios Rest. El servicio de cifrado se diseña para trabajar de forma local en cada uno de los usuarios, y el servicio de descifrado para uso remoto (servidor). Los usuarios se comunican a través del servidor para gestionar la distribución de llaves y establecer una comunicación segura entre ellos, este flujo de trabajo y la interacción entre los componentes se representa en la Figura 6.

Figura 6. Propiedades del Servicio de Cifrado CHAT.



Fuente: Elaboración propia.

Aplicación de mensajería: El componente de mensajería es desarrollado para utilizar los servicios creados del módulo de cifrado. El desarrollo se realiza con ReactJs, el cual provee un rendimiento adecuado y permite generar aplicaciones intuitivas; posteriormente se emplea el Electrón para generar una aplicación de escritorio, lo que permite su instalación y uso en el ambiente de desarrollo propuesto, La estructura y funcionalidad del componente se observa en la Figura 7.

Figura 7. Servicio de mensajería entre dos clientes.

Fuente: Elaboración propia.

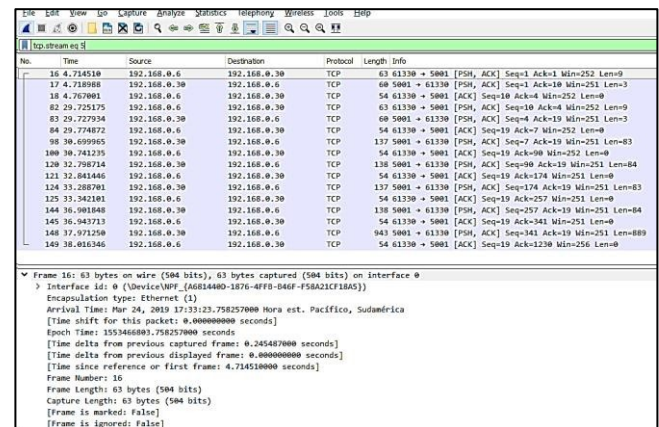
4. Pruebas del Prototipo

Inicialmente, el servicio de mensajería se utiliza sin el prototipo de cifrado sobre el ambiente en grid, permitiendo la comunicación directa entre los clientes; sin embargo, al interceptar los paquetes de datos mediante la herramienta de monitoreo (sniffer) Wireshark, se evidencia que la información es fácil de interpretar, debido a que esta viaja en texto claro, lo cual es considerado una vulneración de la seguridad de la información, dicha vulneración se aprecia en Figura 8.

Figura 8. Captura de mensajes sin prototipo de cifrado.

Fuente: Elaboración propia.

Seguidamente, con la implementación del prototipo de cifrado, se validan los paquetes capturados y los mensajes cifrados como se muestran en las Figuras 9 y 10, por medio del análisis del tráfico generado por el servicio de mensajería utilizando herramientas de escaneo de paquetes. Se evidencia que no es posible interpretar fácilmente el mensaje que se está transmitiendo, debido al método de cifrado híbrido implementado.

Figura 9. Escaneo de paquetes.

Fuente: Elaboración propia.

Figura 10. Captura de mensajes con implementación del prototipo de cifrado

Fuente: Elaboración propia.

5. Resultados y Discusión

Una posible vulneración sobre el algoritmo RSA mostraría en claro la llave del proceso AES, permitiendo descifrar la información enviada, pero que este proceso de vulneración sea exitoso una vez, no garantiza que las próximas transmisiones de mensajes puedan ser descifradas con la misma llave encontrada, debido a que estas llaves se generan nuevamente cada vez que se envía la información. Si en efecto, la información fue descifrada, y el atacante altera los datos e intenta engañar al receptor con paquetes corruptos, el sistema doble de verificación de integridad, proporcionado por las firmas digitales del algoritmo RSA y el Vector de

Inicialización del algoritmo AES, permite identificar que la información fue alterada por un tercero, y por lo tanto no se presenta en la interfaz de usuario (UI), registrándose internamente como un origen de información no válido. Se genera un prototipo de cifrado que puede ser integrado a un servicio de comunicación que requiera seguridad en el envío de los datos, por medio de servicios de mensajería, sin la necesidad de generar certificados de seguridad. Los análisis a los sistemas criptográficos usados para la implementación del modelo híbrido aún permiten determinar un nivel adecuado de eficiencia y seguridad, según estudios realizados.

6. Conclusiones

El prototipo planteado presenta una buena funcionalidad, pero puede verse afectado al realizar análisis de vulnerabilidad más profundos y extensos, realizados con métodos y herramientas de criptoanálisis potentes.

Se pueden implementar distintos métodos y alternativas para la protección de información, ya sea de forma personal o corporativa. Aunque existen herramientas de software que permiten la implementación de los algoritmos de cifrado clásico, es importante proponer y probar distintos métodos e ideas, para que las nuevas tecnologías sean fáciles de integrar en soluciones de comunicaciones ágiles, actuales y seguras.

La utilización de distintas herramientas permiten que la información pueda cifrarse, con el fin mantener la seguridad y confidencialidad de los datos, juegan un papel importante y relevante, ya que la disponibilidad y variedad de sistemas de comunicación cada día son mayores, por lo que la implementación de criptosistemas, pueden prevenir alteraciones, suplantaciones y robo de la información de distinta índole.

El avance en materia de seguridad de la información es grande, pero también aumentan las formas de ataque a los sistemas de cifrado de la información, por lo que el estudio y análisis constante de nuevos

métodos de cifrado, que permitan la mejora de seguridad en la información, es vital para el desarrollo social y tecnológico.

Referencias

- [1] A. Al Hasib, "A Comparative Study of the Performance and Security Issues of AES and RSA Cryptography, de Convergence Information Technology," *International Conference*, Finlandia, 2008.
- [2] Shamir R.L. Rivest y L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Magazine Communications of the ACM*, Vol. 21 pp. 120–126, 1978. doi <https://doi.org/10.1145/359340.359342>
- [3] A. Castro, Lechtaler, M. Cipriano, E. García, J. Liporace, A. Maiorano, E. Malvacio, N. Tapia, "Estudio de técnicas de criptoanálisis," *XXI Workshop de Investigadores en Ciencias de la Computación*. 2021. [en línea]. Disponible en <http://sedici.unlp.edu.ar/handle/10915/77269>
- [4] J. C. Mendoza, "Universidad Politécnica Salesiana de Ecuador," [En línea]. Disponible en <https://dspace.ups.edu.ec/bitstream/123456789/8185/1/Demostraci%C3%B3n%20de%20cifrado%20sim%C3%A9trico%20y%20asim%C3%A9trico.pdf>.
- [5] A. W. Dent, "Hybrid Cryptography," 3 Jun. 2009. [En línea]. Disponible en <https://eprint.iacr.org/2004/210.ps>
- [6] M. G. Escobar, "Clúster de alto rendimiento en un cloud: ejemplo de aplicación en criptoanálisis de funciones hash," *Universidad de Almería*. p. 60, 2011. [En línea]. Disponible en <http://repositorio.ual.es/bitstream/handle/10835/1202/PFC.pdf?sequence=1>
- [7] A. Pousa, "Universidad Nacional de la Plata," Dic. 2011. [En línea]. Disponible en: https://postgrado.info.unlp.edu.ar/wp-content/uploads/2014/07/Pousa_Adriana.pdf.
- [8] A. Lenstra, "Key Lengths," [En línea]. Disponible en: <https://infoscience.epfl.ch/record/164539/files/NPDF-32.pdf>.
- [9] R. Avinash, A. Potnis, S. Kumar, P. Dwivedy y S. Soofi, "Internation Journal Of Engineering Research and Applications," Ag. 2017. [En línea]. Disponible en: http://www.ijera.com/papers/Vol7_issue8/Part-1/O0708019094.pdf

- [10] A. Faget, "What are Cryptographic Signatures? Introduction to the Most Common Schemes," 14 Nov. 2018. [En línea]. Disponible en: <https://coindoo.com/what-are-cryptographic-signatures-introduction-to-the-most-common-schemes/>
- [11] O. Goldreich, "Modern Cryptography, Probabilistic Proofs and Pseudorandomness (Second Edition - author's copy)," Springer. pp. 1-2, 2000. [En línea]. Disponible en: <http://www.wisdom.weizmann.ac.il/~oded/PDF/mcppp-v2.pdf>
- [12] J. Chavez, "Sistema de encriptación RSA", 2022. [En línea]. Disponible en: https://biblus.us.es/bibing/proyectos/usa/abreproy/10586/fichero/memoria_PFC.pdf
- [13] H. J. Eslava Blanco, J. F. Rocha, y J. I. Morales, "Estudio de tráfico sobre una plataforma de virtualización," *Visión Electrónica*, vol. 5, no. 2, pp. 78-94, 2011. [En línea]. doi: <https://doi.org/10.14483/22484728.3572>
- [12] B. Pushpa, "Hybrid Data Encryption Algorithm for Secure Medical Data Transmission in Cloud Environment," 2020 *Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, pp. 329-334, 2020.
- [13] M. Suresh, M. Neema, "Hardware implementation of blowfish algorithm for the secure data transmission in Internet of Things," *Procedia technology*, vol. 25, pp. 248-255, 2016.
- [14] B. T. Asare, K. Quist-Aphetsi & L. Nana, "Secure Data Exchange Between Nodes in IoT Using TwoFish and DHE," 2019 *International Conference on Cyber Security and Internet of Things (ICSIoT)*, Accra, Ghana, pp. 101-104, 2019.
- [15] P. Patel, R. Patel, & N. Patel, "Integrated ECC and Blowfish for smartphone security," *Procedia Computer Science*, vol. 78, pp. 210-216, 2016.