

FORO DE DISCUSIÓN CON CIFRADO DE MENSAJES USANDO CAJAS P Y CAJAS S.

María del Rocío Hernández Flores¹, Bárbara Emma Sánchez Rinza²
Benemérita Universidad Autónoma de Puebla
Faculta de Ciencias de la Computación¹, Faculta de Ciencias Físico Matemáticas²
Ciudad Universitaria
hf223470495@alm.buap.mx, barbara.sanchez@correo.buap.mx

RESUMEN

Este trabajo presenta el desarrollo e implementación de un sistema de cifrado y descifrado de texto utilizando cajas de permutación (P) y cajas de sustitución (S) aplicado a un entorno de comunicación tipo foro o chat. El sistema está compuesto por un servidor central y múltiples clientes que intercambian mensajes cifrados en tiempo real. Para garantizar la confidencialidad de los mensajes, se diseñó un esquema de cifrado basado en bloques de 32 bits, donde se aplican transformaciones iterativas con cajas P y S. Las funciones de cifrado y descifrado fueron integradas tanto en el servidor como en los clientes, permitiendo que los mensajes sean procesados de manera automática y transparente para los usuarios.

El sistema también asegura la persistencia del historial de mensajes, permitiendo que nuevos usuarios conectados recuperen los mensajes previos sin comprometer la seguridad. Los resultados experimentales muestran que el cifrado y descifrado se ejecutan correctamente, manteniendo la integridad del contenido y demostrando la viabilidad del uso de técnicas criptográficas clásicas en sistemas interactivos de mensajería. Esta implementación sienta las bases para futuras extensiones hacia esquemas más robustos y adaptativos.

Palabras Clave: cifrado simétrico, cajas S y P, seguridad en comunicaciones

ABSTRACT

This work presents the development and implementation of a text encryption and decryption system using permutation boxes (P-boxes) and substitution boxes (S-boxes), applied to a forum-style communication environment. The system consists of a central server and multiple clients that exchange encrypted messages in real time. To ensure message confidentiality, a block-based encryption scheme was designed using 32-bit blocks, where iterative transformations with P-boxes and S-boxes are applied. The encryption and decryption functions were integrated into both the server and the clients, allowing messages to be processed automatically and transparently for users.

The system also preserves the message history, enabling newly connected users to retrieve previously exchanged messages without compromising security. Experimental results demonstrate that encryption and decryption are executed correctly, maintaining message integrity and proving the feasibility of using classical cryptographic techniques in interactive messaging systems. This implementation lays the groundwork for future extensions toward more robust and adaptive schemes.

Keywords: symmetric encryption, S-boxes and P-boxes, communication security

1. INTRODUCCIÓN

El cifrado y la protección de datos constituyen elementos esenciales para garantizar la seguridad en la transmisión de información. Dos conceptos fundamentales en los algoritmos de cifrado modernos son las cajas P (permutación) y las cajas S (sustitución), ya que desempeñan un papel clave en la construcción de sistemas criptográficos robustos. Estos componentes son comunes en algoritmos de cifrado simétrico, como el Estándar de Cifrado de Datos (DES) y el Algoritmo de Cifrado Avanzado (AES), y están diseñados para transformar los datos de manera no lineal y aparentemente aleatoria [1], [2].

Las cajas S (Substitution boxes) son tablas de búsqueda no lineales utilizadas para sustituir partes del texto cifrado, lo que introduce complejidad y confusión en el mensaje. Estas cajas aplican sustituciones que convierten un conjunto de bits de entrada en otro conjunto de bits de salida de forma no lineal, una propiedad crucial para evitar que los atacantes puedan identificar patrones simples en los datos cifrados [3].

El funcionamiento básico de una caja S se basa en una tabla predefinida, en la que los bits de entrada se utilizan como índices para seleccionar una fila y una columna que determinan la salida correspondiente. En los algoritmos de cifrado, múltiples cajas S se aplican de manera secuencial o paralela para incrementar la confusión. Este término hace referencia a la dificultad para establecer una relación directa entre el texto cifrado y la clave utilizada, lo que reduce la efectividad de los ataques de fuerza bruta [4].

Por ejemplo, en el algoritmo DES se emplean ocho cajas S, cada una de las cuales recibe un bloque de 6 bits de entrada y produce un bloque de 4 bits de salida. Estas cajas han sido diseñadas con propiedades matemáticas específicas que maximizan la resistencia frente a ataques criptoanalíticos, como los ataques diferenciales y lineales [5].

Mientras que las cajas S introducen confusión, las cajas P (Permutation boxes), también conocidas como redes de permutación, aportan difusión al proceso de cifrado. La difusión se refiere a la dispersión de los bits del mensaje original a lo largo del texto cifrado, asegurando que cada bit de salida esté influenciado por varios bits de entrada. Este mecanismo garantiza que cualquier cambio en el texto claro o en la clave se propague ampliamente a través del texto cifrado [1], [6].

En su forma más simple, una caja P reordena los bits de su entrada conforme a un patrón predefinido. De este modo, los bits que originalmente se encontraban próximos pueden terminar separados, lo que incrementa la complejidad del mensaje cifrado y dificulta la detección de patrones. Aunque la permutación no modifica el contenido de los datos, sí altera su posición; al combinarse con las cajas S y múltiples rondas de cifrado, esta estrategia resulta altamente efectiva para impedir la inversión del proceso sin el conocimiento de la clave [3], [6].

Un ejemplo representativo del uso de permutación es la red de Feistel empleada en DES. Tras la aplicación de las cajas S, los resultados son sometidos a una permutación que contribuye a la resistencia del algoritmo frente a ataques que intenten explotar su estructura interna [2], [5].

La combinación de cajas S y cajas P constituye una estrategia central en el diseño de algoritmos de cifrado seguros. Mientras que las cajas S introducen complejidad no lineal (confusión), las cajas P se encargan de mezclar los bits (difusión), generando un cifrado resistente a distintos tipos de ataques criptoanalíticos [4], [6].

En el algoritmo DES, las cajas S y P forman parte de un esquema de cifrado tipo Feistel, en el cual el mensaje se divide en dos mitades; una de ellas se transforma mediante una clave y las cajas S, mientras que la otra se permuta. Este proceso se repite a lo largo de varias rondas, incrementando exponencialmente la dificultad de descifrar el mensaje sin la clave correspondiente. En algoritmos más avanzados, como AES, se utilizan cajas S que operan sobre bytes completos en lugar de bits, y las permutaciones se aplican sobre matrices de bytes, lo que proporciona un nivel de seguridad aún mayor [1], [7].

2. AVANCES RECIENTES EN EL DISEÑO DE CAJAS S Y P

Las cajas S se introdujeron formalmente en la criptografía con el algoritmo Data Encryption Standard (DES), donde desempeñaban un papel fundamental en la generación de confusión sobre el mensaje cifrado. En DES, las ocho cajas S no solo proporcionaban seguridad, sino que también establecieron un estándar para la evaluación de la no linealidad en sustituciones criptográficas [2], [5]. No obstante, con el avance del criptoanálisis, las limitaciones de DES y sus cajas S fueron rápidamente explotadas, lo que dio lugar a la necesidad de algoritmos más robustos, como el Advanced Encryption Standard (AES) [1].

AES mejoró significativamente el diseño de las cajas S al incorporar una única caja que opera sobre bytes completos en lugar de bloques de bits. Esta caja S se construye mediante una combinación de transformaciones algebraicas sobre cuerpos finitos, lo que maximiza su resistencia frente a ataques diferenciales y lineales [1], [8].

Una línea de investigación emergente es el uso de cajas S dinámicas, cuyo contenido varía durante el proceso de cifrado. Este enfoque ha sido explorado para incrementar la resistencia frente a ataques de temporización y análisis diferencial [9]. Las

cajas S dinámicas modifican la tabla de sustitución en función de la clave o del estado interno del sistema, proponiendo así una alternativa adaptable a distintos escenarios criptográficos.

Otro enfoque reciente involucra el uso de redes neuronales para la construcción de cajas S. Esta línea de investigación explora cómo las redes pueden entrenarse para aproximar funciones de sustitución con propiedades criptográficas deseadas [10]. Estas cajas S generadas por inteligencia artificial podrían adaptarse dinámicamente a diversos ataques, ofreciendo un nuevo horizonte en el diseño criptográfico.

Por otro lado, las cajas P han experimentado una evolución menos acelerada, pero continúan siendo un componente esencial en la construcción de cifrados. Inicialmente concebidas como permutaciones lineales simples, investigaciones recientes han propuesto estructuras más complejas, como redes de permutación caóticas, que introducen comportamientos no deterministas en el reordenamiento de bits [11].

En el caso de AES, la permutación se realiza a nivel de bloques de bytes. En esta arquitectura, se reordenan las columnas del estado, y esta permutación se combina con operaciones de mezcla para lograr una difusión rápida y eficaz a través de múltiples rondas [1], [7].

En entornos con restricciones de hardware, como dispositivos embebidos, el diseño de cajas P eficientes es crucial. Para aplicaciones que requieren alta velocidad o bajo consumo energético, se han propuesto cajas P optimizadas para implementaciones en FPGA o ASIC, capaces de procesar grandes volúmenes de datos con baja latencia sin comprometer la seguridad [12].

Con la aparición de nuevos ataques criptoanalíticos, las cajas P y S han sido objeto de investigación intensiva. Ataques diferenciales y lineales buscan explotar correlaciones matemáticas entre texto claro, texto cifrado y clave. Para contrarrestarlos, se han propuesto diseños de cajas S que maximizan la distancia de Hamming entre entradas y salidas, dificultando el análisis estadístico [5], [13].

Un desafío emergente lo representan los ataques basados en aprendizaje automático. Se ha comenzado a explorar si modelos de machine learning pueden entrenarse para identificar patrones en el comportamiento de las cajas S y P, lo que podría comprometer la seguridad de algoritmos actuales [14]. Como respuesta, se ha propuesto el diseño de cajas que generen transformaciones estadísticamente impredecibles.

Finalmente, el crecimiento de la computación cuántica también ha motivado la revalorización del diseño de cajas P y S. Aunque la criptografía cuántica aún se encuentra en desarrollo, se están evaluando propuestas para adaptar estas estructuras a un entorno resistente a algoritmos cuánticos como los de Shor o Grover, que amenazan la solidez de los cifrados simétricos clásicos [15].

3. DESARROLLO

Este trabajo consiste en la implementación del cifrado y descifrado de texto utilizando cajas P (de permutación) y cajas S (de sustitución), aplicadas a un entorno de comunicación tipo

foro o chat entre múltiples usuarios. El sistema permite que los mensajes enviados sean cifrados al momento de la transmisión y descifrados al momento de su recepción, asegurando la confidencialidad durante el intercambio de información.

Además, el diseño contempla que, si un usuario se conecta después de que otros ya hayan enviado mensajes, dicho usuario pueda recuperar y visualizar el historial completo de mensajes previamente cifrados.

El foro está implementado a partir de dos componentes principales: un servidor y un cliente para cada usuario. Ambos códigos comparten funciones encargadas del proceso de cifrado y descifrado, descritas en las siguientes subsecciones.

3.1. Funciones de cifrado y descifrado

La Figura 1 muestra el diagrama de flujo que representa el funcionamiento del sistema de cifrado y descifrado desarrollado. Este sistema opera sobre bloques de 32 bits y utiliza transformaciones basadas en permutaciones (caja P) y sustituciones (cajas S), con la finalidad de incrementar la difusión y confusión del mensaje original, principios fundamentales en criptografía.

El proceso de cifrado inicia con la conversión del texto plano a su representación binaria, donde cada carácter es codificado en 8 bits. Posteriormente, el flujo divide la secuencia binaria en bloques de 32 bits que son sometidos a una serie de transformaciones. Cada bloque es procesado en cuatro rondas, donde en cada una se aplica primero una permutación fija mediante la caja P, seguida de la división del bloque en cuatro sub-bloques de 8 bits. Cada sub-bloque es entonces sustituido mediante una caja S distinta (S1 a S4), y finalmente se reconstruye el bloque de 32 bits. El resultado de estas transformaciones es una secuencia binaria cifrada, que representa el texto protegido.

El proceso de descifrado (Figura 1) es simétrico e inverso al cifrado. Se parte del texto cifrado en formato binario, el cual se divide en bloques de 32 bits. Cada bloque es procesado en cuatro rondas inversas, donde se realiza primero la separación en sub-bloques de 8 bits, luego se aplica la caja S inversa correspondiente a cada sub-bloque ($S1^{-1}$ a $S4^{-1}$), se reconstruye el bloque completo y finalmente se aplica la permutación inversa mediante la caja P^{-1} . Al finalizar el proceso, los bloques recuperados son concatenados y convertidos nuevamente a texto, obteniendo así el mensaje original.

Este flujo asegura que la transformación sea reversible y permite recuperar con exactitud el mensaje original siempre que se respeten las mismas reglas de permutación y sustitución utilizadas durante el cifrado.

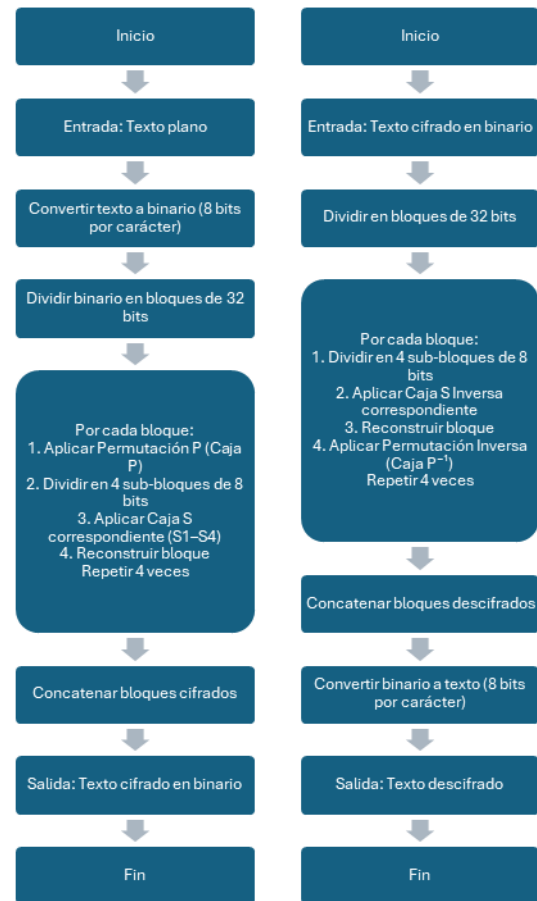


Figura 1. Diagrama de flujo del proceso de cifrado (derecha) y descifrado (izquierda) implementado en el sistema. El cifrado transforma el texto plano en una secuencia binaria mediante permutaciones fijas (caja P) y sustituciones no lineales (cajas S). El descifrado invierte este proceso aplicando las cajas S inversas y la permutación inversa (caja P^{-1}) para recuperar el mensaje original.

Las siguientes funciones son responsables de transformar el texto plano en un texto cifrado mediante cajas P y S, así como de recuperar el texto original durante el proceso de descifrado:

- `text_to_bin`: convierte un texto en su representación binaria, generando un string de 0s y 1s.
- `bin_to_text`: convierte una cadena binaria nuevamente a texto.
- `p_box`: realiza una permutación del bloque de entrada conforme al orden especificado.
- `inverse_p_box`: revierte la permutación realizada por la caja P.
- `s_box`: aplica una caja S o su inversa a un sub-bloque de 8 bits basado en un índice específico. Las cajas S emplean tablas de sustitución generadas mediante operaciones de multiplicación modular.

- `inverse_s_box`: genera la versión inversa de cada caja S, permitiendo así la recuperación del mensaje original.
- `encrypt_block`: cifra un bloque de 32 bits aplicando la caja P y las cajas S de manera iterativa en un ciclo de cuatro rondas.
- `decrypt_block`: descifra un bloque de 32 bits invirtiendo el proceso de cifrado, es decir, aplicando las cajas S inversas y la caja P inversa.
- `encrypt`: convierte el texto plano a binario, lo divide en bloques de 32 bits, y aplica el proceso de cifrado a cada bloque.
- `decrypt`: toma un texto binario cifrado, lo divide en bloques de 32 bits, aplica el proceso de descifrado correspondiente y devuelve el texto original.

3.2. Funciones del servidor

El servidor es responsable de recibir los mensajes cifrados de los distintos usuarios y distribuirlos a todos los clientes conectados, así como de conservar el historial completo de los mensajes. Contiene las siguientes funciones clave:

- `broadcast`: envía un mensaje cifrado a todos los clientes conectados al servidor.
- `handle_client`: maneja la conexión con un cliente individual. Al conectarse, el cliente recibe el historial de mensajes cifrados. Luego, esta función recibe continuamente mensajes del cliente, los guarda en el historial y los redistribuye.

3.3. Funciones del cliente

El cliente permite a los usuarios interactuar con la interfaz gráfica del chat, enviar mensajes cifrados y recibir mensajes descifrados del servidor. Las funciones principales son:

- `send_message`: toma el mensaje ingresado por el usuario, lo cifra utilizando la función `encrypt` y envía tanto la longitud como el contenido cifrado al servidor.
- `receive_messages`: recibe continuamente los mensajes del servidor, los descifra usando la función `decrypt` y los muestra en la interfaz.
- `show_message`: inserta los mensajes recibidos en el área de texto y ajusta la vista para mostrar los mensajes más recientes automáticamente.

4. RESULTADOS

Para validar el correcto funcionamiento del sistema de cifrado y descifrado implementado con cajas P y cajas S, se realizaron pruebas de comunicación entre múltiples clientes a través del servidor. En estas pruebas, los mensajes fueron cifrados automáticamente antes de su envío y descifrados al momento de su recepción.

La Figura 2 muestra la salida en consola del servidor durante una sesión de prueba. En ella se visualizan tanto los mensajes recibidos por el servidor en su forma cifrada (representados en binario), como su versión descifrada, lo cual permite verificar la

integridad del sistema de cifrado y la capacidad de recuperación del mensaje original.

```
D:\UN\Criptografia\Python>python server_cps.py
[Mensaje cifrado recibido] 1101110000100011110010011100000100100000110000110010001100
[Mensaje descryptado] Hola soy Mariano
[Mensaje cifrado recibido] 1101110000100011110010011100000100100000110000110010001100
[Mensaje descryptado] Hola soy Maria
[Nueva conexión] Un cliente se ha conectado.
[Mensaje cifrado recibido] 1101110000100011110010011100000100100000110000010010000011
[Mensaje descryptado] Hola a todos
```

Figura 2. Consola del servidor mostrando mensajes cifrados en binario y su recuperación mediante descifrado

La Figura 3 presenta la interfaz gráfica del cliente, donde se observan los mensajes intercambiados entre distintos usuarios en texto plano. Cada mensaje mostrado ha sido cifrado previamente antes de ser enviado al servidor y descifrado al ser recibido por los clientes, lo que confirma que el proceso de cifrado y descifrado es transparente para el usuario final.

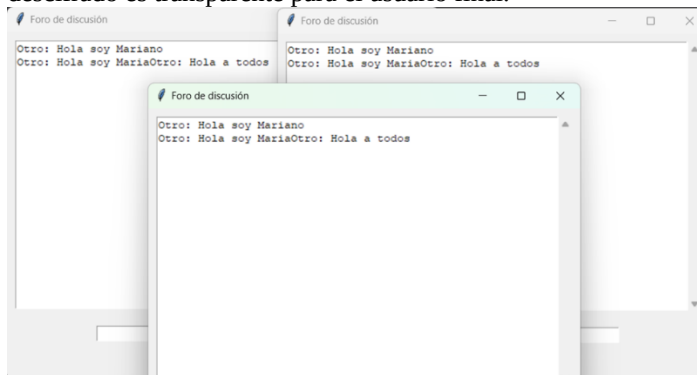


Figura 3. Interfaz gráfica del cliente mostrando mensajes en texto plano intercambiados en el chat.

Estos resultados demuestran que el sistema es capaz de cifrar mensajes en tránsito y recuperarlos correctamente en el extremo receptor. Además, se verifica que los nuevos usuarios conectados al foro reciben el historial completo de mensajes previamente enviados, lo que valida la persistencia y compatibilidad del cifrado con múltiples sesiones.

5. CONCLUSIONES

En este trabajo se implementó un sistema de cifrado y descifrado de texto basado en cajas P y cajas S aplicado a un foro de discusión entre múltiples usuarios. La arquitectura propuesta permite que los mensajes enviados a través del servidor sean cifrados automáticamente antes de su transmisión y descifrados en los clientes al momento de su recepción, garantizando así la confidencialidad del contenido durante la comunicación.

El uso combinado de cajas de sustitución (S) y de permutación (P) permitió construir un esquema de cifrado ligero pero eficaz, capaz de dificultar la identificación de patrones en el texto cifrado. Las funciones diseñadas para este propósito operan sobre bloques de 32 bits, aplicando transformaciones iterativas que incrementan la seguridad del sistema sin requerir recursos computacionales excesivos.

Los resultados experimentales, obtenidos mediante pruebas funcionales entre múltiples clientes, confirman que el cifrado y descifrado se realizan correctamente, sin pérdidas de información, y de manera transparente para los usuarios. Asimismo, se verificó que los clientes que se conectan con posterioridad a la sesión activa pueden recuperar el historial completo de mensajes, manteniendo la coherencia y accesibilidad del sistema.

Este proyecto demuestra que es posible integrar técnicas criptográficas fundamentales en sistemas de comunicación interactivos, incluso en contextos de bajo consumo de recursos. Además, sienta las bases para futuras mejoras, como la incorporación de autenticación de usuarios, cifrado adaptativo o mecanismos de resistencia frente a ataques modernos basados en aprendizaje automático.

REFERENCIAS

- [1] J. Daemen and V. Rijmen, *The Design of Rijndael: AES — The Advanced Encryption Standard*. Berlin, Germany: Springer, 2002.
- [2] National Bureau of Standards, “Data Encryption Standard (DES),” FIPS PUB 46, 1977.
- [3] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. New York, NY, USA: Wiley, 1996.
- [4] C. E. Shannon, “Communication theory of secrecy systems,” *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.
- [5] E. Biham and A. Shamir, “Differential cryptanalysis of the full 16-round DES,” in *Advances in Cryptology — CRYPTO’ 92*, vol. 740, Berlin, Germany: Springer, 1993, pp. 487–496.
- [6] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. Pearson, 2017.
- [7] NIST, “Announcing the Advanced Encryption Standard (AES),” FIPS PUB 197, Nov. 2001.
- [8] C. Cid, S. Murphy, and M. Robshaw, “Small scale variants of the AES,” in *Fast Software Encryption (FSE)*, vol. 2887, Springer, 2003, pp. 145–161.
- [9] T. Ali, M. Khan, and Z. Khan, “Design of dynamic S-boxes based on nonlinear chaotic systems,” *Nonlinear Dynamics*, vol. 76, no. 1, pp. 145–153, 2014.
- [10] J. Hwang, H. Kim, and Y. Lee, “Design of neural S-boxes for block ciphers using machine learning,” in *Proc. IEEE Int. Conf. on Big Data and Smart Computing (BigComp)*, 2020, pp. 449–452.
- [11] M. Wang and X. Liao, “A new chaotic permutation technique for image encryption,” *Signal Processing*, vol. 92, no. 5, pp. 1202–1209, May 2012.
- [12] K. Järvinen, “Efficient instruction set extensions for AES on 32-bit RISC processors,” in *Proc. ACM/SIGDA FPGA*, 2009, pp. 121–130.
- [13] M. Tavares and L. Massey, “A proposed cipher standard: CAST-128,” in *Workshop on Selected Areas in Cryptography*, Springer, 1996.
- [14] R. Huang, J. Li, and W. Yu, “On the vulnerability of substitution-permutation networks against deep learning-based attacks,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1355–1367, 2020.
- [15] N. Al-Kuwari, M. F. Yaseen, and A. R. Al-Ali, “Towards quantum-safe symmetric key cryptography: challenges and future directions,” *Future Internet*, vol. 13, no. 4, p. 96, 2021.