Generación de Claves Criptográficas y Encriptación de Comunicación Mediante Análisis de Métodos Existentes para Mejorar la Seguridad de Sofware de IoT

Bazurto Christopher[1][0009−0005−3529−2095], Drouet Stephen[2][0009−0004−2263−5178], and Pilozo Andy[3][0000−0002−1266−9620]

Departamento de Ciencias de la Computación, Grupo de Investigación en Modelos de Producción de Software (GrIMPSoft), Universidad de las Fuerzas Armadas ESPE, Sangolquí, Ecuador.
{cabazurto,sddrouet,adpilozo}@espe.edu.ec

Abstract. Security in the Internet of Things (IoT) has been compromised due to device resource constraints, making traditional cryptographic algorithms like RSA and AES ineffective. This study presents a system for generating and sharing cryptographic keys using Artificial Intelligence (AI) algorithms to enhance IoT security. The methodology integrates elliptic curve cryptography (ECC) with optimization and AI techniques to generate secure, efficient keys. Experimental results show that the approach improves performance by reducing processing time and resource consumption while increasing cryptographic robustness. Encryption and decryption times are low, with a total latency of 43,319 cycles or 2.67 ms per operation. The key optimization algorithm requires only 31,863 cycles or 1.99 ms per key, reducing the key size from 20 bytes to 8 bytes. However, key generation time exceeds one second, posing a limitation for dynamic key creation. Pre-generation strategies or secure storage are recommended. This work provides an innovative solution to strengthen IoT security.

Keywords: Internet of Things · Artificial Intelligence · Cybersecurity · Encryption

1. Introducción

La tecnología del Internet de las Cosas (IoT) ha experimentado un crecimiento exponencial en las últimas décadas, impulsando la interconexión de dispositivos en áreas clave como la salud, la industria y el hogar [35]. Sin embargo, este avance ha traído consigo una creciente preocupación sobre la seguridad, ya que los dispositivos IoT operan en entornos heterogéneos y con recursos limitados, lo que los hace vulnerables a una amplia gama de ciberataques. En este contexto, diversos estudios han propuesto enfoques criptográficos avanzados para garantizar la protección de los datos manejados por estos dispositivos. Por ejemplo, Huang

et al.[16] sugieren el uso de algoritmos como Modified Paillier-assisted Advanced Encryption Standard (MP-AES) para mejorar la seguridad de las transmisiones ópticas en redes IoT, mientras que Yoon et al. [32] proponen el uso de funciones físicamente no clonables (PUF) para autenticar dispositivos IoT de forma eficiente, lo cual subraya la necesidad de soluciones innovadoras frente a las amenazas actuales.

Alluhaidan y Prabu [4] informan sobre los avances en la criptografía aplicada a IoT y como los algoritmos tradicionales enfrentan serias limitaciones al implementarse en dispositivos con restricciones de memoria y procesamiento, métodos como RSA (Rivest, Shamir y Adleman) y Advanced Encryption Standard (AES), aunque efectivos, requieren un alto costo computacional que resulta ineficiente en dispositivos de bajo consumo. Además, Alluhaidan y Prabu [4] mencionan la creciente sofisticación de los ataques, como los de fuerza bruta, diccionarios, híbridos y aquellos basados en aprendizaje automático, ha evidenciado la necesidad de algoritmos más eficientes y adaptables destacando la urgencia de desarrollar algoritmos ligeros para superar estos desafíos. Mientras que otros trabajos, como el de Yoon et al. [32], proponen la integración de esquemas de autenticación basados en PUF para mejorar la protección contra ataques avanzados.

El objetivo de este trabajo es desarrollar un sistema para la generación y compartición de claves criptográficas, basado en algoritmos de inteligencia artificial (IA), con el fin de mejorar la seguridad en aplicaciones IoT. Este sistema busca superar las limitaciones de los métodos tradicionales al ofrecer una solución eficiente en términos de recursos y altamente resistente a ataques avanzados. La propuesta se inspira en metodologías de optimización y el uso de inteligencia computacional en criptografía, lo cual permitirá crear un mecanismo de generación de claves más seguro y adaptable a los entornos cambiantes de IoT [33]. Además, se aprovecharán técnicas de IA como algoritmos selección para mejorar la distribución y entropía de las claves generadas.

El presente artículo se organiza de la siguiente forma: en primer lugar, se presenta una revisión de los trabajos relacionados con la criptografía en IoT, analizando investigaciones recientes, destacando sus fortalezas y limitaciones. Posteriormente, se describe la metodología propuesta, detallando los algoritmos y su integración en los sistemas IoT. En la sección de evaluación experimental, se comparan los resultados obtenidos con los de métodos tradicionales, analizando aspectos como el tiempo de procesamiento, el consumo de recursos y la robustez criptográfica. Finalmente, se discuten las implicaciones de los resultados y las áreas de mejora, concluyendo con las principales aportaciones de este trabajo para el campo de la seguridad en IoT.

2. Trabajos relacionados

En el contexto de IoT, la seguridad no se limita a la protección criptográfica; abarca también la atestación, la gestión de evidencias, la privacidad, la trazabilidad y la certificación continua. Un ejemplo es el marco “SAFEHIVE”, el cual

permite la atestación de enjambres IoT formados por dispositivos heterogéneos, garantizando la integridad y flexibilidad en redes densas como en ciudades inteligentes sin interrumpir el funcionamiento general [11]. Varios investigadores han propuesto soluciones complementarias: Liu et al. [22] implementan privacidad diferencial basada en el algoritmo de Laplace, mientras que Zhu et al. [35] expanden este concepto incorporando pruebas de conocimiento cero. Liu et al. [21] aportan una perspectiva diferente al introducir un sistema híbrido que utiliza blockchain y tokens de eventos para mejorar la privacidad en la gestión de datos públicos y privados.

El interés por escalabilidad y eficiencia en criptografía también se evidencia en el ámbito blockchain y su intersección con IoT. La utilización de zk-Rollups y dispositivos inteligentes soberanos reduce la carga computacional en las capas base, proporcionando así seguridad y capacidad de procesamiento a gran escala en redes de infraestructura física descentralizada [10]. En paralelo, el uso de Transport Layer Security (TLS) sobre Message Queuing Telemetry Transport (MQTT), un protocolo ampliamente adoptado en IoT, si bien introduce cierta sobrecarga, es manejable y se ve condicionada por las condiciones de la red y el hardware disponible, demostrando que la criptografía aplicada con criterios de selección adecuados es viable en escenarios típicos de IoT [26].

Los protocolos de comunicación seguros en IoT presentan retos significativos, especialmente en términos de eficiencia energética y viabilidad. Estudios como los de Prantl et al. [26] y Bideh et al. [7] analizan el rendimiento del protocolo MQTT con TLS, destacando su impacto en el tiempo de conexión y la eficiencia energética. Prantl et al. [26] abordan estos aspectos desde la perspectiva del consumo energético, mientras que Bideh et al. [7] demuestran que MQTT supera a Constraint Application Protocol (CoAP) en redes con pérdidas, logrando ahorros energéticos de hasta 91 %. Estas comparaciones resaltan cómo las condiciones de la red y la selección adecuada de algoritmos criptográficos son determinantes para su implementación segura, dado que los dispositivos IoT suelen estar limitados en recursos. Además, innovaciones como la autenticación mediante funciones físicas magnéticas no clonables han emergido como soluciones prometedoras, mejorando la precisión y reduciendo los costos computacionales en dispositivos restringidos [17].

La certificación y verificación continua de servicios, junto con la privacidad y seguridad en IoT, representan desafíos clave que requieren soluciones innovadoras y especializadas. Banse et al. [6] proponen un enfoque semántico para la certificación continua en la nube mediante el lenguaje de políticas Rego, permitiendo evaluaciones independientes del proveedor y conformidad con múltiples estándares en entornos de múltiples nubes. En el ámbito industrial, Pennekamp et al. [26] desarrollan una plataforma para el intercambio seguro de parámetros de producción basada en Bloom filters, mientras que Ren et al. [27] presentan un protocolo de aprendizaje federado con encriptación homomórfica y doble enmascaramiento para proteger modelos locales y globales. Adicionalmente, la verificación formal del protocolo Mesh Commissioning Protocol (MeshCoP) en redes

Thread demuestra la importancia de los modelos simbólicos para garantizar propiedades de seguridad antes de la implementación masiva. Herramientas como cpp-tiny-client destacan en la generación de clientes API adaptados a las limitaciones de hardware [29], mientras que soluciones como la de Object Security for Constrained RESTful Environments (OSCORE) grupal, logran comunicación grupal segura con autenticación y protección de datos en entornos IoT de recursos limitados, incluso en escenarios de multidifusión [13].

La autenticación y la gestión de redes distribuidas en entornos IoT y 5G enfrentan desafíos significativos debido a su naturaleza heterogénea, dinámica y restringida. Un enfoque basado en el Swin Transformer como el propuesto por Ai et al. [3] combina redes neuronales convolucionales y transformers para procesar datos estructurados en 2D, logrando mayor precisión y robustez al adaptarse a entornos cambiantes sin depender de umbrales manuales. Por otro lado, la gestión de redes distribuidas y la escalabilidad en infraestructuras de colaboración se fortalecen con soluciones basadas en blockchain. Wang y Ge [31] proponen un modelo de consenso Proof-of-Stake (PoS) mejorado que reduce la centralización al evaluar el comportamiento de los nodos mediante un sistema de clasificación, mientras que Lin et al. [20] introducen el protocolo de consenso Prueba de Mayoría (PoM), optimizando la verificación de transacciones en redes 5G. Estas iniciativas demuestran cómo la autenticación avanzada en capa física y los protocolos de consenso innovadores convergen para ofrecer mayor seguridad y eficiencia en sistemas distribuidos modernos.

La autenticación y la escalabilidad en entornos IoT enfrentan desafíos significativos debido al aumento constante de dispositivos y la complejidad de las redes. Ibrahim et al. [17] presentan Magnetic Physical Unclonable Functions (MAG-PUF), un sistema que utiliza emisiones magnéticas no intencionadas como funciones físicas no clonables, alcanzando una precisión de autenticación superior al 99 %, lo que refuerza la seguridad frente a ataques. Por otro lado, Upadhyay et al. [30] realizan un análisis simbólico formal del protocolo Thread, específicamente del MeshCoP, verificando 78 consultas de seguridad y demostrando su efectividad para autenticar dispositivos no confiables. En cuanto a la escalabilidad, arquitecturas basadas en rollups y pruebas de conocimiento cero distribuyen la carga de cómputo fuera de la cadena principal, aumentando la capacidad de procesamiento sin comprometer la seguridad ni la descentralización [10]. Estas soluciones, desde la autenticación avanzada hasta métodos formales de verificación y tecnologías de escalabilidad, abordan eficazmente los retos de seguridad y rendimiento en redes IoT complejas.

La necesidad de intercambio seguro de datos en tiempo real es otra problemática presente. El cifrado de extremo a extremo adaptado a dispositivos con recursos limitados se convierte en un reto, requiriendo algoritmos ligeros y eficientes que aseguren la privacidad y confidencialidad de las comunicaciones a pesar de las limitaciones energéticas, computacionales y de memoria, por ello, la autenticación End to End (E2E) con algoritmos simétricos ligeros mejora la protección de datos, garantizando comunicaciones seguras incluso en equipos con restricciones

severas [4]. La integración de pruebas de conocimiento cero en protocolos de pertenencia a conjuntos, como Zero-Knowledge Set Membership Proof Protocol (ZSMPP), reduce la sobrecarga computacional y de ancho de banda, reforzando la confidencialidad sin sacrificar eficiencia [34]. Estas innovaciones permiten asegurar el intercambio de datos, verificar identidades y proteger información sensible, respaldando la confiabilidad de la información compartida en ecosistemas IoT.

En entornos IoT, la optimización de recursos y la seguridad son fundamentales debido a las limitaciones de los dispositivos. Un enfoque eficiente para abordar estos desafíos es el uso de la criptografía de curva elíptica (ECC), que ofrece alta seguridad con menor consumo computacional en comparación con RSA. Martínez Calle [23] destaca que los certificados implícitos basados en ECC eliminan la necesidad de intercambiar certificados completos, mejorando la autenticación y reduciendo la sobrecarga en la red, lo que la convierte en una solución ideal para IoT.

La criptografía ligera es una pieza central para asegurar IoT en entornos de recursos limitados. Un nuevo algoritmo de criptografía ultraligero para sistemas Radio Frequency Identification (RFID) llamado SLIM es un cifrado ligero diseñado para el Internet de las Cosas en Salud (IoHT), que equilibra seguridad, simplicidad y bajo costo, protegiendo datos sensibles y garantizando inmunidad ante ataques diferenciales y lineales [1]. La selección del cifrado adecuado es un proceso complejo, por lo cual, Ning et al. [25] usan un marco híbrido multicriterio basado en Criteria Importance Through Inter criteria (CRITIC) y Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) para analizar un total de 10 métodos de cifrado ligeros, concluye que el algoritmo criptográfico Klein es una alternativa óptima para IoHT, dado su balance entre seguridad, rendimiento y eficiencia energética, estas investigaciones muestran que optimizar la elección del cifrado mejora la resiliencia y la vida útil de los dispositivos.

En síntesis, los trabajos relacionados exploran enfoques como las pruebas de conocimiento cero y los algoritmos ligeros adaptados al IoT, entre los que destacan Klein y variantes de AES. Mientras que las versiones ligeras de AES ofrecen alta seguridad, su implementación suele requerir hardware especializado. Por otro lado, Klein se presenta como una alternativa más equilibrada, al combinar simplicidad, eficiencia energética y un nivel de seguridad adecuado, lo que lo hace especialmente apto para dispositivos con recursos limitados. En este contexto, se plantea el desarrollo de un algoritmo de generación de claves criptográficas basado en Klein, aprovechando técnicas de inteligencia artificial para incrementar su adaptabilidad y eficiencia. Asimismo, se considera la incorporación de principios de privacidad diferencial, con el objetivo de fortalecer la protección de datos sensibles durante la generación de claves. Con base en estas ideas, el desarrollo propuesto buscará abordar desafíos clave en entornos IoT, integrando seguridad criptográfica y tecnologías avanzadas de manera innovadora y eficiente.

3. Diseño y Metodología del Sistema

Para la implementación del sistema, se han seleccionado componentes de hardware y software que permiten la recolección, procesamiento y transmisión de datos en un entorno IoT. Estos elementos fueron elegidos considerando accesibilidad, eficiencia energética y compatibilidad con criptografía ligera, asegurando su viabilidad en dispositivos embebidos con recursos limitados.

3.1. Materiales y Herramientas

Para la implementación del sistema, se han seleccionado componentes de hardware y software que permiten la recolección, procesamiento y transmisión de datos en un entorno IoT. Estos elementos fueron elegidos considerando accesibilidad, eficiencia energética y compatibilidad con criptografía ligera, asegurando su viabilidad en dispositivos embebidos con recursos limitados.

El hardware incluye microcontroladores Arduino Uno, reconocidos por su bajo costo y facilidad de programación [8]. Estos dispositivos están conectados a sensores analógicos, como potenciómetros, permitiendo la captación de datos en tiempo real. La comunicación entre ellos se realiza mediante puertos seriales COM1 y COM2.

Para la transmisión y gestión de datos, se ha implementado un broker MQTT basado en Eclipse Mosquitto [5], que facilita la comunicación eficiente entre dispositivos IoT y las capas superiores del sistema. Node-RED [9] se encarga de la integración y orquestación del flujo de datos, optimizando la interoperabilidad del sistema.

El procesamiento se realiza mediante un servidor Flask que utiliza MongoDB [18] para manejar grandes volúmenes de datos no estructurados con alta escalabilidad [19]. Para garantizar la seguridad, se emplea la biblioteca de criptografía ligera ECC [14], que genera claves seguras de manera eficiente, ideal para entornos IoT con limitaciones de recursos. El algoritmo Klein, propuesto por [12], se encarga de la encriptación de los datos utilizando las claves generadas por uECC. Además, se ha implementado una optimización basada en inteligencia artificial que ajusta dinámicamente la selección de claves, mejorando su distribución y entropía sin afectar el rendimiento, lo que refuerza la seguridad sin incrementar la carga computacional.

En conjunto, esta combinación tecnológica permite una arquitectura modular y escalable, adecuada para aplicaciones donde la seguridad y el análisis en tiempo real son críticos, como la monitorización ambiental y la manufactura inteligente.

3.2. Arquitectura del Sistema

La arquitectura presentada combina tecnologías IoT, un broker de mensajería basado en el protocolo MQTT y un backend desarrollado para guardar los datos analíticos de la propuesta. En la Figura 1 se ilustra el esquema general de

esta arquitectura, destacando los flujos de información y la interacción entre los distintos componentes.

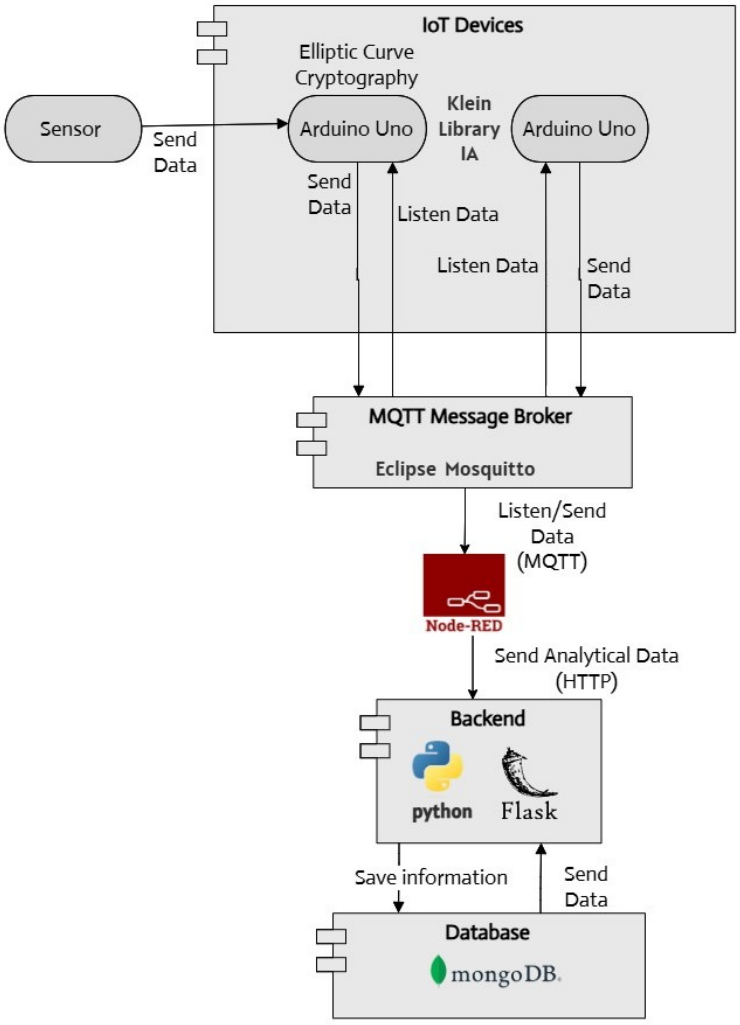

Figura 1. Diagrama de Arquitectura del Sistema.

En este diseño, los dispositivos IoT cumplen un papel fundamental al recolectar información desde sensores conectados a microcontroladores Arduino Uno. Estos dispositivos están equipados con criptografía basada en el algoritmo Klein, lo que garantiza la protección de los datos durante su transmisión. Además, los Arduino Uno facilitan una comunicación bidireccional segura, permitiendo tanto

el envío como la recepción de datos, optimizando la interacción con otras capas del sistema.

El broker de mensajería MQTT, implementado con Eclipse Mosquitto, se encarga de gestionar la distribución de los datos, asegurando una interoperabilidad robusta y una escalabilidad adecuada para redes complejas. Posteriormente, los datos procesados en el broker son estructurados y enriquecidos mediante Node-RED, una herramienta que actúa como intermediario antes de enviar la información al backend. La recolección de datos se realiza en el backend a través de solicitudes HTTP, donde un servidor basado en Flask gestiona y almacena la información en una base de datos MongoDB. Esta base de datos, con su arquitectura NoSQL, optimiza la organización y el manejo de grandes volúmenes de datos no estructurados. El flujo de datos sigue un ciclo continuo que parte desde la recolección inicial en los sensores, asegurando una transmisión cifrada hacia el broker MQTT, hasta su recolección en el backend y el almacenamiento final en MongoDB.

Para complementar la arquitectura IoT, se implementó un sistema de adquisición y transferencia de datos diseñado específicamente para dispositivos de bajo costo y alta accesibilidad, como los Arduino Uno. En la Figura 2 se presenta el esquema de conexiones empleado en este sistema, el cual permite la lectura, transmisión y recepción de datos analógicos a través de microcontroladores y una computadora como intermediario.

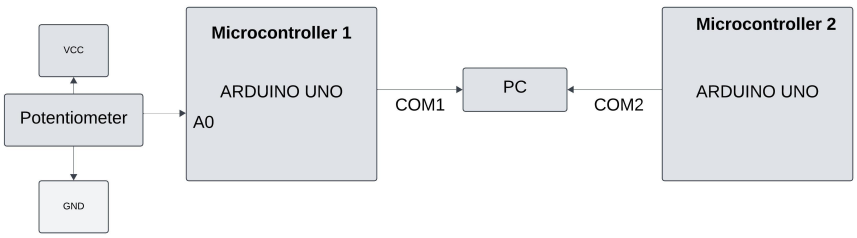

Figura 2. Esquema de conexión entre dispositivos Arduino Uno para adquisición y transferencia de datos.

Para lograr el intercambio de datos seguros el algoritmo Klein requiere una llave de 8 bytes la cual cada uno de los microcontroladores debe poseer, para ello a continuación se procede a explicar este proceso.

3.3. Generación y Compartición de Claves

El proceso de cifrado y descifrado entre dos dispositivos Arduino UNO se basa en la generación de pares de claves públicas y privadas utilizando el algoritmo de criptografía de curvas elípticas ECC. Este esquema permite establecer una clave compartida segura sin necesidad de transmitir claves privadas, mitigando así los riesgos de interceptación. La Figura 3 ilustra el proceso de generación e intercambio de claves entre los microcontroladores.

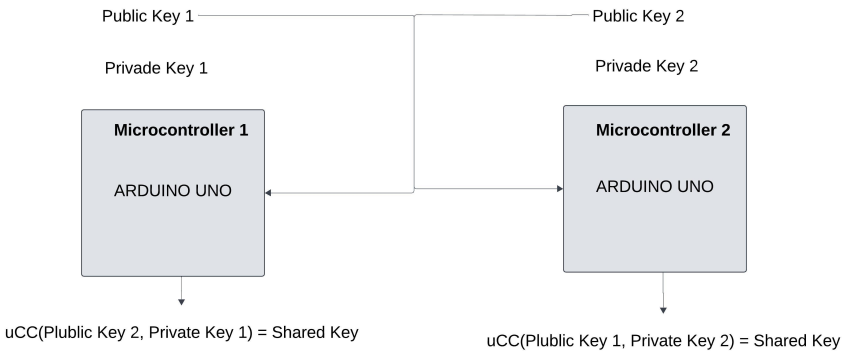

Figura 3. Intercambio de claves entre dos microcontroladores Arduino UNO.

Cada microcontrolador genera su propio par de claves y comparte solo la clave pública con el otro dispositivo. Posteriormente, cada uno calcula la clave compartida combinando su clave privada con la clave pública recibida. La clave compartida se genera mediante el procedimiento de Elliptic Curve Diffie-Hellman (ECDH) [28], adaptado a dispositivos de bajo costo con recursos limitados. Dado que ECC garantiza que ambas claves compartidas sean idénticas, se logra una comunicación segura. Conjuntamente, se aplicó un algoritmo basado en métodos de IA para la optimización de la clave.
Para verificar la coincidencia de las claves compartidas, se utiliza el cifrado Klein. El microcontrolador 2 genera un número aleatorio N, lo cifra con la clave compartida y lo envía al microcontrolador 1. Este último lo descifra, le resta 1, lo vuelve a cifrar y lo envía de regreso. Si el microcontrolador 2 descifra el número y obtiene el valor esperado $(N-1)$, se confirma que ambas claves coinciden; en caso contrario, el proceso se repite. Una vez que los dispositivos tienen la llave compartida puede iniciar la comunicación segura mediante el cifrado klein. El siguiente esquema detalla la generación y validación de claves compartidas mediante operaciones criptográficas basadas en ECC. A continuación, se detalla este

procedimiento mediante fórmulas basadas en teoría de conjuntos, describiendo cada etapa de la interacción y su contribución a la seguridad del sistema.

Definiciones: Se presentan los elementos clave utilizados en el esquema de intercambio de claves mediante ECC. Se definen las claves privadas y públicas de cada microcontrolador, la clave compartida generada, el número aleatorio utilizado para la verificación y la notación para cifrado y descifrado.

- $K_{A_{\text{priv}}}$ y $K_{A_{\text{pub}}}$: Clave privada y pública de A.
- $K_{B_{\text{priv}}}$ y $K_{B_{\text{pub}}}$: Clave privada y pública de B.
- $ECC(K_{\text{priv}}, K_{\text{pub}})$: Operación del algoritmo ECC para generar claves compartidas.
- K_{AB}: Clave compartida generada mediante ECC.
- N: Número aleatorio generado por B.
- $\{x\}_{K_{AB}}$: Encriptación de x utilizando K_{AB}.

Conjunto de Operaciones: Se describen las operaciones matemáticas y los conjuntos de datos involucrados en la generación y el intercambio de claves. Se establecen las claves generadas por cada dispositivo, la forma en que se obtiene la clave compartida y los datos intercambiados durante el proceso.

- Conjunto de claves generadas por A:

$$K_A = (K_{A_{\text{priv}}}, K_{A_{\text{pub}}}) \tag{1}$$

- Conjunto de claves generadas por B:

$$K_B = (K_{B_{\text{priv}}}, K_{B_{\text{pub}}}) \tag{2}$$

- Claves compartidas:

$$K_{AB} = ECC(K_{A_{\text{priv}}}, K_{B_{\text{pub}}}) = ECC(K_{B_{\text{priv}}}, K_{A_{\text{pub}}}) \tag{3}$$

- Conjunto de datos intercambiados:

$$D = (K_{A_{\text{pub}}}, K_{B_{\text{pub}}}, \{N\}_{K_{AB}}, \{N-1\}_{K_{AB}}) \tag{4}$$

Pasos para la generación y compartición de claves Se detalla el procedimiento paso a paso para establecer una clave compartida segura entre dos microcontroladores Arduino UNO. Se incluyen la generación de claves públicas y privadas, el intercambio de claves públicas, la creación de la clave compartida, la verificación mediante cifrado y descifrado de un número aleatorio y la validación final de coincidencia de claves.

1. Cada dispositivo genera un par de claves pública y privada:

$$K_A = \{K_{A_{\text{priv}}}, K_{A_{\text{pub}}}\}, \quad K_B = \{K_{B_{\text{priv}}}, K_{B_{\text{pub}}}\} \tag{5}$$

2. A envía $K_{A_{\text{pub}}}$ a B:

$$B \leftarrow K_{A_{\text{pub}}} \tag{6}$$

3. B genera la clave compartida K_{AB} con ECC utilizando $K_{A_{\text{pub}}}$ y $K_{B_{\text{priv}}}$, luego se optimiza la clave con algoritmo de selección:

$$K_{AB} = OPT(ECC(K_{B_{\text{priv}}}, K_{A_{\text{pub}}})) \tag{7}$$

4. B genera N y lo encripta con K_{AB}:

$$N \in \mathbb{N}, \quad N \rightarrow \{N\}_{K_{AB}} \tag{8}$$

5. B envía $K_{B_{\text{pub}}}$ y $\{N\}_{K_{AB}}$ a A:

$$A \leftarrow (K_{B_{\text{pub}}}, \{N\}_{K_{AB}}) \tag{9}$$

6. A genera K_{AB} usando $K_{A_{\text{priv}}}$ y $K_{B_{\text{pub}}}$, luego optimiza la clave con algoritmo de selección:

$$K_{AB} = OPT(ECC(K_{A_{\text{priv}}}, K_{B_{\text{pub}}})) \tag{10}$$

7. A desencripta $\{N\}_{K_{AB}}$ para obtener N:

$$N \leftarrow \{N\}_{K_{AB}} \tag{11}$$

8. A calcula:

$$N' = N - 1 \tag{12}$$

9. A encripta N' con K_{AB} y lo envía a B:

$$N'_{\text{enc}} = \{N - 1\}_{K_{AB}} \tag{13}$$

$$B \leftarrow \{N - 1\}_{K_{AB}} \tag{14}$$

10. B desencripta $\{N - 1\}_{K_{AB}}$ para obtener $N - 1$:

$$N - 1 \leftarrow \{N - 1\}_{K_{AB}} \tag{15}$$

11. B verifica si:

$$N' = N - 1 \tag{16}$$

Si la igualdad se cumple, la comunicación es segura. En caso contrario, se regenera K_{AB} y el proceso reinicia desde la generación de las claves compartidas, las claves publicas y privadas no se vuelven a generar.

En este apartado, presentamos un enfoque para optimizar la selección de una clave de 8 bytes a partir de una clave original de 20 bytes. El objetivo es reducir la longitud de la clave generada por ECC sin comprometer su entropía, asegurando que la selección sea determinista y eficiente. Para lograr esto, utilizamos una búsqueda heurística basada en puntuaciones, inspirada en técnicas de optimización utilizadas en inteligencia artificial.

La selección de la clave óptima se basa en una función de evaluación que asigna una puntuación a cada subconjunto de 8 bytes según su distribución y entropía. La puntuación se define como:

$$S(K) = \sum_{i=0}^{7} (k_i \times (i+1)) \mod 256 \tag{17}$$

donde k_i representa el i-ésimo byte seleccionado y el índice $i+1$ actúa como un peso para dar más importancia a las posiciones finales.

Para evitar una selección secuencial simple, se introduce un índice inicial basado en un hash determinista de los 20 bytes originales, definido como:

$$H = \left(\prod_{i=0}^{19} (H_{\text{prev}} \times 33) \oplus k_i \right) \mod 20 \tag{18}$$

Esta estrategia garantiza que, para una misma clave de entrada, el algoritmo siempre seleccione el mismo conjunto de 8 bytes, asegurando consistencia en los resultados.

Para mejorar la calidad de la selección, el algoritmo prueba cinco combinaciones distintas, alterando ligeramente el punto de inicio para evitar sesgos en la selección. Finalmente, se elige la clave con la mejor puntuación. Este proceso simula una búsqueda heurística inspirada en IA, optimizando la selección de la clave sin la necesidad de entrenamiento en modelos complejos.

La integración de criptografía ligera en los microcontroladores asegura la confidencialidad de los datos en todas las etapas del sistema, desde su adquisición hasta su almacenamiento. Este enfoque garantiza no solo la protección contra ciberataques, sino también una transmisión eficiente en términos de recursos computacionales, lo que lo hace adecuado para dispositivos con capacidades limitadas.

La combinación de un diseño modular, tecnologías robustas y una arquitectura segura posiciona este sistema como una solución innovadora para entornos donde la protección de datos y el análisis en tiempo real son críticos. Esta metodología puede extenderse a diversas áreas de investigación y aplicaciones prácticas, proporcionando un marco versátil y adaptable para sistemas IoT.

4. Resultados

Para evaluar la eficiencia del esquema criptográfico aplicado en dispositivos IoT, se realizaron mediciones en procesos clave, como la generación de claves, su optimización y las operaciones de cifrado y descifrado. Estos parámetros fueron seleccionados debido a su impacto directo en la seguridad y el rendimiento del sistema.

El análisis de rendimiento se basa en el cálculo del número promedio de ciclos de ejecución para cada operación [24], utilizando la siguiente ecuación:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n} \tag{19}$$

donde X_i representa el número de ciclos registrado en la medición i y n es el número total de mediciones. Para evaluar la dispersión de los valores obtenidos, se empleó la desviación estándar [24]:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}} \tag{20}$$

El tamaño de la muestra necesario para obtener mediciones con un nivel de confianza del 99 % se calculó mediante la fórmula para una muestra infinita [24]:

$$n = \frac{Z_{\alpha/2}^2 \cdot p \cdot (1-p)}{E^2} \tag{21}$$

donde $Z_{\alpha/2} = 2{,}58$ es el valor crítico para un nivel de confianza del 99 %, p es la proporción esperada de éxito en la población, asumida como 0.5 para maximizar el tamaño de la muestra, y E representa el margen de error permitido, establecido en 0.02. Estos valores fueron elegidos para garantizar la precisión en los resultados y obtener mediciones lo más exactas posible. Además, dado que es posible recopilar una gran cantidad de datos, se puede aumentar el número de muestras para mejorar la precisión de los resultados.

Asimismo, para evaluar la latencia total de comunicación, se calculó el retardo combinado de cifrado y descifrado mediante la siguiente ecuación:

$$\text{Latencia} = \text{Ciclos de Cifrado} + \text{Ciclos de Descifrado} \tag{22}$$

Para corroborar los resultados al utilizar un Arduino Uno con un procesador de 16 MHz, se aplicó la siguiente ecuación para obtener el tiempo en milisegundos y compararlo con el tiempo medido:

$$\text{Tiempo (ms)} = \frac{\text{Ciclos} \times 1000}{\text{Velocidad del procesador}} \tag{23}$$

Dado que las mediciones del proceso de encriptación se realizaron para un total de 4 bloques, el número promedio de ciclos de reloj por bloque se obtiene mediante la siguiente ecuación.

$$\#\text{Ciclos por Bloque} = \frac{\text{Latencia}}{4} \tag{24}$$

Estas métricas proporcionan un marco analítico sólido para evaluar la viabilidad del esquema criptográfico en entornos IoT, permitiendo comparar su rendimiento con otros sistemas implementados en trabajos relacionados. De esta manera, se garantiza la obtención de datos precisos y relevantes para la implementación propuesta.

4.1. Resultados Experimentales

Al aplicar la fórmula para el cálculo de muestras en poblaciones infinitas, se determinó que se requieren al menos 4160 datos para garantizar la precisión estadística de los resultados. Este tamaño muestral permite obtener estimaciones confiables sobre el desempeño del sistema, reduciendo el margen de error y asegurando que las mediciones reflejen con precisión el comportamiento del esquema criptográfico en entornos IoT. Los resultados obtenidos en las pruebas de rendimiento permiten evaluar la eficiencia del sistema en términos de tiempo de ejecución y optimización de recursos computacionales.

La Figura 4 presenta el número de ciclos de procesador medidos para las operaciones de cifrado y descifrado de un total de 16 bytes (equivalente a 2 bloques), proporcionando una visión detallada de su costo computacional. En promedio, el cifrado requiere 28,000 ciclos, con una desviación estándar de 4,000 ciclos, lo que corresponde a un tiempo estimado de 1.75 ms, mientras que la medición real arrojó un valor de 1.71 ms. Por su parte, el descifrado demanda 15,300 ciclos, con una desviación estándar de 705 ciclos, resultando en un tiempo estimado de 0.96 ms, en concordancia con el valor medido. En total, la latencia de comunicación asciende a 43,319 ciclos o 2.67 ms, un retardo bajo y adecuado para dispositivos IoT, donde la velocidad en la transmisión de datos es un factor crítico. Además, el número promedio de ciclos de reloj por bloque es de 10800.

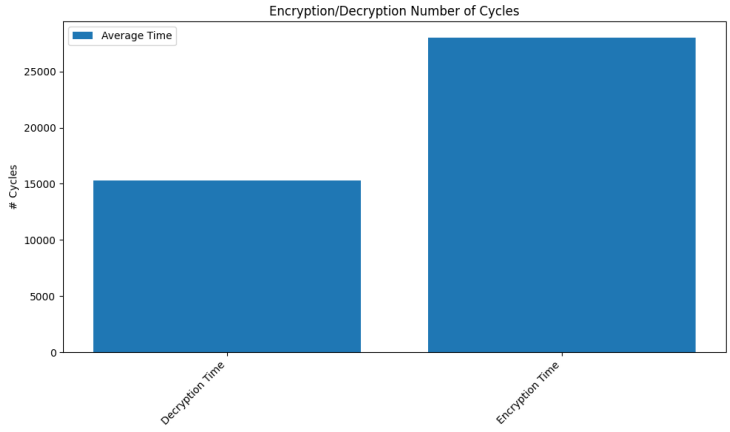

Figura 4. Número de ciclos de cifrado y descifrado.

Por otro lado, la Figura 5 ilustra el número de ciclos necesarios para la generación y optimización de claves, procesos esenciales para la seguridad del sistema. La creación de las claves pública y privada requirió un total de 1,732,037 ciclos (17.3 M ciclos), con un tiempo promedio de 1082 ms y una desviación estándar de 725,775 ciclos (0.72 M ciclos). Por su parte, la derivación de la clave compartida demandó 16,762,656 ciclos (16.76 M ciclos), con un tiempo de 1048 ms y una desviación estándar de 7,514 ciclos (7.51 K ciclos).

Adicionalmente, la optimización de la clave, que consiste en reducir su tamaño de 20 bytes a 8 bytes, presentó un total de 31,863 ciclos (0.03 M ciclos), con un tiempo de procesamiento de apenas 1.99 ms, lo que representa solo el 0.19 % del tiempo total requerido para generar la clave original. Estos resultados demuestran que la optimización tiene un impacto insignificante en el rendimiento global del sistema.

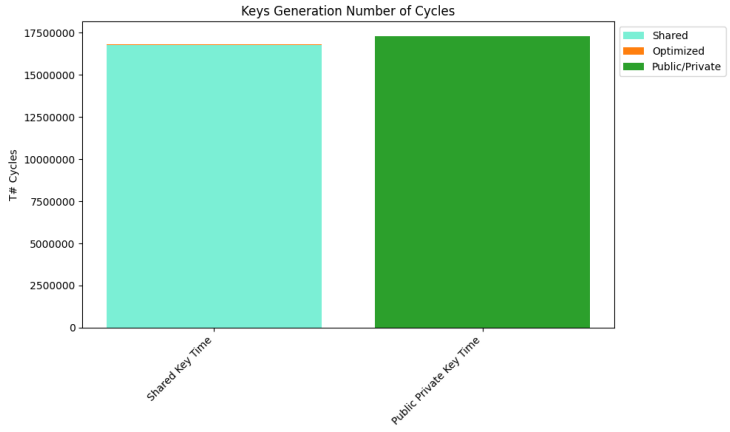

Figura 5. Número de ciclos de generación y optimización de claves.

4.2. Análisis de Resultados

Los datos obtenidos permiten evaluar la viabilidad del esquema criptográfico en dispositivos IoT. Los tiempos de cifrado y descifrado son lo suficientemente bajos para garantizar una comunicación fluida, con una latencia total de 43,319 ciclos o 2.67 ms por operación en el microprocesador Arduino UNO. Esto indica que la sobrecarga introducida por el cifrado es mínima, un aspecto crucial en sistemas donde la transmisión en tiempo real es un requisito esencial.

Por otro lado, el algoritmo de optimización de claves demostró ser altamente eficiente, requiriendo solo 31,863 ciclos (0.03 M ciclos) o 1.99 ms por clave. Este proceso permite reducir el tamaño de la clave compartida de 20 bytes a 8 bytes, disminuyendo la carga en la transmisión y el almacenamiento, lo que mejora la eficiencia del sistema en dispositivos con recursos limitados.

Sin embargo, el tiempo de generación de claves representa un desafío, con valores superiores a un segundo. Esta duración puede ser una limitación en escenarios donde se requiere la creación dinámica de claves para cada sesión. En tales casos, sería recomendable implementar estrategias de pre-generación de claves o almacenamiento seguro para mitigar este impacto.

4.3. Discusión

En el artículo de Palacios y Lechón [2], se analizan diversos algoritmos de cifrado, tanto clásicos como ligeros. Entre estos últimos, se estudian Simon, Present y Speck, destacando la importancia de mantener tiempos de descifrado aceptables. Se menciona que tiempos excesivamente altos pueden generar latencias inadecuadas para ciertas aplicaciones, mientras que tiempos demasiado bajos,

en el orden de microsegundos, pueden hacer que el sistema sea vulnerable a ataques de fuerza bruta. En este contexto, nuestra implementación demuestra ser adecuada, ya que el tiempo requerido para el proceso de descifrado es de 0.96 milisegundos, un valor dentro del rango aceptado por el autor.

Al comparar los resultados obtenidos con estudios previos, se observan discrepancias significativas en el consumo de ciclos de reloj. Hosseinzadeh et al. [15] reportan que Klein tiene un consumo de 17 ciclos por reloj, un valor considerablemente inferior a los 10,800 ciclos obtenidos en nuestra implementación. Sin embargo, al analizar la métrica de ciclos por byte, la diferencia se reduce: mientras que nuestro experimento arroja 1,353 ciclos por byte, el estudio reporta 761 ciclos, lo que sugiere que factores como el hardware utilizado, optimizaciones en la implementación o diferencias en las condiciones de prueba pueden influir en los resultados. De manera similar, Ning et al. [25] informan un consumo de 271 ciclos de reloj por bloque, un valor aún más bajo que el presentado en esta propuesta, lo que refuerza la variabilidad en las mediciones dependiendo del contexto de ejecución. A pesar de estas diferencias, los tiempos obtenidos en nuestra evaluación siguen siendo adecuados para entornos IoT, especialmente en aplicaciones de transmisión en tiempo real. Esto concuerda con los hallazgos de Hosseinzadeh y Hosseinzadeh, quienes resaltan la eficiencia de KLEIN-80 en implementaciones de hardware, aunque sin proporcionar tiempos específicos en software. En este sentido, la latencia total de 2.67 milisegundos obtenida en nuestra prueba sigue siendo competitiva dentro del rango aceptable para dispositivos IoT, manteniendo un equilibrio entre seguridad y eficiencia computacional.

En cuanto a la generación de claves, los resultados muestran tiempos significativamente mayores para ECC, con valores cercanos a un segundo, en comparación con el estudio de Martínez Calle [23], que reporta tiempos entre 1 y 150 milisegundos. Esta diferencia podría deberse a variaciones en el hardware, la complejidad de las operaciones o la variabilidad en los datos. Aunque estos tiempos pueden ser aceptables en entornos IoT con baja sensibilidad a la latencia, la alta dispersión sugiere la necesidad de optimización o el uso de curvas más ligeras. A pesar de que los tiempos de generación de claves son elevados en comparación con certificados X.509 con RSA (350 ms en casos extremos), esto no necesariamente representa una desventaja. En muchos escenarios, la generación de claves solo se realiza al inicio de la comunicación, por lo que estrategias como la reutilización de claves dentro de un período seguro podrían mitigar el impacto de estos tiempos. La viabilidad de esta implementación dependerá, por tanto, del equilibrio entre seguridad, recursos disponibles y requisitos de latencia del sistema.

Conclusiones y Trabajos Futuros

La evaluación realizada confirma la factibilidad del esquema propuesto para la generación y el intercambio de claves criptográficas en entornos IoT con recursos limitados, al equilibrar adecuadamente seguridad y rendimiento. Tras recolectar

4160 datos necesarios para garantizar la precisión estadística, se observó que el proceso de cifrado y descifrado de 16 bytes requiere, en promedio, 28,000 ciclos (1.71 ms) y 15,300 ciclos (0.96 ms), respectivamente, sumando una latencia de 2.67 ms. Estos valores sugieren que el impacto sobre la transmisión de datos en tiempo casi real es mínimo, lo cual resulta esencial en aplicaciones industriales y de infraestructura crítica, donde la velocidad de respuesta es prioritaria.

Por otro lado, la creación de las claves pública y privada mediante ECC presenta un tiempo de 1082 ms, mientras que la derivación de la clave compartida demanda 1048 ms, lo que evidencia un costo computacional significativo en microcontroladores de baja potencia. No obstante, la optimización de la clave, orientada a reducir su longitud de 20 a 8 bytes, se ejecuta en apenas 31,863 ciclos (1.99 ms), representando solo el 0.19 % del tiempo total necesario para la generación de la clave. Ello confirma que el procedimiento de optimización no afecta de manera sustancial el rendimiento global y, simultáneamente, minimiza el uso de recursos en dispositivos con capacidades de hardware restringidas.

Derivado de estos hallazgos, se recomienda profundizar en la adopción de curvas elípticas más ligeras o en la aplicación de técnicas de precomputación para mitigar el elevado costo de la generación de claves, especialmente en escenarios que requieran su renovación frecuente. Asimismo, la incorporación de funciones físicamente no clonables (PUF) representa una vía prometedora para incrementar la unicidad de los dispositivos y reforzar la autenticidad de las transacciones, dificultando ataques de clonación. En paralelo, la integración de algoritmos de aprendizaje automático orientados a la rotación dinámica de claves permitiría adaptar los parámetros criptográficos según el comportamiento de la red y la aparición de nuevas amenazas.

Finalmente, como trabajo futuro se propone explorar estrategias de privacidad diferencial y criptografía de siguiente generación posibilitaría fortalecer la protección de datos sensibles y garantizar la robustez de este esquema en arquitecturas IoT cada vez más heterogéneas. De esta manera, se sientan las bases para un sistema de seguridad integral, escalable y adaptable, que equilibre la demanda de recursos con la necesidad creciente de salvaguardar la confidencialidad e integridad de la información en múltiples dominios de aplicación.

Referencias

1. Aboushosha, B., Ramadan, R.A., Dwivedi, A.D., El-Sayed, A., Dessouky, M.M.: Slim: A lightweight block cipher for internet of health things. IEEE Access **8**, 203747–203757 (2020). https://doi.org/10.1109/ACCESS.2020.3036589
2. Acosta, E.A.P., Gabriel, L.A.V., et al.: Evaluación de algoritmos de criptografía ligera en dispositivos iot de bajo coste. Reincisol. **3**(6), 2935–2961 (2024)
3. Ai, X., Yue, Q., Li, H., Li, W., Tu, S., Rehman, S.U.: Physical layer authentication based on transformer. In: Proceedings of the 2023 13th International Conference on Communication and Network Security. p. 203–208. ICCNS '23, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3638782.3638813, https://doi.org/10.1145/3638782.3638813

4. Alluhaidan, A.S.D., Prabu, P.: End-to-end encryption in resource-constrained iot device. IEEE Access **11**, 70040–70051 (2023). https://doi.org/10.1109/ACCESS.2023.3292829
5. Banks, A., Gupta, R.: Mqtt version 3.1.1. OASIS Standard (2014), disponible en: https://mqtt.org/
6. Banse, C., Kunz, I., Haas, N., Schneider, A.: A semantic evidence-based approach to continuous cloud service certification. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing. p. 24–33. SAC '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3555776.3577600, https://doi.org/10.1145/3555776.3577600
7. Bideh, P.N., Sönnerup, J., Hell, M.: Energy consumption for securing lightweight iot protocols. In: Proceedings of the 10th International Conference on the Internet of Things. IoT '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3410992.3411008, https://doi.org/10.1145/3410992.3411008
8. Community, A.: Arduino uno: A microcontroller board for embedded applications. Arduino Documentation (2021), disponible en: https://www.arduino.cc/
9. Community, N.R.: Node-red: Low-code programming for event-driven applications (2021), disponible en: https://nodered.org/
10. Fan, X., Xu, L.: Towards a rollup-centric scalable architecture for decentralized physical infrastructure networks: A position paper. In: Proceedings of the Fifth ACM International Workshop on Blockchain-Enabled Networked Sensor Systems. p. 9–12. BlockSys '23, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3628354.3629534, https://doi.org/10.1145/3628354.3629534
11. Ferro, L., Bravi, E., Sisinni, S., Lioy, A.: Safehive: Secure attestation framework for embedded and heterogeneous iot devices in variable environments. In: Proceedings of the 2024 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems. p. 41–50. SaT-CPS '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3643650.3658609, https://doi.org/10.1145/3643650.3658609
12. Gong, Z., Nikova, S., Law, Y.: Klein: A new family of lightweight block ciphers. In: Preneel, B., Takagi, T. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2011. Lecture Notes in Computer Science, vol. 6917, pp. 1–18. Springer, Nara, Japan (2011). https://doi.org/10.1007/978-3-642-23951-9_1
13. Gunnarsson, M., Malarski, K.M., Höglund, R., Tiloca, M.: Performance evaluation of group oscore for secure group communication in the internet of things. ACM Trans. Internet Things **3**(3) (Jul 2022). https://doi.org/10.1145/3523064, https://doi.org/10.1145/3523064
14. Gupta, P., Sharma, R.: Implementation of lightweight cryptography in iot using ecc. International Journal of Advanced Computer Science and Applications **11**(5), 117–125 (2020)
15. Hosseinzadeh, J., Hosseinzadeh, M.: A comprehensive survey on evaluation of lightweight symmetric ciphers: hardware and software implementation. Advances in Computer Science: an International Journal **5**(4), 31–41 (2016)
16. Huang, Z., Qin, F., Li, Z.: Iot integration of securable optical transmission using paillier assisted advanced encryption standard. Optical and Quantum Electronics **55**(13), 1140 (2023)
17. Ibrahim, O.A., Sciancalepore, S., Di Pietro, R.: Mag-puf - authenticating iot devices via magnetic physical unclonable functions. In: Proceedings of

the 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks. p. 290–291. WiSec '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3507657.3529656, https://doi.org/10.1145/3507657.3529656
18. Inc., M.: Mongodb: The flexible nosql database (2022), disponible en: https://www.mongodb.com/
19. Leavitt, N.: Will nosql databases live up to their promise? Computer **43**(2), 12–14 (2010)
20. Lin, W., Xu, X., Qi, L., Zhang, X., Dou, W., Khosravi, M.R.: A proof-of-majority consensus protocol for blockchain-enabled collaboration infrastructure of 5g network slice brokers. In: Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure. p. 41–52. BSCI '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3384943.3409421, https://doi.org/10.1145/3384943.3409421
21. Liu, L., Zhu, Y., Li, J.: Hybrid communication and storage system with user privacy preservation for public management, analysis and prediction. In: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking. MobiCom '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3372224.3418164, https://doi.org/10.1145/3372224.3418164
22. Liu, Q., Huang, Y., Jin, C., Zhou, X., Mao, Y., Catal, C., Cheng, L.: Privacy and integrity protection for iot multimodal data using machine learning and blockchain. ACM Trans. Multimedia Comput. Commun. Appl. **20**(6) (Mar 2024). https://doi.org/10.1145/3638769, https://doi.org/10.1145/3638769
23. Martínez Calle, J.J.: Certificados implícitos para autenticación en iot (2021), https://repositorio.uniandes.edu.co/bitstream/1992/55592/1/26335.pdf
24. Montgomery, D.C., Runger, G.C.: Applied Statistics and Probability for Engineers. Wiley, 7th edn. (2022)
25. Ning, L., Ali, Y., Ke, H., Nazir, S., Huanli, Z.: A hybrid mcdm approach of selecting lightweight cryptographic cipher based on iso and nist lightweight cryptography security requirements for internet of health things. IEEE Access **8**, 220165–220187 (2020). https://doi.org/10.1109/ACCESS.2020.3041327
26. Prantl, T., Iffländer, L., Herrnleben, S., Engel, S., Kounev, S., Krupitzer, C.: Performance impact analysis of securing mqtt using tls. In: Proceedings of the ACM/SPEC International Conference on Performance Engineering. p. 241–248. ICPE '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3427921.3450253, https://doi.org/10.1145/3427921.3450253
27. Ren, Y., Li, Y., Feng, G., Zhang, X.: Privacy-enhanced and verification-traceable aggregation for federated learning. IEEE Internet of Things Journal **9**(24), 24933–24948 (2022). https://doi.org/10.1109/JIOT.2022.3194930
28. Research, C.: SEC 1: Elliptic curve cryptography. Standards for Efficient Cryptography Group (SECG) (2009), https://secg.org
29. Springborg, A.A., Andersen, M.K., Hattel, K.H., Albano, M.: cpp-tiny-client: a secure api client generator for iot devices. In: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing. p. 202–205. SAC '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3477314.3508387, https://doi.org/10.1145/3477314.3508387

30. Upadhyay, P., Sharma, S., Bai, G.: Symbolic verification of mesh commissioning protocol of thread. In: Proceedings of the 17th Innovations in Software Engineering Conference. ISEC '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3641399.3641446, https://doi.org/10.1145/3641399.3641446
31. WANG, J., GE, L.: Consensus algorithm of proof-of-stake based on credit model. In: Proceedings of the 2022 4th International Conference on Blockchain Technology. p. 88–94. ICBCT '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3532640.3532652, https://doi.org/10.1145/3532640.3532652
32. Yoon, S., Kim, B., Kang, Y., Choi, D.: Puf-based authentication scheme for iot devices. In: 2020 international conference on information and communication technology convergence (ICTC). pp. 1792–1794. IEEE (2020)
33. Zhang, X., Zhou, Z., Niu, Y.: An image encryption method based on the feistel network and dynamic dna encoding. IEEE Photonics Journal **10**(4), 1–14 (2018)
34. Zhou, Y., Dai, B., Li, C.: A zero-knowledge set membership proof scheme based on the sm2 algorithm. In: Proceedings of the 2024 Guangdong-Hong Kong-Macao Greater Bay Area International Conference on Digital Economy and Artificial Intelligence. p. 510–515. DEAI '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3675417.3675502, https://doi.org/10.1145/3675417.3675502
35. Zhu, W., Li, Y., Wang, W., Zhu, J., Wei, Y.: Data privacy protection method of smart iot platform based on differential privacy. In: Proceedings of the 5th International Conference on Information Technologies and Electrical Engineering. p. 149–154. ICITEE '22, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3582935.3583097, https://doi.org/10.1145/3582935.3583097