PROPOSTA DE OTIMIZAÇÕES NO ALGORITMO AES PARA USO EM APLICAÇÕES IOT

YURI SILVA VAZ[1]; RAFAEL IANKOWSKI SOARES[2]; JÚLIO C. B. MATTOS[3]

[1]*Universidade Federal de Pelotas – ysvaz@inf.ufpel.edu.br*
[2]*Universidade Federal de Pelotas – rafael.soares@inf.ufpel.edu.br*
[3]*Universidade Federal de Pelotas – julius@inf.ufpel.edu.br*

1. INTRODUÇÃO

O termo Internet das Coisas, do inglês *Internet of Things* (IoT), pode ser descrito como a presença pervasiva de uma variedade de dispositivos, tais como sensores e/ou atuadores, que têm a capacidade de interação entre si com o objetivo de atingirem objetivos em comum (MARWEDEL, 2021). Dispositivos IoT estão cada vez mais presentes em nosso cotidiano, podendo eles estarem dentro de casas, indústrias ou até mesmo monitorando cidades inteiras (SADHU et al. 2022). Previsões estimam que estas aplicações crescerão em número, atingindo a marca de 29 bilhões de dispositivos no ano de 2027 (SINHA, 2023).

Uma vez que as estatísticas indicam um grande crescimento em número de aplicações IoT, há que se preocupar com a segurança da informação que transitará via rede, pois vazamentos e/ou manipulações destes dados podem gerar grandes transtornos para seus usuários. Algoritmos criptográficos podem ser uma boa estratégia para proteção das aplicações, porém, a grande maioria das aplicações IoT são desenvolvidas em dispositivos com recursos computacionais limitados, além de serem alimentados por bateria, logo, preocupação com desempenho e consumo energético é primordial para o bom funcionamento das aplicações (TSAI et al. 2018).

Diante deste cenário apresentado, onde de um lado há a preocupação com a segurança das aplicações e do outro há uma restrição de desempenho e consumo energético, surge um desafio de como aplicar algoritmos criptográficos nestas aplicações de forma que o impacto em termos computacionais seja o menor possível. Há uma classe de algoritmos criptográficos chamada *lightweight cryptography* (LWC), focados em aplicações IoT e baixo consumo. Estes algoritmos são projetados para ocupar pouco espaço de armazenamento, além de utilizarem pouco recurso computacional (TOSHIHIKO, 2017).

O AES é um algoritmo criptográfico de chave simétrica, ou seja, a chave que é utilizada para encriptar um dado é a mesma utilizada para decriptar. Este algoritmo segue sendo utilizado atualmente, tendo sido validado tanto por agências governamentais quanto pela comunidade acadêmica. Isto se deve ao fato de ter uma implementação relativamente simples garantindo segurança e razoável tempo de execução. O AES pode ser facilmente decomposto em funções menores, o que o torna um algoritmo amplamente utilizado na literatura para análises (MEDELEANU et al. 2015).

Este trabalho propõe uma versão *lightweight* do algoritmo AES, tanto em termos de desempenho como em termos de consumo de memória. Para isto, foram realizadas otimizações nas funções *SubBytes* e *MixColumns.* O diferencial deste trabalho comparado aos demais é a combinação de otimizações, sendo elas tanto em desempenho, melhorando o tempo de execução, quanto em consumo de memória, obtendo uma versão bem mais leve do algoritmo. Todas as

otimizações foram validadas, aprovadas em requisitos de segurança e comparadas ao algoritmo AES versão original. Foi possível verificar uma redução considerável, tanto em termos de tempo de execução quanto em consumo de memória, garantindo um ótimo nível de segurança, assim, sendo viável sua utilização em aplicações IoT.

2. METODOLOGIA

A otimização proposta para o estágio de *SubBytes* se baseia na implementação de uma s-box menor, contendo apenas 16 entradas. Esta s-box é modelada como um vetor com índice de 0 a 15 e seu conteúdo indo de F a 0. A Fig. 1 exemplifica a ideia da tabela desenvolvida.

Índice	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Valor	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Figura 1 - S-Box Desenvolvida com 16 Bytes

A matriz de estado é composta por 16 entradas, dispostas entre as 4 linhas e 4 colunas da matriz. Cada byte, representado em hexadecimal, é dividido em dois dígitos, onde estes dois dígitos são utilizados como índice da s-box proposta. Quando há um dígito de A a F, em hexadecimal, este é utilizado como decimal, variando de 10 a 15, respectivamente. Utilizando este método, não há necessidade de uma s-box para encriptação e uma s-box para decriptação (s-box reversa), pois utilizamos a mesma tabela para ambas operações. Utilizando o hexadecimal 3B como exemplo, ao passar pelo estágio de *SubBytes*, ele se tornaria C4 e, ao passar pelo processo de decriptação, utilizando a mesma s-box, ele voltaria ao seu valor 3B original.

O algoritmo original do AES armazena duas s-box de 256 bytes cada, uma para o processo de encriptação e outra para o processo de decriptação, totalizando 512 bytes de consumo em armazenamento. Já a alternativa proposta utiliza apenas 16 bytes, pois utiliza uma única tabela tanto para encriptar quanto para decriptar a informação do bloco, resultando numa redução de 96,87% em bytes.

Já a otimização sugerida para o estágio de *MixColumns* se baseia na utilização de operações mais simples e menos custosas em termos de recursos computacionais. Para isto, foi desenvolvida uma função que realiza multiplicação em campos de Galois pela constante 2 que serve de base para todas as outras multiplicações, ou seja, todas as outras multiplicações são feitas utilizando esta função e utilizando funções xor para somas. As multiplicações são sempre pelas constantes 2 e 3, no processo de encriptação, e pelas constantes 9, 11, 13 e 14 no processo de decriptação. Além disto, o laço de repetição presente originalmente na função do AES foi expandido, baseado na técnica de *loop unrolling.*

Esta otimização proposta para o estágio de *MixColumns* apresentou bons ganhos em termos de tempo de execução, tornando o algoritmo AES bem mais rápido comparado a sua versão original.

3. RESULTADOS E DISCUSSÃO

As otimizações implementadas foram avaliadas em termos de desempenho, utilizando como métricas o tempo de encriptação e o consumo de memória, e em termos de segurança, avaliando efeito avalanche e critério de balanceamento, além de serem submetidas aos testes do NIST.

Para obtenção dos valores dos tempos de execução, cada teste foi executado 1000 vezes em série e foi extraída a média aritmética destes valores. Como entrada do algoritmo foram utilizadas strings de tamanho 10, 25, 70, 100, 1000, 2000 e 10000 bytes. A Tabela 1 traz o comparativo dos tempos de execução entre o algoritmo AES original e o algoritmo proposto. Pode-se observar uma redução significativa em termos de tempo de encriptação, atingindo uma redução de aproximadamente 90% com o uso da otimização proposta.

Tabela 1 - Tempos de Encriptação do AES Original e Modificado

Entrada(B)	AES Original(µs)	AES Modif.(µs)	Redução(%)
10	40,96	8,23	79,92
25	74,60	12,92	82,68
70	172,95	21,96	87,30
100	243,57	29,25	87,99
1000	2130,79	220,75	89,64
2000	4222,14	434,43	89,71
10000	21091,54	2167,31	89,72

Já para extração dos resultados de consumo de memória, foram compiladas ambas versões dos algoritmos na IDE do Arduino. Como é possível verificar na Tabela 2, foi possível obter uma redução de 31,82% em memória de programas e 89% em memória dinâmica.

Tabela 2 - Consumo de Memória do AES Original e Modificado

	AES Original(B)	AES Modif.(B)	Redução(%)
Programas	6392	4358	31,82
Dinâmica	894	98	89,04

Em questões de segurança, o teste de efeito avalanche foi utilizado para verificar qual o percentual de bits é alterado na mensagem cifrada quando é realizada uma mudança de apenas 1 bit na mensagem original. Foi obtido um efeito avalanche de 50,41%, o que está de acordo para este teste, onde devem-se atingir valores próximos de 50%. Já em questões de critério de balanceamento, que é um teste utilizado para verificar a eficiência da s-box, foram obtidos 48,92% de 0's e 51,08% de 1's. O algoritmo modificado apresentou uma melhor distribuição de 0's e 1's quando comparado ao algoritmo original. Por último, a modificação proposta foi submetida aos testes do NIST, no qual 6 *strings* de 1000 bytes cada, geradas aleatoriamente, foram aprovadas em cada um dos testes disponíveis.

4. CONCLUSÕES

Este trabalho propôs otimizações em dois estágios do algoritmo AES, *SubBytes* e *MixColumns*, onde se buscou redução de consumo de memória e redução de tempo de execução, respectivamente. A modificação realizada em *SubBytes* consistiu em implementar uma s-box menor, saindo de duas s-boxes de 256 bytes cada do algoritmo original, para apenas uma s-box de 16 bytes. Já no estágio *MixColumns*, a modificação se concentrou em utilizar operações de adição e multiplicação mais leves, fazendo com que o algoritmo tivesse uma execução mais rápida. A média de redução, entre os 7 diferentes tamanhos de entrada avaliados, quando comparados ao AES original, foi de 86,71%, levando o algoritmo a ter um tempo de execução bem menor. Já em termos de consumo de memória, houve uma redução de 31,82% em consumo de memória de programas e 89,04% em memória dinâmica, tornando o algoritmo bem mais leve, fazendo com que seja mais viável sua utilização em dispositivos com recursos computacionais limitados. Por último, o algoritmo proposto foi validado em termos de segurança com os testes de efeito avalanche e de balanceamento, além de ser aprovado no teste de aleatoriedade do NIST. Concluindo, é possível afirmar que este *lightweight* AES é apropriado para utilização em aplicações IoT, pois foram atingidas grandes reduções tanto em consumo de memória quanto em tempo de execução. Como trabalhos futuros, serão avaliadas ambas versões, original e proposta, do algoritmo AES em plataformas apropriadas para desenvolvimento de aplicações IoT, com o intuito de extrair métricas tanto de desempenho quanto de consumo energético.

5. REFERÊNCIAS BIBLIOGRÁFICAS

MARWEDEL, P. **Embedded System Design**. Dortmund, Germany: Springer, 2021.

SADHU, P.K.; YANAMBAKA, V.P.; ABDELGAWAD, A. Internet of Things: Security and solutions survey. **Sensors,** v.22, n.19, 2022.

SINHA, S. **State of IoT 2023: Number of connected IoT dvices growing 16% to 16.7 billion globally**. IoT Analytics, 24 mai. 2023. Acessado em 6 set. 2023. Online. Disponível em: https://iot-analytics.com/number-connected-iot-devices/

TSAI, K.L.; HUANG, Y.L.; LEU, F.Y.; YOU, I.; TSAI, C.H. AES-128 Based Secure Low Power Communication for LoRaWAN IoT Environments. **Special Section on Security and Trusted Computing for Industrial Internet of Things,** v.6, 2018.

TOSHIHIKO, O. **Lightweight Cryptography Aplicable to Various IoT Devices.** NEC, 2017. Acessado em 6 set. 2023. Online. Disponível em: https://www.nec.com/en/global/techrep/journal/g17/n01/170114.html

MEDELEANU, F.; RACUCIU, C.; ROGOBETE, M. Consideration About The Possibilities To Improve Aes s-box Cryptographic Properties By Multiplication. **Proceedings Of The Romanian Academy,** v.16, p. 339-344, 2015.