Performance Evaluation of Symmetric Cryptographic Algorithms in Resource Constrained Hardware for Wireless Sensor Networkst

Juan C. Gonzalez-Arango, Diana C. Ocampo-Munera, Luis Castano-Londono, David Goez-Sanchez, Ricardo Velasquez-Velez

***Abstract*—Wireless sensor networks (WSN) allow to exchange information and to take immediate and remote actions in natural, industrial, military or domestic environment systems. These networks are vulnerable to cyber-attacks, hence, they are vulnerable to being intercepted, interrupted or modified. However, for the last twenty years, the amount of information exchanged through communication networks around the world has considerably increased and thus its vulnerability. As a result, encryption algorithms are fundamental to protect information. In this context, security, performance and energy consumption become a paramount design factor for an engineer when designing and implementing WSN's. It is not an easy task to reach an optimal balance between these factors. In this paper, we evaluate three well-known symmetric encryption algorithms in an embedded development platform: the ARDUINO MEGA 2560. Our experiment measures the algorithm's encryption average execution time and energy consumption. Additionally, we measure the impact of the channels of a small wireless sensor network with two nodes. The evaluation demonstrates the feasibility of implementing cryptographic algorithms in devices with limited resources such as memory, computing power and life span (energy consumption). Furthermore, the low impact on the network channels when comparing unencrypted communication vs encrypted communication evidence the potential of symmetric encryption algorithms.**

***Index Terms*—Symmetric Cryptographic Algorithms , power consumption, wireless sensor network, resource constrained hardware.**

I. Introduction

Las redes de sensores inalámbricos (WSNs) permiten el intercambio de información y la ejecución de acciones inmediatas y remotas en diferentes tipos de sistemas en entornos naturales, industriales, militares o domésticos. Las WSNs consisten generalmente en varios nodos autónomos que se encuentran dispersos en un entorno específico que debe ser monitoreado y en los cuales se realizan diversas tareas heterogéneas [1]. Las acciones básicas ejecutadas por estos dispositivos suelen consistir en:

Juan C. Gonzalez-Arango, Diana C. Ocampo-Munera, Luis Castano-Londono y David Goez-Sanchez, Departamento de Electrónica y Telecomunicaciones, Instituto Tecnológico Metropolitano, Medellin, Colombia. E-mail: juangonzalez84540@correo.itm.edu.co, dianaocampo24705@correo.itm.edu.co, luiscastano@itm.edu.co, germangoez@itm.edu.co.

Ricardo Velasquez-Velez, Departamento de Electrónica y Telecomunicaciones, Universidad de Antioquia, Medellin, Colombia. E-mail: randres.velasquez@udea.edu.co.

Corresponding author: Luis Castano-Londono (e-mail: luiscastano@itm.edu.co).

- Muestreo de variables ambientales como temperatura, presión, luz, particulación, entre otras.
- Procesamiento de la información recolectada, el cual puede consistir en el cifrado de datos para proteger dicha información.
- Transmisión de la información procesada a otros nodos de la misma red o incluso a otras redes.

El uso de estas redes ha crecido considerablemente en los últimos años [2], y su uso tiende a incrementarse en loa años siguientes de acuerdo con las proyecciones de los diferentes tipos de aplicaciones asociadas al Internet de las cosas (IoT) [3]. Debido a esto, la cantidad de información privada intercambiada o almacenada a través de las redes de comunicación ha aumentado considerablemente y con ello su vulnerabilidad. La información compartida a través de redes públicas y privadas es vulnerable a la interceptación, modificación o interrupción [4], por lo cual se requiere el uso de algoritmos de encriptación para proteger dicha información. En este sentido, en muchos estudios se han desarrollado algoritmos robustos para la encriptación de la información.

Las WSNs son vulnerables a los ataques debido a su despliegue y operación en áreas abiertas [4]. Por esta razón, varios investigadores han centrado su interés en mejorar los aspectos de seguridad de las redes de servicios mundiales. Se han proporcionado diferentes criterios de evaluación para los problemas de seguridad presentados por las redes en términos de las limitaciones de hardware y de seguridad de la información que gestionan, a fin de minimizar el consumo de recursos sin comprometer la seguridad e integridad de los datos y de la red. Por otra parte, con el auge de la Internet de los cosas (IoT) y de las WSNs, algunos esfuerzos se centran en la mejora de los algoritmos criptográficos existentes. La implementación de estos algoritmos puede sobrecargar y causar limitaciones tanto a los nodos como a la red de sensores, utilizando recursos y capacidad de canal debido a la latencia, tráfico, uso de memoria y consumo de energía [1].

Los nodos de una WSN suelen tener recursos limitados en términos de memoria, capacidad de procesamiento y suministro de energía. Esta última es una limitación crítica para este tipo de sistema alimentado por baterías [4], debido al tamaño del nodo y al funcionamiento en entornos donde el cambio o la recarga de la batería se hace difícil. En algunos casos, la batería puede recargarse utilizando métodos de recolección de energía del mismo entorno [5].

Otro aspecto a considerar es la limitada capacidad del

nodo para ejecutar cálculos complejos. Tal es el caso de los nodos dispersos en entornos agresivos o críticos en los cuales es necesario manejar datos sensibles [6]. En este caso, es necesario que el nodo implemente una serie de servicios de seguridad, como autenticación, y encriptación de datos [7]. De esta manera, es posible proteger la información manipulada por los nodos de la red contra ataques externos. Sin embargo, los servicios de seguridad reducen la duración de la batería del nodo debido a la cantidad de energía necesaria para su funcionamiento. Como resultado, el consumo de energía se convierte en una limitación crítica para estos sistemas. Evidentemente, esto supone un reto complejo para el diseñador que necesita encontrar el mejor balance entre el desempeño y la vida útil del sistema.

Estos factores se vuelven relevantes a la hora de elegir el algoritmo de encriptación adecuado, teniendo en cuenta las limitaciones de recursos y los requisitos del sistema. Por esta razón, este trabajo propone implementar diferentes algoritmos criptográficos en un sistema embebido para evaluar el rendimiento de una red de comunicaciones en términos de latencia, uso de memoria y consumo de energía. Este documento está organizado de la siguiente manera: en la sección II se presentan los algoritmos de encriptación ejecutados en una plataforma de hardware embebido. A continuación se presentan los experimentos realizados y los resultados obtenidos. Por último, se presentan las conclusiones y el trabajo futuro.

II. Metodología

Para el desarrollo de este trabajo se eligieron tres algoritmos de encriptación simétrica en lugar de los asimétricos y de compendio de mensajes, ya que son los más eficientes en términos de consumo de energía y capacidad de procesamiento [8], [9]. La principal diferencia entre los tipos de algoritmos es que los simétricos usan una sola clave para encriptar y desencriptar, mientras que el asimétrico usa dos claves, una publica y una privada [10]. Se evalúa el algoritmo Advanced Encryption Standard (AES), el algoritmo Rivest Cipher 4 (RC4) y Blowfish. El AES es considerado uno de los más seguros según la Agencia de Seguridad Nacional (NSA) de los Estados Unidos de América, ya que no se conoce ningún ataque factible contra él, excepto el ataque al canal auxiliar. El RC4 es actualmente la encriptación más utilizada en protocolos de conexiones WiFi en sus modos WEP, WPA, SSL / TLS [11]. Aunque se ha demostrado que su seguridad ha sido vulnerada a nivel de redes de sensores, puede tener vigencia ya que los sensores inalámbricos están activos poco tiempo y en general la comunicación es en una dirección. Blowfish es un algoritmo de encriptación relevante por su seguridad y simplicidad, pero poco eficiente en comparación con otros algoritmos de cifrado como Twofish. Blowfish maneja bloques de 64 bits a diferencia de Twofish que maneja bloques de 128 bits [12]. Bajo circunstancias en que la capacidad de cómputo y tamaño de la información enviada no es tan relevante, manejar bloques de 128 bits es más seguro. Sin embargo, generalmente los sistemas embebidos y las redes de sensores tienen capacidad de información limitada en tamaño, por lo cual resulta conveniente tener equilibrio entre un algoritmo robusto y una baja carga de bits en la red. Adicionalmente, este algoritmo no está patentado, es licencia libre y se encuentra disponible para todos los usos. En la Tabla I se muestran algunas características de los algoritmos de cifrado implementados [13].

A. Sistema Embebido y la Tecnología de Comunicación

El sistema embebido utilizado para implementar los algoritmos de encriptación y desplegar la WSNs fue seleccionado teniendo en cuenta el tamaño, consumo de energía y los recursos de hardware disponibles. La plataforma Arduino Mega 2560 fue elegida a pesar de estar basada en un microcontrolador con menos prestaciones que otras plataformas disponibles en el mercado. Esta plataforma tiene suficiente capacidad de memoria para implementar los códigos de los algoritmos de encriptación y los programas de prueba. También incluye cuatro puertos serie (UART) y una conexión al PC a través de USB, permitiendo la programación, comunicación y realización de pruebas sin interferir con el resto del hardware conectado. Adicionalmente, se implementó una red de pruebas. Para seleccionar la tecnología inalámbrica para el despliegue de esta red de prueba se analizó la capacidad del canal, el ancho de banda, la latencia, el consumo de energía y la cantidad de equipos compatibles. En la Tabla II se muestran algunas características de tres de las tecnologías inalámbricas disponibles para WSN más utilizadas en la actualidad [14].

La comparación muestra que WiFi tiene una capacidad de canal de 54 Mbps y un ancho de banda de 20 MHz, con los valores más altos para estas características entre las tres tecnologías. Tiene un consumo de 330 mA por transmisión y una recepción de 160 mA, que es superior a las de las otras dos tecnologías. Además, tiene una latencia de 3s y soporta hasta 32 nodos. Por otra parte se muestra un consumo excesivo de energía tanto para la transmisión como para la recepción, lo que la convierte en una tecnología ineficiente para las WSNs. Además, el crecimiento de la red en el campo es limitado con un límite de 32 nodos. El Bluetooth tiene una capacidad de canal de 1 Mbps y un ancho de banda de 1 MHz. Tiene un consumo de 40 mA para la transmisión y 0,2 mA para la recepción, por lo que resulta adecuado para los requerimientos de las redes de sensores. Sin embargo, tiene una latencia de 10 s y el nodo coordinador sólo admite conexión con ocho nodos activos de forma simultánea, aunque admite hasta 255 dispositivos en modo estacionado [15]. ZigBee tiene una capacidad de canal de 250 Kbps y un ancho de banda de 1 MHz, con la menor capacidad de canal entre las tres tecnologías. Es adecuado para implementar una WSN, excepto cuando se utilizan datos como vídeo o audio. Su consumo típico de 30 mA para la transmisión y 0,003 mA para la recepción lo convierte en el mejor en términos de consumo de energía, un aspecto clave en las WSNs. Además soporta hasta 65535 nodos, con un retraso en la comunicación variable dependiendo de la topología implementada [14].

La tecnología que mejor se adaptó a las necesidades de la implementación realizada es Zigbee, ya que tiene

TABLA I
CARACTERÍSTICAS DE LOS ALGORITMOS ESTUDIADOS.

Característica	AES	RC4	Blowfish
Longitud de la clave	128, 192 Y 256 bits	Variable de 40 hasta 2048 bits	Variable de 32 a 448 bits
Tamaño de bloque	128, 192 o 256 bits	N/A	64 bits
Creación	2002	1987	1993
Rondas	10 (128 bits), 12 (192 bits), 14 (256 bits)	Variable de 1 a 256	16
Vulnerabilidad conocida	Ataques de canal auxiliar. Ataques de fuerza bruta por 7, 8 y 9 rondas	Ataques de texto plano. Fallas en el algoritmo KSA	Ataques diferenciales de segundo orden (4 rondas). Ataque SWEET32

TABLA II
CARACTERÍSTICAS DE LAS TRES TECNOLOGÍAS INALÁMBRICAS MÁS UTILIZADAS ACTUALMENTE PARA WSN .

Tecnología inalámbrica	WIFI (802.11)	BLUETOOTH (Clase 1)	ZIGBEE (802.15.4)
Nodos	32	Combinación AMA y PMA hasta 256 radios, solo 8 en el mismo instante	65535/255 x SUB NET
Consumo en TX	400 mW	100 mW máximo	50 mW
Velocidad de TX	54 Mbps	1 Mbps	250 Kbps

un consumo de transmisión inferior al 90,9% respecto a Wi-Fi y un 25% inferior al de Bluetooth. ZigBee permite deshabilitar la funcionalidad de encriptación (AES de 128 bits), tiene una buena respuesta en modo de bajo consumo, es autoconfigurable, y tiene modulación de espectro ensanchado por secuencia directa (DSSS). Además, permite enrutamiento, diferentes topologías de red y transmisión de datos a largas distancias.

B. Algoritmos de Cifrado

En esta sección se describen los algoritmos de cifrado seleccionados en este trabajo: AES, RC4 y Blowfish. Los tres tienen en común que transforman un mensaje sin prestar atención a su estructura lingüística ni a su significado, para que este no sea comprensible o difícilmente comprensible para cualquier persona que no disponga de la clave de descifrado del algoritmo.

1) Estándar de Cifrado Avanzado (AES): Originalmente conocido como Rijndael, lleva el nombre de sus desarrolladores. Rijndael es el ganador del concurso AES organizado por el National Institute of Standards and Technology US (NIST) para elegir al sucesor del algoritmo de encriptación DES, cuando este mostró debilidades en la seguridad [16].

AES es un algoritmo de cifrado de clave simétrica que cifra bloques de 128, 192 o 256 bits utilizando longitudes de clave de 128, 192 o 256 bits. El algoritmo consiste en una primera ronda llamada AddRoundKey y entre 10, 12 o 14 rondas dependiendo del tamaño del bloque y de la longitud de la clave. Las primeras rondas n-1 son similares y realizan cuatro transformaciones: sustitución de bytes, rotación de filas, multiplicación/mezcla de columnas y finalmente una XOR con clave redonda o AddRoundKey [17].

El objetivo de la sustitución de bytes es hacer que el AES sea resistente a los ataques criptográficos lineales y diferenciales. Para ello, el bloque de 128 bits se organiza como una matriz de 4x4 bytes, cada uno de los cuales es sustituido por un byte según una tabla conocida como búsqueda S-box. La rotación de las filas opera sobre una matriz 4x4 y permite que los bits sean difundidos entre múltiples rondas. El propósito de la multiplicación/mezcla de columnas es también distribuir bits y consiste en aplicar una transformación lineal a la matriz invertible resultante del paso anterior. Finalmente, la transformación AddRoundKey consiste en poner la XOR entre la matriz y la llave redonda organizada como una matriz 4x4. Las claves redondas se derivan de la clave de cifrado mediante un algoritmo de planificación de claves. AES requiere una clave de 128 bits para cada ronda más una adicional. Durante la última ronda sólo se realizan transformaciones de bytes de reemplazo y filas de rotación de AddRoundKey [18].

Para descifrar el texto, el procedimiento seguido es exactamente el opuesto al proceso de cifrado. Por lo tanto, una ronda estándar consiste en bytes de transformación que sustituyen a las filas invertidas, a las filas de rotación inversa, a la multiplicación y a las columnas AddRoundKey inversas. La ronda final realiza el mismo proceso excepto la columna de multiplicación inversa [18].

AES se utiliza con diferentes modos de operación dependiendo del escenario de aplicación, siendo los más populares el modo de Cadena de Bloques de Cifrado (CBC) y el Modo de Contador de Enteros (ICM o CTR). El modo CBC es una XOR entre cada bloque de texto plano con el texto cifrado anterior antes de la encriptación. Por lo tanto, cada texto cifrado depende de todos los bloques de texto plano procesados hasta ese momento. El modo CTR utiliza el algoritmo AES para cifrar un bloque arbitrario llamado contador y luego hace la XOR con el texto plano para producir el texto cifrado. Los incrementos del contador se realizan después de cada bloque de datos procesado. Por esta razón, el texto cifrado es diferente incluso si los bloques de datos son idénticos, evitando los ataques mediante la observación de patrones de repetición en el cifrado de texto [19]. En este trabajo el modo de operación utilizado es CBC.

2) Cipher 4 or RC4: RC4 es un algoritmo de cifrado desarrollado en 1987 por Ronald Rivest para RSA Data Security Inc. El algoritmo fue mantenido bajo secreto comercial hasta septiembre de 1994, cuando fue filtrado anónimamente en Internet [20].

RC4 es un algoritmo de cifrado simétrico de flujos de

datos que utiliza una clave de longitud variable de 1 a 256 bytes (8-2048 bits). Su función es producir una cadena de bits pseudo-aleatorios que se combinan con el texto plano utilizando operaciones XOR para el cifrado [21]. El algoritmo RC4 utiliza una tecla para inicializar un vector de estado S de 256 posiciones. Si una aplicación requiere el uso de una clave más corta, entonces esa clave se repite tantas veces como sea necesario para completar 2048-bits. Cada elemento de S almacena un valor de 8 bits que va de 0 a 255. Estos valores corresponden a diferentes permutaciones de 8 bits. El algoritmo RC4 se compone del Algoritmo de Programación de Claves (KSA) para inicializar y permutar los elementos del vector S, y el Algoritmo de Generación Pseudo Aleatorio (PRGA) que utiliza el vector de estado S para generar una cadena de bits pseudo-aleatorios [11].

La mayoría de las operaciones utilizadas en el RC4 funcionan con el tipo de datos *byte*. Este algoritmo también utiliza permutaciones aleatorias. Cada byte de salida requiere de ocho a dieciséis operaciones, por lo cual el algoritmo puede ejecutarse de forma bastante eficiente con el software [22].

RC4 es uno de los algoritmos de encriptación más utilizados. El algoritmo es bastante simple, eficiente y rápido. Además, RC4 utiliza entre otros protocolos de aplicación WEP, WPA, TKIP, SSL / TLS, Secure Shell, y Remote Desktop Protocol. Se ha demostrado que el RC4 no es seguro en modo WEP, y la mayoría de las debilidades descubiertas están relacionadas con el algoritmo KSA [11].

3) Blowfish: Blowfish es un algoritmo de encriptación de clave asimétrica desarrollado en 1994 por Bruce Schneier [2]. Se trata de una red Feistel y de un cifrado en bloque de clave secreta con un tamaño de bloque de 64 bits y 16 rondas. La clave puede ser de cualquier longitud desde 32 hasta 448 bits. Las entradas en este algoritmo son el texto plano y la clave secreta, el texto plano se divide en dos partes de 32 bits cada una. La encriptación de datos se realiza a través de una red Feistel de 16 rondas y utiliza un gran número de S-boxes dependientes de la clave. Cada ronda consta de una permutación dependiente de clave y una sustitución dependiente de clave y datos. Blowfish utiliza muchas subclaves que deben ser precalculadas antes de cualquier encriptación o descifrado de datos. Por esta razón, hay una subclave de 18 entradas P y cuatro subclaves S-boxes de 256 entradas.

C. Montaje Experimetal

Los experimentos fueron realizados a partir de la implementación de versiones estándar de los algoritmos de encriptación seleccionados en el sistema de desarrollo [23][24][25]. Para los tres algoritmos se utilizó una longitud de clave de 256 bits y se encriptó texto plano de 17 bytes 1024 veces. En cada caso, se realizaron las siguientes actividades:

1) Compilación y verificación de la correcta ejecución de los algoritmos en la MCU usando el IDE de Arduino.
2) Ejecución de algoritmos para inicializar las claves y cifrar/descifrar un conjunto de datos determinado para estimar el tiempo de ejecución de cada operación.
3) Recopilación de datos utilizando una de las interfaces de entrada/salida del sistema de desarrollo.

El software está diseñado para tomar la clave de inicialización, tiempos de encriptación y descifrado. También mide el tiempo que tarda el mensaje en ser enviado de forma inalámbrica vía radio para calcular la latencia del canal cuando se compara el envío del texto cifrado con otro texto plano no cifrado.

El montaje experimental consistió en la implementación una red de comunicación inalámbrica usando módulos del estándar IEEE 802.15.4 Zigbee. Cada módulo Zigbee se conecta a un sistema Arduino que se encarga de ejecutar los algoritmos de cifrado y medir la latencia correspondiente a la transmisión del bloque de datos cifrados usando la red inalámbrica. Cada bloque de datos es cifrado en AES, RC4 o Blowfish, empaquetado usando el modo API del IEEE 802.15.4 y trasmitido por el módulo de comunicación. El receptor realiza el proceso inverso, es decir, recibe la trama API, extrae los datos y aplica el método de descifrado. Al finalizar este proceso, se mide el tiempo y se compara con el tiempo inicial del proceso. Además, se mide la potencia y el consumo de energía durante todo el proceso transmisión y recepción. Para calcular el consumo de energía, se utilizaron dos multímetros de comunicación serie UNI-T 61C con el fin de obtener una aproximación de los valores RMS y hacer una comparación cualitativa en las mediciones de corriente y voltaje, y el software ULTRADMM. Este último permite la captura simultánea de las medidas de los dos multímetros y proporciona el tiempo transcurrido.

III. Resultados y Discusión

Los resultados se muestran para cada algoritmo y para cada fase del proceso (inicialización de clave, cifrado, transmisión, recepción, descifrado). En la Tabla III se muestran los tiempos de inicialización de clave (T_{KEYIN}), de encriptación (T_{ENC}), de descifrado (T_{DES}) y total (T_{TOTAL}) para cada uno de los algoritmos implementados en el sistema de desarrollo.

TABLA III
TIEMPOS DE EJECUCIÓN DE LOS ALGORITMOS EN MILISEGUNDOS.

	AES	RC4	Blowfish
T_{KEYIN}	0,314±3,1E-06	0,817 ± 3,5E-06	72,577±1,62E-03
T_{ENC}	592,991±9,4E-05	64,488±8,2E-05	36,300±5,49E-01
T_{DEC}	874,081±2,9E-05	68,655±3,3E-05	44,015±0,89E-01
T_{TOTAL}	1467,386	133,960	152,891

En Fig. 1 se muestra el tiempo de inicialización de la clave. El tiempo promedio de inicialización de clave para el algoritmo AES es 313,73 μs, para RC4 es 816,67 μs y para el Blowfish 72.577 ms.

En Fig. 2 se observa que cifrando el texto sin formato el rendimiento del algoritmo RC4 en términos de tiempo de ejecución es mejor que para AES aproximadamente nueve (9) veces. Adicionalmente, se encuentra que el tiempo de ejecución para Blowfish es menor que el de RC4 casi dos (2) veces, presentando el mejor rendimiento en cuanto a la velocidad de procesamiento para la etapa de cifrado. Para descifrar el texto, se observa que el algoritmo RC4 se ejecuta en menos tiempo que AES casi trece (13) veces, y Blowfish

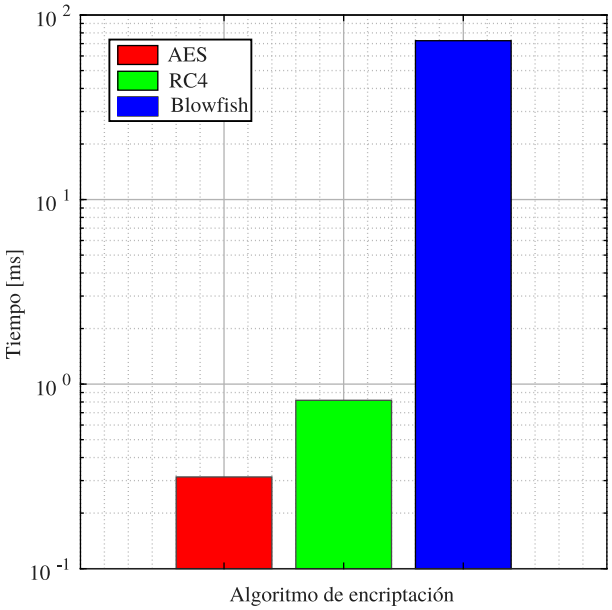


Fig. 1. Tiempos de inicialización de clave de los algoritmos en milisegundos.

es más rápido que AES aproximadamente veinte (20) veces y que RC4 una vez y media (1.5). Con relación al tiempo total, correspondiente a la suma de los tiempos de inicialización de clave, cifrado y descifrado de cada algoritmo, RC4 se ejecutó en menos tiempo que AES aproximadamente once (11) veces y supera a Blowfish por 19 ms.

Los tiempos de encriptación y descifrado son muy similares entre sí en el algoritmo RC4, con una variación de aproximadamente 4 ms cada etapa. Al contrario, AES muestra una diferencia entre los tiempos de cifrado y descifrado de casi 300 ms. Los tiempos de encriptación y descifrado en Blowfish muestran una diferencia de 7,715 ms.

La Tabla IV presenta los tiempos de transmisión tanto para texto plano sin cifrar como para texto cifrado con los tres algoritmos. Se muestra que los tiempos de transmisión con encriptación o transmisión no encriptada son similares, con una desviación estándar de entre 0,289 y 4,54 microsegundos.

TABLA IV
TIEMPOS DE TRANSMISIÓN.

Algoritmo	Tiempo de transmisión promedio [s]	Desviación estándar
Sin encriptación	1,100359	2,79573E-06
AES	1,100361	3,79958E-06
RC4	1,100374	4,54657E-06
Blowfish	1,101252	2,89501E-05

Los tiempos de la Tabla IV y el tamaño de la información encriptada indican que los algoritmos simétricos no añaden carga a la red porque el texto cifrado tiene la misma longitud en bytes que el texto sin formato. Además, el canal de comunicación no está sobrecargado con claves, ya que las mismas se mantienen privadas. Este resultado es importante porque en las redes de sensores el envío de datos se convierte en un consumo crítico, teniendo en cuenta que la mayor parte del consumo de energía se debe más a la transmisión que a la adquisición y procesamiento de datos [26]. En los casos evaluados, la utilización de un algoritmo de encriptación añade tiempo y consumo de energía en la etapa de procesamiento, pero en la transmisión de datos no habrá aumento en el tiempo de transmisión y en el consumo de energía derivado de la encriptación.

En cuanto a la demanda de corriente al ejecutar los algoritmos se observa que es similar en los tres casos al rededor de 5 mA, mientras que en funcionamiento normal la demanda de corriente es de 3 mA. La demanda de corriente mientras se ejecuta el algoritmo se muestra en 3a para AES, en 3b para RC4, y en 3b para Blowfish. En cada caso se observa el incremento en el valor de la corriente durante la ejecución de los algoritmos. También se puede ver que en la implementación del algoritmo AES mientras se mantiene el pico de corriente (duración de la ejecución de algoritmos de encriptación y descifrado) es mucho mayor en el algoritmo AES que en el RC4 o Blowfish.

La Tabla V muestra el consumo de potencia durante la operación del procesador sin carga de algoritmos de cifrado y con carga de algoritmos de cifrado.

TABLA V
CONSUMO DE POTENCIA.

	Potencia promedio [mW]	Desviación [mW]
Procesador sin ejecución de algoritmo de cifrado	14,50	0,212
Con ejecución AES	19,85	0,374
Procesador sin ejecución de algoritmo de cifrado	14,43	0,129
Con ejecución RC4	19,87	0,680
Procesador sin ejecución de algoritmo de cifrado	14,40	0,129
Con ejecución Blowfish	19,86	0,680

La potencia requerida para el proceso de encriptación es cercano entre los tres algoritmos, siendo la diferencia entre AES y RC4 es del 0,1%. Respecto al funcionamiento del sistema embebido sin ejecutar el proceso de encriptación, tiene un incremento de aproximadamente 37% en el consumo de energía. Lo que indica, que encriptar la información se debe realizar solo cuando la aplicación o los requerimientos de seguridad lo ameriten. La Fig. 4. muestra la potencia y el consumo de energía durante la ejecución de los algoritmos de encriptación. El consumo medio de energía de AES durante la ejecución del cifrado y descifrado es de 0,351 julios, mientras que en para RC4 es de 0,066 julios y para Blowfish 0.10 julios. El consumo de energía es mayor cuando se implementa el algoritmo AES aproximadamente cinco (5) veces con relación a RC4 y aproximadamente tres (3) veces con relación a Blowfish.

Como discusión final, se puede decir que las redes de sensores están pensadas para aplicaciones que requieren baja velocidad de transferencia de datos, mientras que el suministro de energía debe ser de meses sin requerir cambio de fuente de energía. En este caso, implementar técnicas de seguridad de la información demanda recursos extras en los dispositivos, mayor tiempo de procesamiento y consumo energético. Los

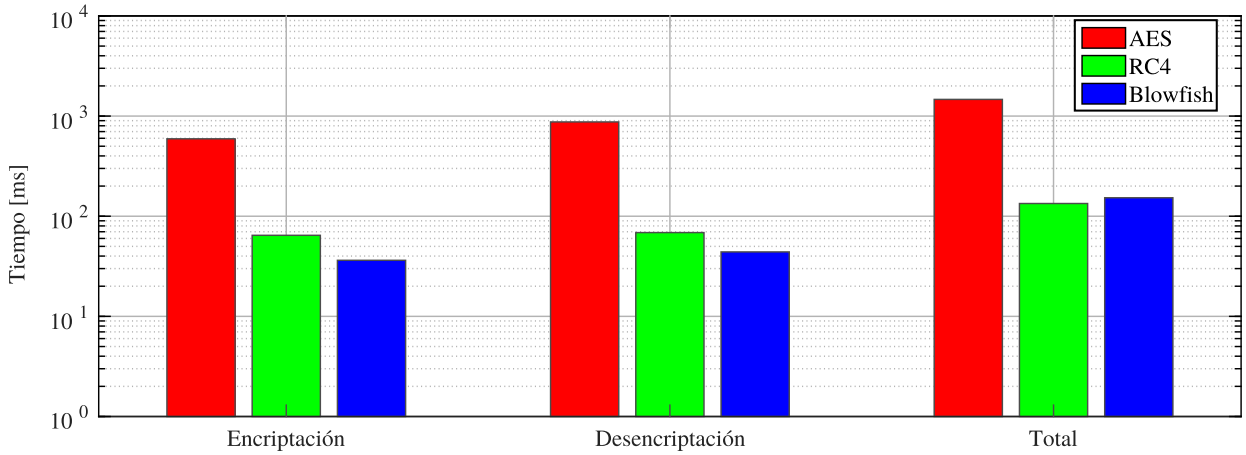


Fig. 2. Tiempos en milisegundos para encriptación y descifrado con cada algoritmo.

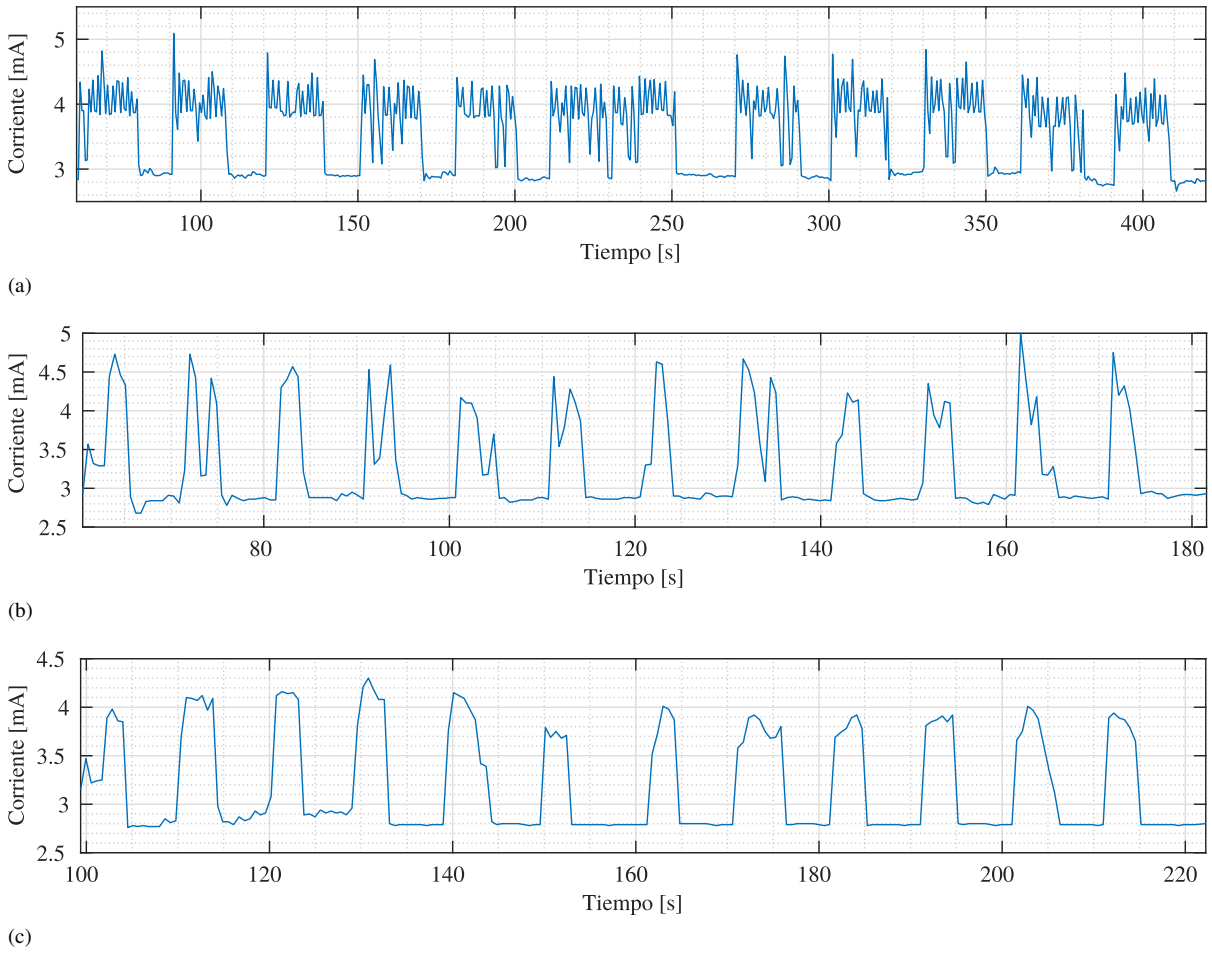


Fig. 3. Corriente eléctrica para a) AES, b) RC4, y c) Blowfish.

resultados obtenidos en este trabajo muestran que es posible usar algoritmos de cifrado simétricos sobre redes de sensores inalámbricos con capacidad reducida, ya que el tamaño de información encriptada es el mismo que la información original sin encriptar. Por tanto, la implementación de este tipo de seguridad en redes de sensores no sobrecarga la red con información muerta.

Respecto a otros trabajos reportados en la literatura [27][28][29], se confirma la viabilidad de implementar este tipo de algoritmos para aumentar la seguridad de la información. Además, respecto a estos trabajos se analizó el consumo energético del sistema integrado entre el Arduino y el radio Zigbee y la factibilidad de implementar algoritmos de encriptación simétricos al no aumentar el trafico en la red. Por tanto, es un aporte a la implementación en redes de sensores con aplicaciones de IoT, con prestaciones reducidas.

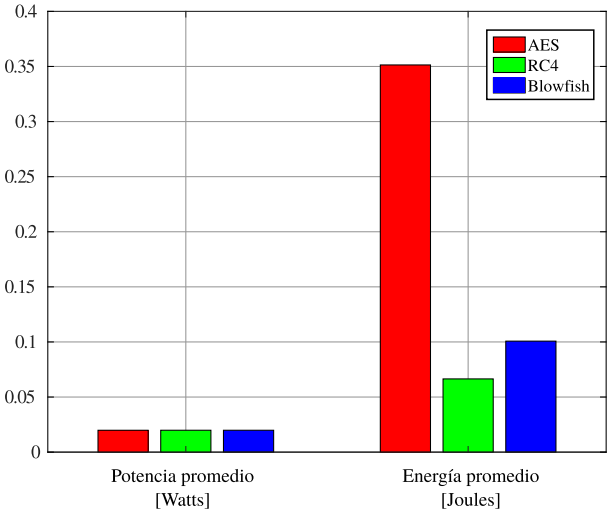


Fig. 4. Potencia y energía promedio para AES, RC4, y Blowfish.

IV. Conclusiones

En este trabajo se realizó la evaluación del rendimiento de los algoritmos de cifrado simétrico AES, Blowfish y RC4, como los utilizados en las WSNs de capacidad reducida. Los resultados permiten concluir que en el caso de los algoritmos simétricos usados no se genera tráfico extra al canal de comunicaciones, ya que la clave no es enviada. La clave es privada y conocida solo por el transmisor y el receptor, por lo que no hay diferencia en el envío de encriptación con claves de 128, 192 o 256 bits. En este trabajo, las claves usadas fueron de 256 bits y texto plano de 17 bytes, dando como resultado un texto cifrado con el mismo número de bytes. Por lo tanto, para la transmisión es completamente transparente si se envía el texto plano o el texto cifrado, ya que ambos tienen el mismo número de bytes.

La medición del consumo de energía de los tres algoritmos, muestra que al realizar la encriptación aumentaron los requisitos de potencia de forma similar para los tres algoritmos. Sin embargo, debido al mayor tiempo de ejecución el algoritmo AES consume más energía respecto a RC4 y Blowfish. Este último consume menos energía que AES y RC4 en el proceso de cifrado y descifrado, lo que permite concluir que Blowfish es más eficiente en el consumo de energía que AES y RC4. El aumento de la demanda de energía en una WSN cuando se realiza encriptación, permite concluir que es recomendable encriptar la información si los requerimientos de seguridad lo ameritan, además de tener en cuenta el tiempo de carga de las baterías.

El trabajo futuro incluye la evaluación de otros algoritmos que puedan generar un mejor rendimiento y la implementación de mecanismos de auto adaptación para optimizar el consumo de energía, y la evaluación de Blowfish en microprocesadores de 32 bits.

Acknowledgment

Este trabajo fue apoyado por el grupo de investigación AEyCC (COL0053581) del Instituto Tecnológico Metropolitano a través del proyecto "Caracterización de algoritmos de cifrado orientados a nodos en redes de sensores inalámbricas con la capacidad de auto-adaptar su configuración de seguridad de acuerdo a las condiciones del entorno" - P14208.

References

[1] M. Ruiz, E. Alvarez, A. Serrano, and E. Garcia, "The convergence between wireless sensor networks and the internet of things; challenges and perspectives: A survey," *IEEE Latin America Transactions*, vol. 14, no. 10, pp. 4249–4254, 2016.

[2] F. E. Murphy, E. Popovici, P. Whelan, and M. Magno, "Development of an heterogeneous wireless sensor network for instrumentation and analysis of beehives," in *2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*. IEEE, 2015, pp. 346–351.

[3] J. Botero-Valencia, L. Castano-Londoño, and D. Marquez-Viloria, "Trends in the internet of things." *TecnoLógicas*, vol. 22, no. 44, pp. 1–2, 2019.

[4] M. Bhalla, N. Pandey, and B. Kumar, "Security protocols for wireless sensor networks," in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*. IEEE, 2015, pp. 1005–1009.

[5] F. A. Aoudia, M. Gautier, and O. Berder, "Rlman: an energy manager based on reinforcement learning for energy harvesting wireless sensor networks," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 2, pp. 408–417, 2018.

[6] Y. Xiaomei and M. Ke, "Evolution of wireless sensor network security," in *2016 World Automation Congress (WAC)*. IEEE, 2016, pp. 1–5.

[7] M. Vai, D. Whelihan, N. Evancich, K. J. Kwak, J. Li, M. Britton, J. Foley, M. Lynch, D. Schafer, and J. DeMatteis, "Systems design of cybersecurity in embedded systems," in *2016 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, 2016, pp. 1–6.

[8] M. Panda, "Data security in wireless sensor networks via AES algorithm," in *2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*. IEEE, 2015, pp. 1–5.

[9] K. Antonopoulos, C. Petropoulos, C. P. Antonopoulos, and N. Voros, "Security data management process and its impact on smart cities' wireless sensor networks," in *2017 South Eastern European Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. IEEE, 2017, pp. 1–8.

[10] J. C. Mendoza, "Demostración de cifrado simétrico y asimétrico," *Ingenius: Revista de Ciencia y Tecnología*, no. 3, pp. 46–53, 2008.

[11] B. Crainicu and A. Gergely, "New cryptanalytic results on the effect of invariance weakness in the ksam mechanism," in *2017 16th RoEduNet Conference: Networking in Education and Research (RoEduNet)*. IEEE, 2017, pp. 1–6.

[12] D. D. Rane, "Superiority of Twofish over Blowfish," vol. 4, no. 11, pp. 4744–4746, 2016.

[13] M. Ebrahim, S. Khan, and U. B. Khalid, "Symmetric algorithm survey: a comparative analysis," *arXiv preprint arXiv:1405.0398*, 2014.

[14] F. Toapanta and T. Guarda, "Implementación de un gateway para protocolos en la norma 802.15. 4 utilizando open source y la placa de desarrollo de hardware (raspberry)," *Revista Ibérica de Sistemas e Tecnologias de Informação*, no. E24, pp. 208–220, 2019.

[15] J. Loo, J. Mauri, and J. Ortiz, "Mobile ad hoc networks," *Mobile Ad Hoc Networks: Current Status and Future Trends*, 2012.

[16] M. A. Alomari, K. Samsudin, and A. R. Ramli, "A study on encryption algorithms and modes for disk encryption," in *2009 International Conference on Signal Processing Systems*. IEEE, 2009, pp. 793–797.

[17] T. Jamil, "The rijndael algorithm," *IEEE potentials*, vol. 23, no. 2, pp. 36–38, 2004.

[18] N.-F. Standard, "Announcing the advanced encryption standard (AES)," *Federal Information Processing Standards Publication*, vol. 197, no. 1-51, pp. 3–3, 2001.

[19] C. Luo, "A simple encryption scheme based on wimax," in *2009 International Conference on E-Business and Information System Security*. IEEE, 2009, pp. 1–4.

[20] M. L. Chávez and F. R. Henríquez, "Sdl specification of a security architecture for worldfip," in *14th International Conference on Electronics, Communications and Computers, 2004. CONIELECOMP 2004.* IEEE, 2004, pp. 149–154.

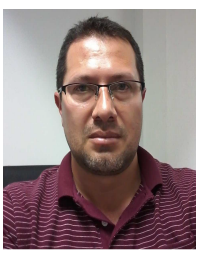
[21] B. Bakhache, J. Ghazal, and S. El Assad, "Enhancement of zigbee and wi-fi security by a robust and fast chaotic algorithm," in *2011 5th International Conference on Network and System Security*. IEEE, 2011, pp. 300–304.

[22] S. Banerjee, D. Sethia, T. Mittal, U. Arora, and A. Chauhan, "Secure sensor node with raspberry pi," in *IMPACT-2013*. IEEE, 2013, pp. 26–30.
[23] B. Schneier, "Schneier on Security," 2017. [Online]. Available: https://www.schneier.com/academic/blowfish/download.html
[24] A. MBED, "AES source code," 2016. [Online]. Available: https://tls.mbed.org/aes-source-code
[25] S. Sriadhi, R. Rahim, and A. S. Ahmar, "RC4 Algorithm Visualization for Cryptography Education," *Journal of Physics: Conference Series*, vol. 1028, no. 1, 2018.
[26] W. He, S.-H. Yang, L. Yang, and P. Li, "In-network data processing architecture for energy efficient wireless sensor networks," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE, 2015, pp. 299–304.
[27] A. Sukiatmodjo, "Speed and power consumption comparison between DES and AES algorithm in arduino," Ph.D. dissertation, UNIKA SOEGIJAPRANATA SEMARANG, 2019.
[28] M. A. Shorif, "A reconfigurable secured wireless sensor networks with xbees and arduino," Ph.D. dissertation, Lethbridge, Alta.: University of Lethbridge, Dept. of Mathematics and . . . , 2014.
[29] A. S. Kabir, "An efficient key distribution scheme in wireless sensor architecture with arduino and xbee," *International Journal of Computer Applications*, vol. 975, p. 8887.

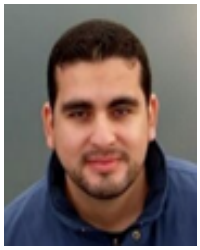
Juan C. Gonzalez-Arango tecnólogo en Telecomunicaciones e Ingeniero de Telecomunicaciones del Instituto Tecnológico Metropolitano, Medellin, Colombia.

Diana C. Ocampo-Munera tecnóloga en Telecomunicaciones e Ingeniera de Telecomunicaciones del Instituto Tecnológico Metropolitano, Medellin, Colombia.

Luis F. Castano-Londono ingeniero Electrónico y MSc en Ingeniería-Automatización Industrial de la Universidad Nacional de Colombia Sede Manizales. Actualmente es candidato a Doctor en Ingeniería - Automática en la misma universidad. Se desempeña como docente del Departamento de Electrónica y Telecomunicaciones de la Facultad de Ingeniería del Instituto Tecnológico Metropolitano e investigador del Grupo Automática, Electrónica y Ciencias Computacionales. Su área de actuación es el diseño digital, particularmente el diseño de sistemas embebidos y sistemas heterogéneos basados en FPGA.



Germán David Góez-Sánchez ingeniero de Telecomunicaciones Instituto Tecnológico Metropolitano ITM, MSc en Automatización y Control Industrial del ITM. Actualmente se desempeña como Docente del Departamento de Electrónica y Telecomunicaciones de la Facultad de Ingeniería e investigador del Grupo Automática y Electrónica y Ciencias Computacionales.



Ricardo A. Velasquez-Velez ingeniero calificado en las áreas de Ingeniería Electrónica y Ciencias de la Computación, con experiencia en investigación a nivel nacional en la Universidad de Antioquia y el Instituto Tecnológico Metropolitano. A nivel internacional ha participado en diferentes grupos de investigación de Alemania, Francia y Suiza. Su trabajo en estos grupos ha sido realizado en las áreas de diseño de sistemas embebidos, modelos de programación paralelos y arquitectura de computadores.