Benchmark para determinar el sistema de cifrado con mejor rendimiento sobre dispositivos inteligentes

Benchmark to determine the best performing encryption system on smart devices

Alber Montoya Benitez[1]
Bayron Ospina[2]

[1] Instituto Tecnológico Metropolitano (Colombia). Correo electrónico: albermontoya@itm.edu.co
orcid: https://orcid.org/0000-0002-7452-8540

[2] Instituto Tecnológico Metropolitano (Colombia). Correo electrónico: bayronospina@itm.edu.co
orcid: https://orcid.org/0000-0002-3993-1430

Recibido: 09-05-2020 Aceptado: 17-09-2020

Cómo citar: Montoya-Benitez, Alber; Ospina, Bayron (2020). Benchmark para determinar el sistema de cifrado con mejor rendimiento sobre dispositivos inteligentes. *Informador Técnico, 84*(2), 175 - 191.
https://doi.org/10.23850/22565035.2782

Resumen

Actualmente las telecomunicaciones y especialmente las comunicaciones móviles han tomado gran importancia y relevancia en las actividades cotidianas de las personas. Sin embargo, el uso de dispositivos móviles se ha visto amenazado por la creciente ola de ataques y *malware* tipo troyanos (e.g., *exploits),* para robo de información y daño de archivos. Con el fin de contrarrestar estos ataques, se han creado técnicas de cifrado de datos y procesos de autenticación. De esta manera se puede evitar violación de la confidencialidad y autenticidad en las comunicaciones. Por otra parte, algunos de los algoritmos de cifrado existentes son inseguros y pueden requerir altos costos computacionales. En este trabajo se realizó el análisis del rendimiento de tres de los principales algoritmos de cifrado definidos por el Instituto Nacional de Estándares y Tecnología (NIST por su sigla en inglés): Rijndael como el Estándar Avanzado de Cifrado (AES por su sigla en inglés), Serpent y Twofish, analizando sus principales características de funcionamiento y realizando pruebas de rendimiento sobre dispositivos inteligentes (smartphones y tablets), con el fin de determinar cuál de estos algoritmos sería el más adecuado para ser implementado en cada equipo. Finalmente, se genera una ecuación llamada costo computacional, que depende de la RAM, CPU y el gasto de batería; con la cual, se pueden realizar análisis para los algoritmos de cifrado simétricos en dispositivos similares a los tratados en este experimento.

Palabras clave: dispositivos inteligentes; seguridad informática; comunicaciones móviles; algoritmos de cifrado; gasto computacional.

Abstract

Currently, telecommunications and especially mobile communications have taken great importance and relevance in people's daily activities. However, the use of mobile devices has been threatened by the growing wave of attacks and Trojans as malware (e.g., exploits) for information theft, and fire damage; To mitigate these attacks there are data encryption techniques and authentication processes. In this way, a breach of confidentiality and authenticity in communications can be avoided. On the other hand, some of the existing encryption algorithms are insecure and may require high computational costs. This work analyzes the performance of three of the main encryption algorithms defined by the National Institute of Standards and Technology (NIST): Rijndael as Advanced Encryption Standard (AES), Serpent and Twofish, specifying their main operating characteristics, analyzing performance tests on smart devices, to determine which of these algorithms is the

most appropriate to be implemented in each device. Finally, an equation called computational cost is generated, which is a function of RAM, CPU, and battery drain; that analyses for symmetries encryption algorithms that can be performed on similar devices to those treated in this experiment.

Keywords: smart devices; computer security; mobile communications; encryption algorithms; computational cost.

1. Introducción

Hoy en día se requiere de dispositivos móviles no solo en las comunicacionessino en gran parte de actividades cotidianas. No obstante, en tendencias como BYOD (traiga su propio dispositivo) su uso se ve amenazado, debido a la presencia de un alto número de ataques. Las técnicas de autenticación y cifrado de datos son protecciones contra tales amenazas, ayudando a que las comunicaciones y el almacenamiento sean seguros y legítimos. En contraste, los algoritmos de cifrado demandan costos computacionales altos y afectan en gran medida el rendimiento de los equipos. Por otro lado, la miniaturización ha reducido el tamaño de los componentes internos de estos dispositivos mediante nuevos desarrollos, la nanotecnología y materiales más resistentes. La reducción del tamaño de los elementos hace que las baterías almacenen menos energía. A largo del tiempo se han realizado algunos estudios sobre el tema, los cuales, son base de este trabajo. En este trabajose presenta un "benchmark" para identificar cuál sería el mejor algoritmo de cifrado, en términos de costo computacional y seguridad, para dos tipos de equipos móviles. Para el experimento, se realizaron diversas pruebas con diferentes tamaños y tipos de archivos, obteniendo valores medios que posteriormente se analizaron y compararon entre sí, con el fin de proponer las mejores opciones y así usarlas en cada dispositivo. El trabajo involucra las variables CPU, RAM y consumo de batería cuando se ejecutan tres de los algoritmos de cifrado más seguros definidos por el Instituto Nacional de Estándares y Tecnología (NIST por su sigla en inglés) (Federal Information Processing Standards Publications [FIPS PUBS] 2001).

Infortunadamente, hay cada vez más ataques a las comunicaciones móviles, en el primer semestre de 2019, más de 440.000 usuarios únicos fueron atacados por amenazas financieras, un 7 % en comparación del año anterior, también el número de ataques financieros móviles en la primera mitad de 2019 fue de 3.740.378, un 107 % más que en el primer semestre del año anterior; actualmente solo un pequeño porcentaje de teléfonos inteligentes y tabletas ha instalado algún software de seguridad *(*Kupreev; Sidorina; Chebyshev; Kuskov, 2019*)*. Además el uso de los canales electrónicos ha aumentado para el sistema financiero, pero incluso un alto porcentaje de usuarios todavía no utiliza servicios en línea por considerarlos inseguros.

Uno de los principales inconvenientes en las telecomunicaciones es la implementación de un sistema de autenticación no cifrado, o en algunos casos, el uso de un cifrado débil que puede ser fácil de atacar, los cuales son ineficientes para proteger información crítica. Por ejemplo, la aplicación WhatsApp utilizaba el sistema crypt5 o crypt7 para proteger las conversaciones, o autenticación WEP en red inalámbrica, la cual, es muy frágil a ataques. Además, el costo computacional requerido por algunos algoritmos afecta el tiempo de funcionamiento de la batería. En la literatura revisada no se encontró ningún método para determinar cuál es el mejor algoritmo para cada dispositivo móvil en función de las características de CPU y memoria mencionadas anteriormente. En este documento se muestra un punto de referencia para los algoritmos de cifrado y se muestra el rendimiento de cada uno en un dispositivo móvil.

2. Contexto

2.1. Amenazas

Existen diversos tipos de ataques a los algoritmos mencionados anteriormente. Estos incluyen el Time-Memory Trade-Off (TMTO), un ataque que reduce el tiempo de criptoanálisis mediante el uso de datos almacenados en

la memoria. Para contrarrestar este ataque, Saberi; Shojaie; Salleh; Niknafskermani; Alavi (2012) agregan un bloque de cifrado en el intercambio de autenticación de mensajes *Advanced Encryption Standard (AES).* Otro tipo de ataque que afecta a estos sistemas de autenticación son los ataques de diccionario, ampliamente utilizados contra los sistemas de autenticación *Wi-Fi Protected Access (WPA*) (Shivkumar; Umamaheswari, 2011). Como consecuencia de los desarrollos de WPA, WPA2 y WPA3, la efectividad de este ataque se minimiza porque se realiza un cambio dinámico de la clave de autenticación. El *Data Encryption Standard (DES)* en sí era vulnerable a este ataque debido a su clave corta de 56 bits, lo que obligó a la aparición de 3DES y AES con longitudes de bits de hasta 128 y 256 bits.

En ataques comunes en SMS de móviles, los troyanos se usan generalmente para enviar mensajes *premium* desde dispositivos, sin embargo, su uso ha sido reemplazado por los de puerta trasera, programas desarrollados para facilitar las malas intenciones de los ciberdelincuentes que abren puertos adicionales. Con una puerta trasera es fácil descargar *malware* en su teléfono o tablet y facilita el robo de información personal que puede incluir: imágenes, videos, números de teléfono, direcciones de correo electrónico y coordenadas de ubicación (Tecnology Day revista IT Now, 2013).

2.2. Protección

De acuerdo con la Norma de Seguridad ISO/IEC 27001, aprobada y publicada en octubre de 2005 por la Organización Internacional de Normalización (ISO) y la Comisión Electrotécnica (IEC) para proporcionar una fuerte seguridad a las comunicaciones, se deben cumplir tres características,disponibilidad, confidencialidad e integridad. Para cumplir con estos parámetros, es necesario asegurar la autenticación como un elemento importante en el proceso de seguridad de tres maneras: biometría, tarjetas inteligentes o tokens y llaves. Las claves se pueden transferir utilizando algoritmos cifrados o no cifrados. Los algoritmos de cifrado se dividen en dos grupos simétricos y asimétricos. A continuación, se explican brevemente los algoritmos clave de este estudio.

2.2.1. AES - Rijndael

Trabaja a nivel de bytes interpretando bytes como un cuerpo de Galois GF (2^8) (FIPS PUBS, 2001), cubre los registros a nivel de 32 bits al considerarlos como polinomios de grado inferior a 4 con coeficientes que son polinomios en GF (2^8) (FIPS PUBS, 2001). En los cálculos, los polinomios irreducibles de grado m se usan de la siguiente manera (Ecuación 1):

$$GF\,(pm) = \{\lambda 0 + \lambda 1X + \lambda 2X2 + \cdots \lambda m - 1Xm - 1;\ \lambda i \in Zp \qquad (1)$$

El tamaño del bloque y la longitud de la clave pueden ser variables en múltiplos de 4 bytes. Las claves predeterminadas son 128, 192 y 256 bits. La estructura consta de dos etapas y, estas a su vez, por una serie de "rondas" que permiten ciertos cambios en la matriz de estado. Se explicarán brevemente en la Figura 1, Figura 2, Figura 3, Figura 4.

SubBytes: cada byte en la matriz de estado se reemplaza con su entrada en una tabla fija de búsqueda de 8 bits (Ecuación 2).

$$S = bij = (aij) \qquad (2)$$

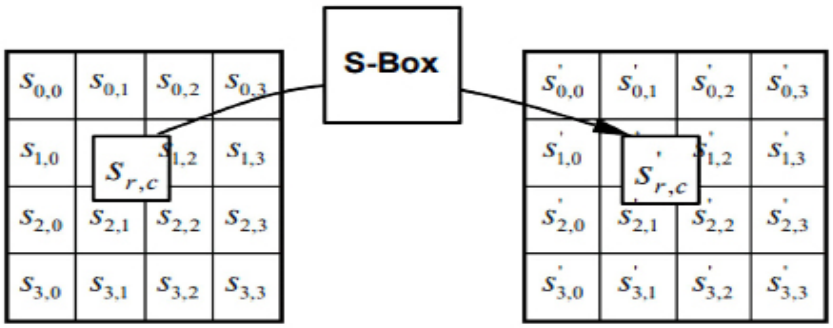

Figura 1. Reemplazo de entradas de matriz
Fuente: FIPS PUBS (2001).

ShiftRows: los bytes en cada fila se desplazan cíclicamente a la izquierda.

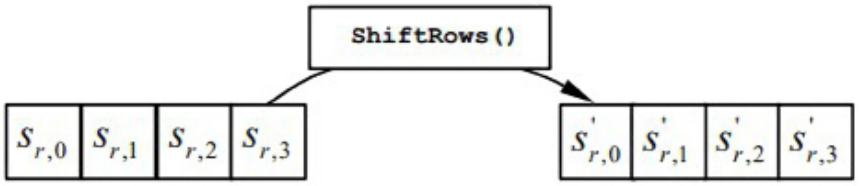

Figura 2. Rotación de elementos
Fuente: FIPS PUBS (2001).

MixColumns: cada columna se multiplica por un polinomio constante *c (x)*.

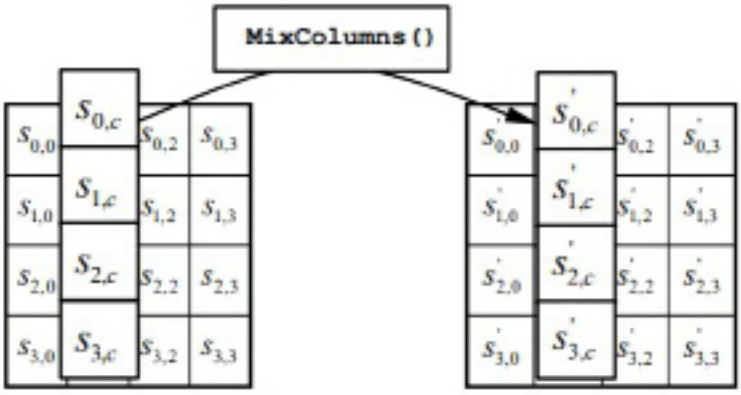

Figura 3. Transformación lineal
Fuente: FIPS PUBS (2001).

AddRoundKey: cada byte se combina con uno de la subclave utilizando la operación XOR(⊕).

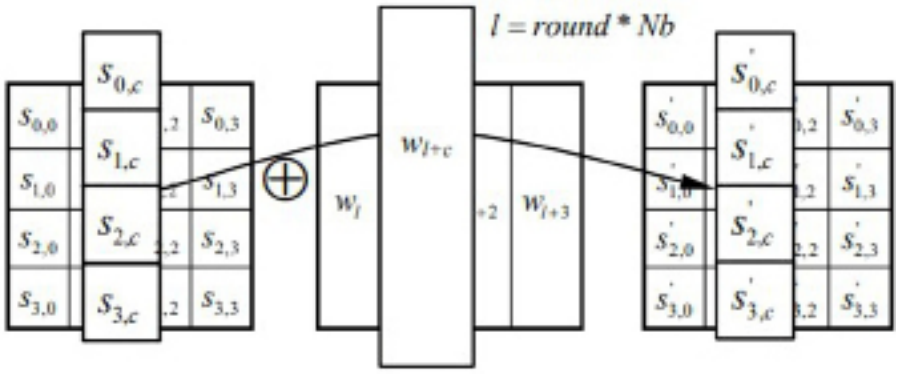

Figura 4. Combinación de Subclave usando XOR
Fuente: FIPS PUBS (2001).

2.2.2. RSA (Rivest, Shamir & Adleman)

A diferencia del algoritmo anterior, este es un algoritmo asimétrico que se aplica tanto para cifrar como para firmar digitalmente (RSA Laboratories, 2002). Consta de tres pasos: generación de claves, cifrado y decodificación.

Generación de claves

Cada usuario elige dos números primos diferentes *p y q. (n)* se calcula con Ecuación 3.

$$n = pq \qquad (3)$$

Y se usa como módulo para las llaves. $\varphi(n)$ se calcula con donde φ es la función de Euler φ (Ecuación 4).

$$\varphi(n) = (p-1)(q-1) \qquad (4)$$

Se elige un número entero positivo *e* menor y primo relativo para (*n*). *e* se muestra como el exponente de la clave pública *d,* así *d* se mantiene como el exponente de clave privada (Ecuación 5).

$$d = e\text{-}1(n) \qquad (5)$$

Cifrado

El usuario A comunica su clave pública (*n, e*) al usuario B y A mantiene en secreto la clave privada. Ahora B quiere enviar un mensaje M a A. Primero, B convierte M en un número entero m menor que n mediante el uso de un protocolo reversible; luego calcula el texto cifrado *c* por la Ecuación 6.

$$C \equiv (mod\ n) \qquad (6)$$

Decodificación

A puede recuperar *m* de *c* utilizando su exponente d de la clave privada (Ecuación 7).

$$m \equiv (mod\ n)$$

2.2.3. Serpent

Fue diseñado por Anderson, Biham y Knudsen (1998). En este algoritmo utilizaron un tamaño de bloque de 128 bits que admite tamaños de clave de 128, 192 y 256 bits de longitud. El cifrado implica 32 rondas de operación de sustitución-permutación en cuatro bloques de 32 bits. Cada ronda usa 32 copias de las mismas S-Box de 4 bits. Serpent fue diseñado para operaciones que se ejecuten en paralelo utilizando 32 desplazamientos de 1 bit (Figura 5).

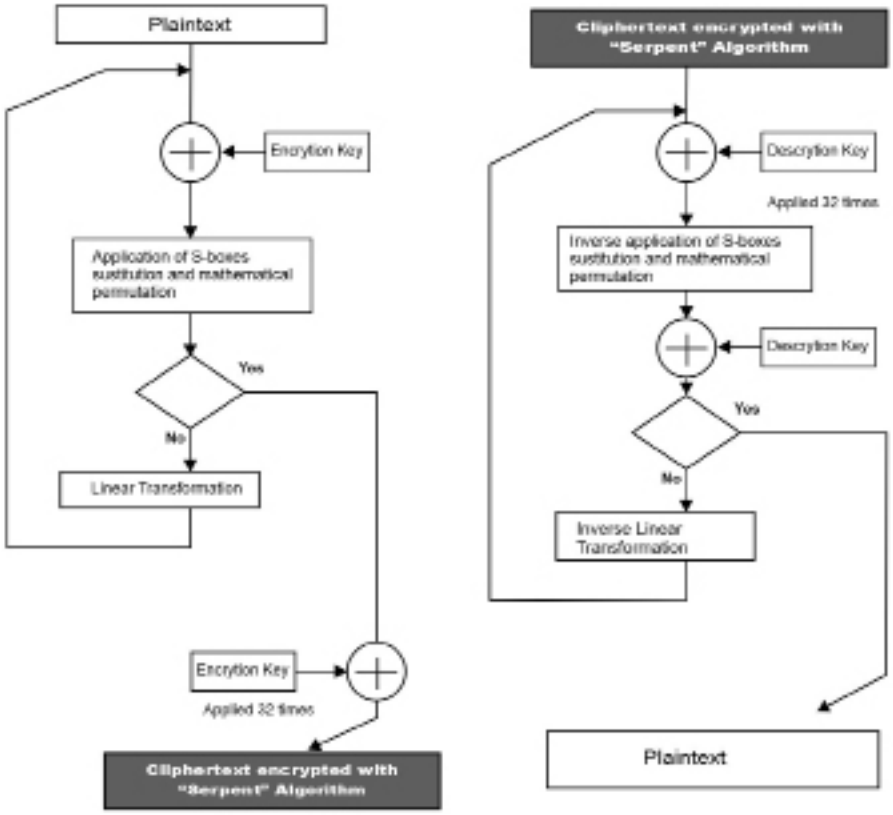

Figura 5. Cifrado Serpent
Fuente: Anderson; Biham; Knudsen (1998).

2.2.4. Twofish

Fue diseñado por Schneier, Kelsey, Withing, Wagner, Hall y Ferguson (1998). Es un método de cifrado simétrico en bloque desarrollado por Counterpane Labs y presentado al concurso NIST en busca de un reemplazo para DES. Su tamaño de bloque es de 128 bits y el tamaño de la clave puede ser de hasta 256 bits. Twofish obtiene otros elementos de diseño, como el cifrado SAFER de la familia de transformadas Pseudo-Hadamard (PHT). Utiliza la misma estructura Feistel de DES (Figura 6). En la mayoría de las plataformas de software, Twofish es un poco más lento que Rijndael para las claves de 128 bits, pero algo más rápido para las claves de 256 bits (Schneier *et al.*, 1998).

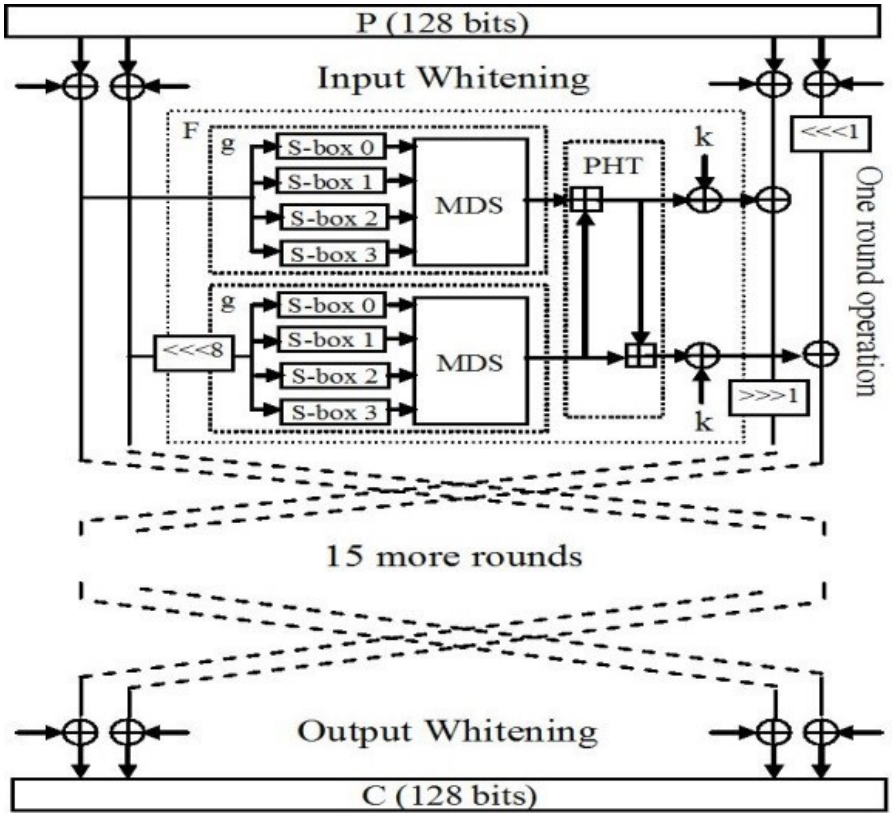

Figura 6. Cifrado Twofish. Tomado de Schneier *et al.* (1998)

2.3. Investigaciones previas

Uno de los estudios publicados por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE por sus siglas en inglés) propone un esquema de seguridad criptográfica que reduce el uso de los recursos utilizados por los dispositivos móviles antes de cargar datos en la nube. El esquema es la modificación de operaciones lógicas y matemáticas en bloques cifrados (Khan; Kiah; Khan; Madani; Khan, 2013), esta propuesta puede reducir el desperdicio de recursos, pero podría comprometer la fortaleza de los algoritmos criptográficos. En otro estudio, se expone un método de autenticación en el que la posición del usuario y la hora del día forman parte del proceso. En este caso, se permitirá la autenticación del usuario solo a ciertas horas del día. El método se basa en el algoritmo hash MD5 y el cifrado DES, pero también es compatible con AES y 3DES (Karimi; Kalantari, 2011). Esta propuesta es bastante adecuada porque aumenta el control y reduce el tiempo de exposición a los ataques.

En una mirada a las tendencias actuales en Colombia y el mundo, se presentan múltiples desarrollos y aplicaciones que se pueden clasificar en el Internet de las cosas (IoT, por sus siglas en inglés). Este es el caso de Dussán, Vanegas, Chavarro y Molina (2016), quienes presentan un prototipo electrónico que monitorea parámetros fisicoquímicos a fin de detectar situaciones críticas de un cultivo de peces. El prototipo desarrollado requiere de equipos con bajos recursos de máquina, tipo Arduino o el microprocesador ATMEGA 328P, que podrían requerir e implementar sistemas de cifrado para la protección de la información.

En un estudio sobre tecnologías de VoIP, se realiza un análisis de amenazas que combinan protocolos de intercambio de claves basado en la curva elíptica Diffie Hellman (ECDH) criptográfica de clave pública con la autenticación basada en la identidad del usuario, que también utilizan la información (identidad, dirección IP, puerto). Se propone el protocolo de intercambio de claves para garantizar confidencialidad e integridad. Bandung y Priyatna (2017) realizaron un análisis de seguridad entre el protocolo propuesto con el protocolo

ECDH existente y compararon su rendimiento de generación de claves y el tiempo de intercambio de claves. Los resultados mostraron que la combinación de ECDH y un mecanismo de autenticación ha demostrado ser segura contra ataques. Con la adición de la autenticación, el tiempo total de ejecución de la clave de generación y la clave de intercambio es más lento en un 11,70 % que el ECDH original. Sin embargo, se puede garantizar que las comunicaciones VoIP todavía se pueden realizar de forma interactiva. Por su parte, Bian, Lu y Kuang (2013) proponen una modificación al algoritmo Blowfish para que se ejecute en un teléfono inteligente. Este sistema mejora la eficiencia del algoritmo para trabajar en teléfonos móviles, aunque esto no es tan seguro como el objetivo de los algoritmos de este trabajo que fueron finalistas en el NIST.

Ren, Boukerche y Nelem (2010) desarrollaron un sistema de cifrado híbrido con claves simétricas y asimétricas, para la autenticación en redes ah-hoc, lo que demuestra ser bastante seguro, aunque podría ser algo complejo y pesado para ciertos dispositivos móviles. De manera similar, Zhang, Xiao, Zhang (2010) desarrollaron un sistema criptográfico basado en el algoritmo IDEA, minimizando el costo computacional cuando se ejecuta en teléfonos móviles. Esto podría ser un desarrollo interesante y es una inspiración para el trabajo actual.

Al revisar investigaciones en campos de la salud, se encontraron investigaciones como la de Tovar; Díaz, Quiñones, Pabón y García (2018), donde se desarrolla un sistema de rehabilitación física teleoperado para pacientes que viven en áreas rurales. Este sistema maneja un conjunto de indicadores críticos que requieren una baja latencia para minimizar el margen de error en la respuesta a los procesos que se manejan en tiempo real. Por tal razón, es importante que se implementen programas de bajo consumo de recursos, pero que a su vez garanticen la confidencialidad de la información. De manera similar en la revisión de Castro *et al.* (2017), se presentan más desarrollos en el campo de la salud con IoT. Por su parte, en la investigación de Qi, Pan y Ding (2011) se utiliza un *Field Programmable Gate Array (FPGA)*, que aumenta la velocidad de cifrado en mensajes cortos en dispositivos móviles, resolviendo el problema anterior. Esta solución puede ser un ejemplo para implementar en algoritmos simétricos.

En cuanto a Alomari y Samsudin (2011) implementaron un XTS-AES en una GPU, usando un procesamiento en paralelo, lograron un cifrado más eficiente y con un impacto menor en los dispositivos. Uskov (2012) presenta un punto de referencia de algoritmos de autenticación y cifrado para MVPN con diferentes resultados con respecto a los algoritmos que conciernen a este trabajo, se demostró que AES tiene el mejor rendimiento en 2 de 3 plataformas probadas. Finalmente, Umaparvathi y Varughese (2010) realizaron un análisis de tres algoritmos: AES, 3DES y Blowfish en MANET, encontrando que AES es el mejor. Sin embargo, se sabe que 3DES es un algoritmo en desuso y Blowfish es menos seguro que el primero.

3. Metodología

3.1. Método aplicado

Las pruebas consisten en medir el rendimiento de diferentes algoritmos cuando se ejecutan en ciertos dispositivos móviles, específicamente se correrán AES, Twofish y Serpent; en un teléfono inteligente y una tableta. Se utilizaron aplicaciones diferentes, obteniendo resultados equivalentes. Con base en el funcionamiento interno de cada aplicación, se propuso la siguiente secuencia de pasos, donde se consideran dos variables de respuesta, CPU y memoria RAM. El experimento se realizó utilizando una técnica denominada diseño de bloques aleatorizados, la cual consiste en los siguientes pasos:

1. Instalación y configuración de software de cifrado (herramienta 1). En este paso se realizaron configuraciones iniciales como el tamaño de la clave en cada aplicación.

2. Instalación y configuración de software de medición (herramienta 2). Esta herramienta toma los valores de las variables CPU y RAM que muestran resultados numéricos y gráficos en una línea de tiempo.

3. Cifrar/descifrar múltiples archivos de diferentes tamaños. Las pruebas se realizaron con diferentes tipos de archivos, los formatos utilizados fueron docx, xlsx, jpg, mp3 y avi, que son los tipos de archivos más comunes para el almacenamiento de datos en el dispositivo analizado.

4. Para ejecutar los algoritmos, se realizó solo código nativo (sin leer ni escribir en el disco). Como prueba alternativa, los algoritmos se ejecutaron en un proceso que aísla la lectura y escritura en el disco, y las aplicaciones tienen la función de obtener explícitamente el consumo de recursos (CPU y RAM) del proceso de cifrado.

5. Hacer mediciones de las variables de CPU y memoria para determinar el gasto de batería. Los valores de las variables de CPU y RAM se tomaron cuando se ejecutaron los algoritmos de cifrado, en el proceso de medición se aislaron otros factores externos, tomando como referencia un punto inicial específico. Finalmente, los resultados se tabularon obteniendo un promedio.

6. Proponer los algoritmos más adecuados para cada tipo de dispositivo móvil y aplicación analizados. Consiste en plantear una expresión matemática para determinar el costo computacional de los algoritmos analizados sobre cada dispositivo.

4. Resultados

4.1. Resultados obtenidos

Rijndael, Twofish y Serpent fueron los algoritmos seleccionados para esta investigación, de acuerdo con que fueron los finalistas en la competencia AES (FIPS PUBS, 2001). Por lo tanto, son altamente utilizados en los sistemas de cifrado. Las pruebas se realizaron en un teléfono inteligente y una tableta cuyas características de hardware y software se muestran en la Tabla 1.

Tabla 1.
Características de dispositivos

SMARTPHONE	TABLET
Samsung Galaxy S10+.	Dell Venue 11 Pro11i-6363blk laptop/tablet
Procesador Qualcomm Snap-dragon Octa-core 2,4 GHz. 128 GB Flash.	Procesador Intel Core i3-4020Y 1,5GHz. Tarjeta gráfica integrada Intel GT2. 128GB Disco de almacenamiento de estado sólido.
Memoria RAM 8 GB.	Ram 4GB DDR3.
Sistema operativo Android 9,0.	Sistema operativo Win 10.
Batería lithium-ion, 4.100 mAh.	Batería 37Wh ion litio.

Fuente: Elaboración propia.

Inicialmente, se instaló la aplicación *Secret Space Encryptor* en el teléfono inteligente, se llevaron a cabo dos tipos de pruebas, un *benchmark* para el código nativo de los 3 algoritmos (Figura 7) y el cifrado de un archivo de 300MB, que midió el rendimiento de la CPU y el uso de RAM (Figura 8, Figura 9). Estos resultados se analizarán gráficamente en la siguiente sección.

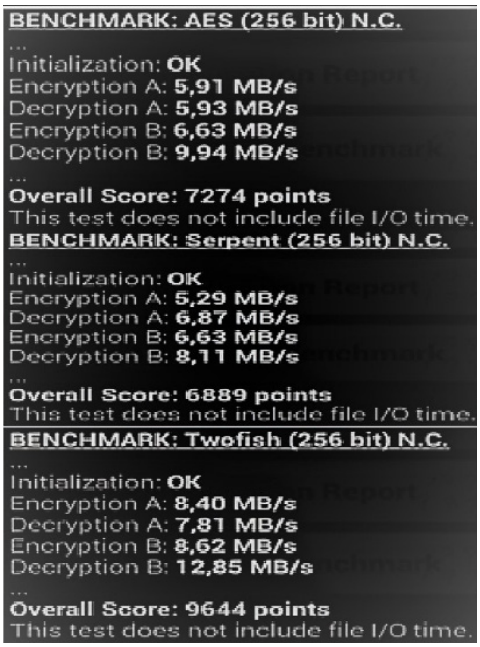

Figura 7. Benchmark Smartphone
Fuente: Elaboración propia.

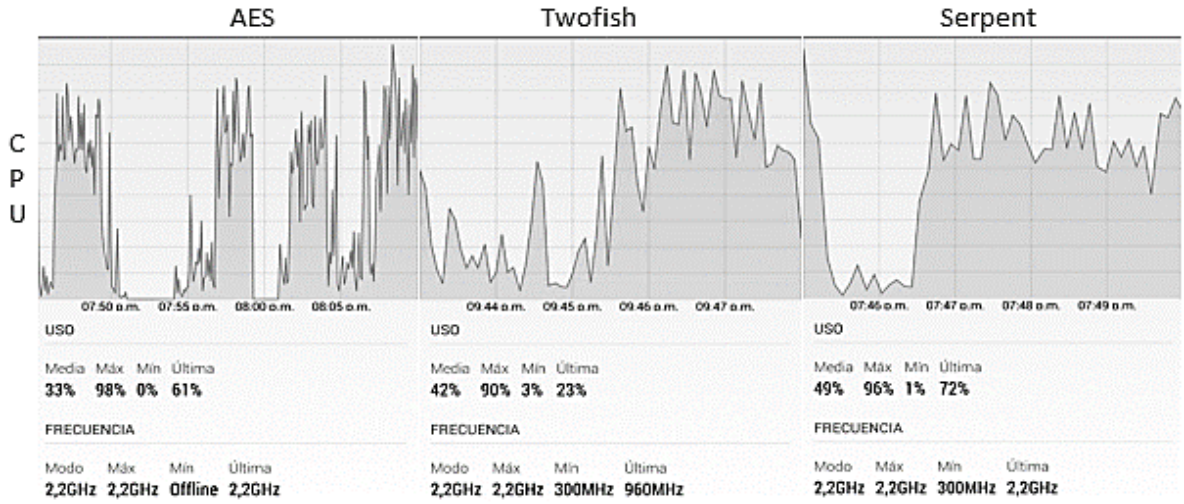

Figura 8. Rendimiento de CPU en Smartphone
Fuente: elaboración propia.

Figura 9. Uso de RAM en Smartphone
Fuente: elaboración propia.

Posteriormente, en la tablet se instaló la aplicación *TrueCrypt* en la que se ejecutaron pruebas de cifrado/descifrado, utilizando la misma longitud de clave, método y tipo de algoritmo utilizado en el teléfono inteligente, así como las posibles combinaciones entre pares, para un tamaño de búfer de 1GB que aísla el proceso de escritura en disco, ya que funciona en una unidad RAM. Los resultados obtenidos por el *software*, muestran en primer lugar de rendimiento (proceso cifrado/descifrado) al algoritmo AES. Twofish fue el segundo, mientras que Serpent fue el último en rendimiento. Los valores estadísticos de las medias aritméticas de la capacidad de procesamiento fueron respectivamente 518MB/s 80MB/s y 47.1MB/s (Figura10).

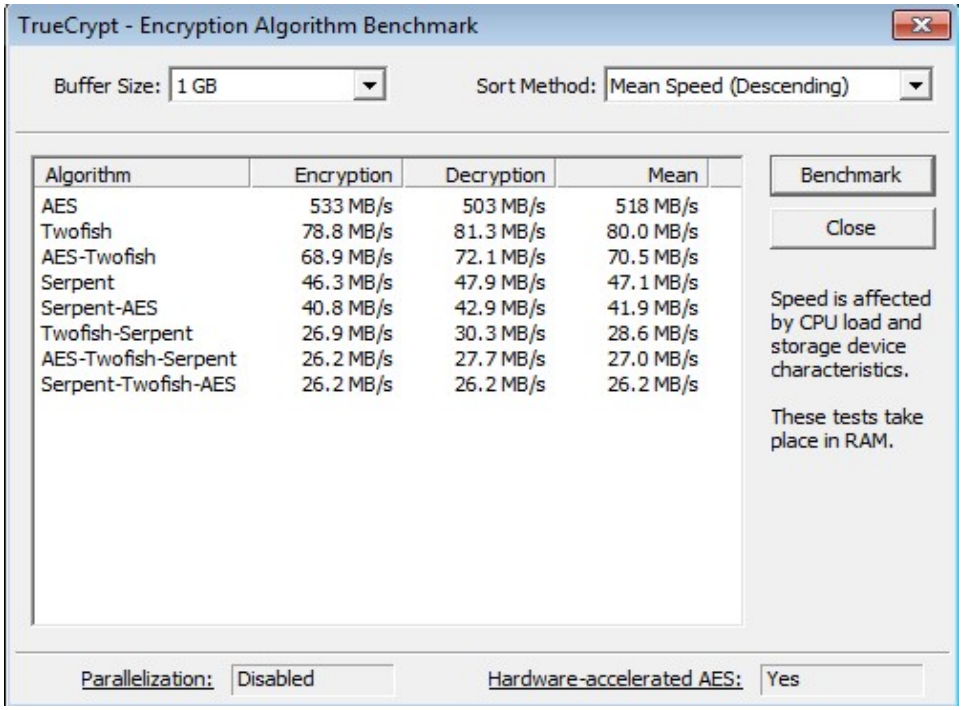

Figura 10. Benchmark en tablet [Mbps]
Fuente: elaboración propia.

En el segundo tipo de prueba, en la tablet, el rendimiento del procesador, el uso de la memoria y la escritura se miden y se muestran para cada algoritmo en el dominio del tiempo (Figura 11). Al ejecutar AES, el porcentaje de utilización del procesador es menor que al ejecutar Twofish y Serpent, aunque el consumo de memoria es menor cuando se ejecuta Twofish.

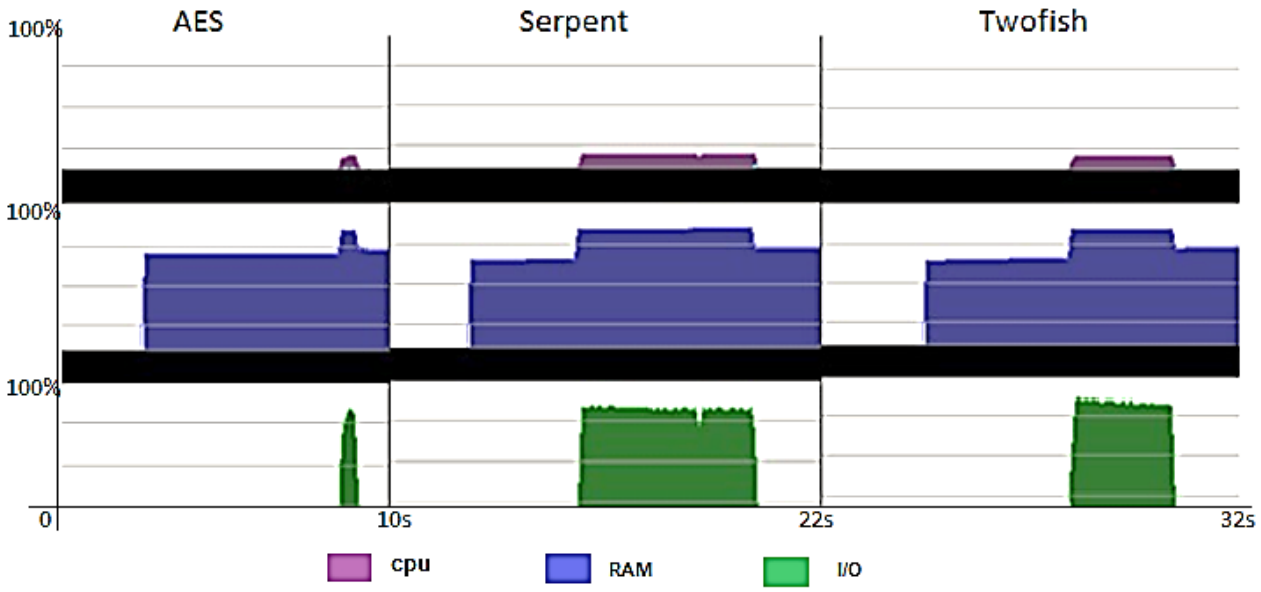

Figura 11. Rendimiento CPU-MEMORIA-I/O
Fuente: elaboración propia.

La velocidad de cada algoritmo para cifrar la información se obtuvo utilizando las mismas pruebas para cada uno.

4.2. Análisis de resultados

En la primera prueba, el *benchmark* en el teléfono inteligente muestra el algoritmo Twofish como el de mejor rendimiento en el consumo de recursos de la máquina, seguido por Rijndael y finalmente Serpent. La velocidad de procesamiento para cada uno fue de 942MB/s, 710MB/s y 673MB/s respectivamente. Entre tanto, en la prueba con la tablet, Rijndael fue el de mejor rendimiento con una amplia ventaja sobre los otros algoritmos (Figura 10). Este comportamiento corrobora la teoría que sugiere que el algoritmo Serpent es más robusto que AES, ya que funciona con más rondas en el proceso de cifrado/descifrado (32 sobre 10) (Anderson; Biham; Knudsen, 1998). Esta característica mejora la fuerza criptográfica del algoritmo, pero lo hace más lento y pesado para ejecutarse sobre dispositivos móviles. En cuanto a Twofish, tiene una estructura diferente (Fiestel), que no lo hace tan fuerte como AES y Serpent, pero se transforma en un algoritmo más responsivo, por tal motivo ejecuta más rápidamente el proceso de cifrado que el algoritmo Serpent.

Tabla 2.
Resumen de Resultados

	ALG/ VBLE	BENCHMARK [MB/s]	SPEED [MB/s]	CPU [%]	MEMORY [%]
SMARTPHONE	RIJNDAEL	710	433	33	39
	SERPENT	673	78	49	35
	TWOFISH	942	96	42	33
TABLET	RIJNDA EL	518	341	46	36
	SERPENT	47.1	42.7	50	47
	TWOFISH	80	70.6	54	39

Fuente: Elaboración propia.

En la segunda prueba con Smartphone, se cifró un archivo de 300 MB. Rijndael se desempeña mejor en términos de procesamiento, ejecutándose con casi un 10 % de utilización de CPU menos que Twofish y un 16 % menos que el algoritmo Serpent. En cuanto al consumo de memoria, el Twofish obtuvo menos consumo, un 33 %, Serpent 35 %, mientras que AES se vuelve más pesado con 39 %. En el caso de la prueba de tablet, Twofish no obtuvo tan buen resultado, un 54 % de uso de CPU, solo un poco más alto que Serpent con 50 % y Rijndael con 46 %. La prueba de la variable de memoria mostró que AES consumió menos memoria que Twofish y Serpent. En la Tabla 2 se presenta un resumen de todos los resultados y promedios.

Al hacer un análisis de estos resultados, se puede inferir que los mismos presentan relación con las características propias de cada equipo, es decir, que el poder de computación de estos dependen principalmente del funcionamiento de la CPU y de la RAM conjuntamente. No obstante, existen más elementos tanto de *hardware* como de *software* que influyen en alguna medida en los procesos internos que ejecutan estos dispositivos, por ejemplo, los procesos de cifrado. Tales características de los equipos son distintas, ya que estos son fabricados en general con ciertos fines específicos y, de esta manera, poder tomar las mejores decisiones a la hora de elegir cual equipo utilizar, según el propósito para el cual se adquieren. De igual forma, de acuerdo con el dispositivo que utilice hacer la mejor selección de los sistemas de cifrado para la protección de información, logrando los mejores resultados de rendimiento.

En conclusión, según la teoría, los tres algoritmos son muy seguros, pero según los valores obtenidos en las pruebas, el algoritmo AES se confirma como el de mejor rendimiento (tasa de cifrado [MB/s]), bajo uso de CPU y consumo de memoria para trabajar en tabletas, mientras que Twofish puede ser una opción adecuada para implementar en teléfonos inteligentes.

4.3. Función de consumo de recursos

El consumo de batería está directamente relacionado con el uso del procesador y la memoria. En función de estas variables, se propuso un indicador llamado *costo computacional* (mediante el cual se puede determinar cuál algoritmo de cifrado usa mayor cantidad recursos de máquina en cada dispositivo móvil. En la Ecuación 8, se describe el indicador de costo.

$$I_C = \tau * (0.6\alpha + 0.4\mu) + \frac{\omega\varphi}{\beta}, \quad \varphi > 0, \ \beta > 0 \qquad (8)$$

Donde:

I_C: *Indicador de Costo del Algoritmo*

τ: *Promedio tiempo de ejecución*

α: *Promedio de la variable CPU*

μ: *Promedio de la variable RAM*

ω: *Número de rondas del algoritmo*

φ: *Longitud de clave*

β: *Tamaño del bloque a cifrar*

Estas variables dependen, de una constante que dará un peso a cada una de ellas en el costo computacional. Las ponderaciones se definieron en 0,6 para la CPU y 0,4 para la RAM, de acuerdo con un análisis de varianza extraído del experimento.

Aplicando esta fórmula a los valores obtenidos del experimento, se obtuvieron los siguientes costos para cada algoritmo al ejecutarse en ambos dispositivos. El indicador es un valor sin unidades (adimensional), su función consiste en determinar el costo computacional (Figura 12, Figura 13).

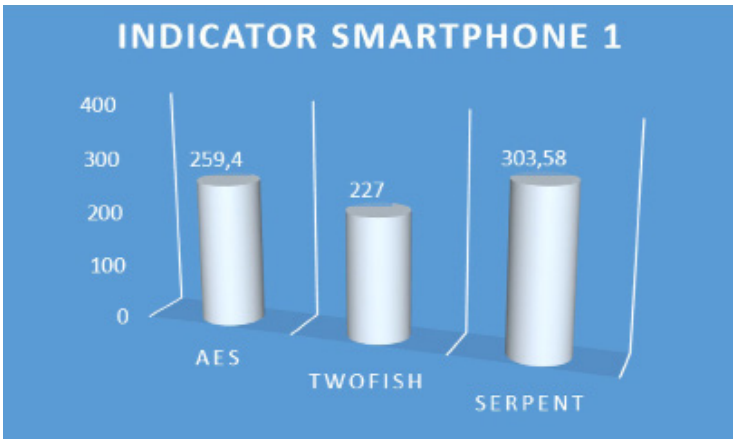

Figura 12. Costo computacional del Smartphone
Fuente: elaboración propia.

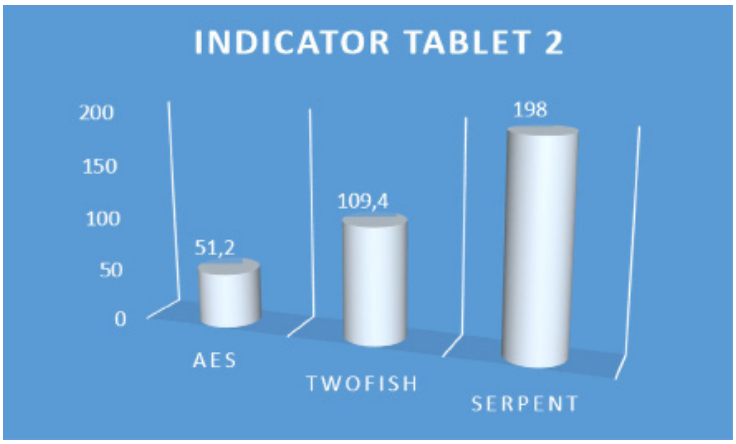

Figura 13. Costo computacional de la tablet
Fuente: elaboración propia.

En la Figura 12, se muestra que en el teléfono inteligente el algoritmo Twofish tiene un costo computacional menor a los demás, mientras que en la Figura 13 se ven los resultados para la Tablet, donde AES tiene el costo computacional más bajo y Serpent tiene el costo computacional más alto, con relación a los resultados obtenidos.

5. Discusión

Teniendo en cuenta que en la actualidad existen diversos métodos de cifrado para proteger la información que se maneja en los diferentes dispositivos, es importante que se pueda implementar el mejor algoritmo en cuanto a rendimiento y gasto energético en cada caso. Esto cobra mayor relevancia en las nuevas tendencias, como lo es el internet de las cosas, donde se manejan equipos/actuadores con bajos recursos de máquina y que, por esta razón en muchas de sus implementaciones no utilizan métodos de cifrado como medida de seguridad en sus sistemas. Aunque los equipos analizados en este trabajo son dispositivos tipo *smartphone* y *tablet*, los resultados obtenidos en las pruebas y, particularmente la ecuación de indicador de costo computacional, pueden abrir un camino para analizar de manera similar el comportamiento de los tres algoritmos trabajados en dispositivos IoT.

6. Conclusiones

Con las pruebas ejecutadas fue posible realizar un "*benchmark*" para los algoritmos AES-Rijndael, Twofish y Serpent. A partir de los resultados obtenidos, se puede concluir que AES-Rijndael es el algoritmo que requiere menos recursos de máquina cuando se lleva a cabo el proceso de cifrado/descifrado, mientras que el algoritmo Serpent utiliza más recursos que los otros dos. Sin embargo, el tiempo requerido para la ejecución del proceso en el teléfono inteligente es mayor cuando se usa AES-Rijndael.

Al incluir la variable de tiempo en los cálculos, se encontró que AES- Rijndael es el mejor algoritmo para ejecutarse en la *tablet* con un 50 % de los recursos que consumiría Twofish, esto es una relación de 2 a 1 en rendimiento. Por su parte, el algoritmo Twofish funciona mejor en el teléfono inteligente con un 87 % de lo que consumiría Serpent Este último resultado es importante, debido a que Twofish se puede presentar como una alternativa muy confiable y segura para este tipo de teléfonos inteligentes y podría ser un fuerte contendor para el AES-Rijndael, que actualmente es el más utilizado.

Por su parte, la estructura y la cantidad de rondas que ejecuta el algoritmo Serpent, lo hacen seguro y fuerte. No obstante, por esta razón, se evidencia que este algoritmo presenta procesamiento lento en los equipos móviles evaluados, por lo que no sería adecuado su uso en dispositivos inteligentes. Adicionalmente, al requerir más recursos de máquina, aumenta el consumo de batería reduciendo la vida útil de la misma.

Por último, se desarrolló una ecuación para estimar el costo computacional. Este indicador permite determinar la cantidad de recursos de máquina que consume cada algoritmo al ejecutarse en un dispositivo móvil. La ecuación se elaboró en función del tiempo, CPU, RAM, ciclos del algoritmo, tamaños de la clave y bloques a cifrar. Esta ecuación permite la elección adecuada de los algoritmos de cifrado que a la vez es esencial para el uso efectivo de la criptografía en la seguridad.

Referencias

Alomari, Mohammad; Samsudin, Khairulmizam (2011). A framework for GPU-accelerated AES-XTS encryption in mobile devices. *TENCON 2011 - 2011 IEEE Region 10 Conference* (pp.144-148). Bali, Indonesia. 10.1109/TENCON.2011.6129080

Anderson, Ross; Biham, Eli; Knudsen, Lars (1998). Serpent: A New Block Cipher Proposal. In *International Conference on Fast Software Encryption* (pp 222–238). Springer, LNCS.

Bandung, Yoanes; Priyatna, Andri (2017). Development of key exchange protocol to enhance security of voice over internet protocol on mobile phone. *International Journal on Electrical Engineering and Informatics,* 9(1), 173-184. http://dx.doi.org.itm.elogim.com/10.15676/ijeei.2017.9.1.12

Bian, Jiali; Lu, Bei; Kuang, Jian (2012). A new hierarchical file encryption system based on smartphone, Computer Science and Network Technology (ICCSNT). In *Proceedings of 2012 2nd International Conference on Computer Science and Network Technology* (pp. 943-946). Changchun, China. 10.1109/ICCSNT.2012.6526082

Castro, Diego; Coral, William; Cabra, José; Colorado, Julián; Méndez, Diego; Trujillo, Luis (2017). Survey on IoT solutions applied to Healthcare. *DYNA*, 84(203), 192-200. http://dx.doi.org/10.15446/dyna.v84n203.64558

Dussán, Sergio; Vanegas, Oscar; Chavarro, Adrián; Molina, Johan (2016). Diseño e implementación de un prototipo electrónico para monitoreo de parámetros físico-químicos en cultivo de tilapia a través de una aplicación móvil. *Informador Técnico*, 80(1), 49-60. https://doi.org/10.23850/22565035.322

FIPS PUBS (2001). *Announcing the ADVANCED ENCRYPTION STANDARD (AES).* Recuperado de: https://www.cisco.com/c/dam/en/us/products/collateral/security/anyconnect-secure-mobility-client/fips.pdf

Karimi, Rohollah; Kalantari, Mohammad (2011). Enhancing security and confidentiality on mobile devices by location-based data encryption, Networks (ICON), *2011 17th IEEE International Conference on Networks* (pp. 241-245). Singapore, Singapore. 10.1109/ICON.2011.6168482

Khan, Abdul; Kiah, M.; Khan, Samee; Madani, Sajjad; Khan, Atta (2013). A study of incremental cryptography for security schemes in mobile cloud computing environments. In *013 IEEE Symposium on Wireless Technology & Applications (ISWTA)* (pp. 62-67). Kuching, Malaysia. 10.1109/ISWTA.2013.6688818

Kupreev, Oleg; Sidorina, Tatyana; Chebyshev, Victor; Kuskov, Vladimir (2019). *Financial threats in H1 2019.* Recuperado de: https://securelist.com/financial-threats-in-h1-2019/91899/.

RSA Laboratories (2002). *PKCS #1 v2.1: RSA Cryptography Standard.* Recuperado de: https://www.cryptrec.go.jp/en/cryptrec_03_spec_cypherlist_files/PDF/pkcs-1v2-12.pdf

Qi, Na; Pan, Jing; Ding, Qun (2011). The Implementation of FPGA-based RSA Public-key Algorithm and its Application in Mobile-phone SMS Encryption System, In *2011 First International Conference on Instrumentation, Measurement, Computer, Communication and Control* (pp. 700-703). Beijing, China. 10.1109/IMCCC.2011.178

Ren, Yonglin; Boukerche, Azzedine; Nelem, Richard (2010). Performance Evaluation of a Hybrid Cryptosystem with Authentication for Wireless Ad hoc Networks, *IEEE Global Telecommunications Conference GLOBECOM 2010* (pp. 6-10). Miami, FL, USA.

Saberi, Iman; Shojaie, Bahareh; Salleh, Mazleena; Niknafskermani, Mahan; Morteza, Seyyed (2012). Improving confidentiality of AES-CCMP in IEEE 802.11i. In *2012 International Joint Conference on Computer Science and Software Engineering (JCSSE)*. (pp. 82-86). Bangkok, Thailand. 10.1109/JCSSE.2012.6261930

Schneier, Bruce; Kelsey, John; Withing, Doug; Wagner, David; Hall, Chris; Ferguson, Niels (1998). *The Twofish Encryption Algorithm: A 128-Bit Block Cipher*. EE.UU.: Wiley.

Shivkumar, S.; Umamaheswari, G. (2011). Performance Comparison of Advanced Encryption Standard (AES) and AES Key Dependent S-Box - Simulation Using MATLAB. In *2011 International Conference on Process Automation, Control and Computing*. (pp.1-6). Coimbatore, India.

Tecnology Day revista IT Now (2013). *Tecnology Day*. Recuperado de: http://revistaitnow.com/2013/05/seguridad/aumentan-ataques-a-dispositivos-moviles/

Tovar, José; Diaz, Juan; Quiñones, Getssy; Pabón, Anabella; García, José (2018). Development of an information system for teleoperated physical rehab care service via Internet. Pilot case: patients with mild knee injury who live in geographically vulnerable zones. *DYNA*, 8(205), 284-293. http://dx.doi.org/10.15446/dyna.v85n204.67961.

Uskov, Alexander (2012). Information Security of IPsec-based Mobile VPN: Authentication and Encryption Algorithms Performance. *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications* (pp.1042-1048). Liverpool, UK. 10.1109/TrustCom.2012.187

Umaparvathi, M.; Varughese, Dharmishtan (2010). Evaluation of symmetric encryption algorithms for MANETs. In 2010 IEEE International Conference on Computational Intelligence and Computing Research (pp.1-3). Coimbatore, India. 10.1109/ICCIC.2010.5705754

Zhang, Wen-Xiang; Xiao, Si-You; Zhang, Yi (2010). Research on Image-Text Encryption Techniques in Mobile Communications, *2010 Second WRI Global Congress on Intelligent Systems* (pp.115-118). Wuhan, China. 10.1109/GCIS.2010.184

ISO/IEC (2005). ISO/IEC 27001: 2005 *Tecnología de la información - Técnicas de seguridad - Especificación para un sistema de gestión de seguridad de la información.* Ginebra, Suiza: ISO/IEC.