ANÁLISE DE DESEMPENHO DE ALGORITMOS CRIPTOGRÁFICOS PÓS QUÂNTICOS EM SISTEMAS OPERACIONAIS EMBARCADOS

PERFORMANCE ANALYSIS OF POST-QUANTUM CRYPTOGRAPHIC ALGORITHMS IN EMBEDDED OPERATING SYSTEMS

Felipe Kendi Alves Yamamoto[1], [i]
Leandro Poloni Dantas[2], [ii]
Marcones Cleber Brito da Silva[3], [iii]
Luiz Carlos Canno[4], [iv]
Fernando Simplicio de Sousa[5], [v]

Data de submissão: (26/05/2023) Data de aprovação: (24/07/2023)

RESUMO

O progresso no desenvolvimento de computadores quânticos tem trazido muita especulação sobre quando esses dispositivos estarão prontos para serem utilizados de forma útil por grandes governos e corporações. Sem dúvida, este advento trará grandes benefícios tecnológicos, mas também trará novos problemas a serem solucionados. Um destes problemas consiste na criptografia de chaves públicas utilizada atualmente em todo mundo, baseada principalmente nos algoritmos RSA e ECC/ECDSA, que seriam facilmente quebrados por um computador quântico. Devido a isso, a comunidade acadêmica e o setor privado têm se esforçado para criar algoritmos que sejam resistentes ao computador quântico, os chamados algoritmos pós-quânticos. É de interesse da comunidade que esses algoritmos sejam executados de maneira eficiente no maior número possível de casos de uso. Entretanto, os sistemas embarcados apresentam particularidades que podem tornar-se empecilhos para o uso desses algoritmos. O presente trabalho tem como objetivo analisar como alguns dos principais algoritmos pós quânticos propostos se comportam em um ambiente de sistema embarcado popularmente utilizado (Raspberry Pi), discutindo questões de processamento e uso de memória.

Palavras-chave: algoritmos pós quânticos; criptografia de chaves públicas; algoritmos criptográficos assimétricos; sistemas embarcados.

ABSTRACT

The progress in the development of quantum computers has brought much speculation about when these devices will be ready for practical use by large governments

[1] Pós-graduando em Sistemas Embarcados no SENAI Anchieta. E-mail: felipe.yamamoto2@senaisp.edu.br.
[2] Professor Dr. na Faculdade de Tecnologia SENAI São Paulo – Campus “Anchieta”. E-mail: leandro.poloni@sp.senai.br
[3] Professor Me. na Faculdade de Tecnologia SENAI São Paulo – Campus “Anchieta”. E-mail: marcones.silva@sp.senai.br.
[4] Professor Especialista na Faculdade de Tecnologia SENAI São Paulo – Campus “Anchieta”. E-mail: luis.canno@sp.senai.br.
[5] Professor Me. na Faculdade de Tecnologia SENAI São Paulo – Campus “Anchieta”. E-mail: fernando.simplicio@sp.senai.br.

and corporations. Undoubtedly, this advent will bring great technological benefits, but it will also bring new problems to be solved. One of these problems is the public key cryptography currently used worldwide, mainly based on RSA and ECC/ECDSA algorithms, which would be easily broken by a quantum computer. Due to this, the academic community and the private sector have been striving to create new algorithms that are resistant to the quantum computer, called Post-Quantum Algorithms. It is of interest to the community that these algorithms be efficiently executed in as many use cases as possible. However, embedded systems have peculiarities that can become obstacles to the use of these algorithms. The present work aims to analyze how some of the main proposed post-quantum algorithms behave in a popularly used embedded system environment (Raspberry Pi), discussing processing and memory usage issues.

Keywords: post-quantum algorithms; public key cryptography; asymmetric cryptographic algorithms; embedded systems.

1 INTRODUÇÃO

Um dos problemas atuais na área da criptografia é a segurança das chaves públicas utilizadas em todo o mundo. Essas chaves são baseadas principalmente nos algoritmos RSA e ECC/ECDSA, que são considerados seguros para a computação clássica. No entanto, com o avanço da computação quântica, esses algoritmos correm o risco de serem quebrados rapidamente (KOBLITZ; MENEZES, 2016).

A computação quântica oferece uma capacidade de processamento exponencialmente maior do que os computadores clássicos, graças à utilização de *qubits*, que são unidades de informação quântica. Essa tecnologia tem o potencial de quebrar os sistemas criptográficos atuais, incluindo os algoritmos de criptografia de chaves públicas (DATTANI; BRYANS, 2014).

O algoritmo *Rivest–Shamir–Adleman* (RSA), por exemplo, se baseia na dificuldade de fatoração de números grandes para garantir a segurança das chaves. Da mesma forma, os *Elliptic Curve Cryptography* (ECC) e *Elliptic Curve Digital Signature Algorithm* (ECDSA) que são amplamente utilizados em sistemas criptográficos modernos, também são vulneráveis a ataques de computação quântica. Esses algoritmos se baseiam na dificuldade de resolver o chamado problema do logaritmo discreto em curvas elípticas (BORGES; MOREIRA; FERREIRA, 2015).

Os algoritmos de curva elíptica utilizam a propriedade matemática das curvas elípticas para fornecer segurança criptográfica. O problema do logaritmo discreto em curvas elípticas é essencialmente a tarefa de encontrar um número inteiro desconhecido, chamado de logaritmo discreto, quando aplicado a uma operação matemática específica em uma curva elíptica. Em outras palavras, dado um ponto *P* na curva elíptica e o valor resultante *Q* após aplicar uma multiplicação repetida n vezes a P, o desafio é encontrar o valor inteiro *n* (KOBLITZ; MENEZES, 2016).

A segurança desses algoritmos reside no fato de que encontrar o logaritmo discreto em curvas elípticas é uma tarefa computacionalmente difícil. A complexidade cresce exponencialmente à medida que o tamanho da curva e o número de bits dos parâmetros envolvidos aumentam. Essa dificuldade é explorada para proteger as chaves criptográficas geradas pelos algoritmos de curva elíptica.

No entanto, em um cenário de computação quântica avançada, os algoritmos de curva

elíptica se tornam vulneráveis a ataques. Os computadores quânticos poderiam usar algoritmos específicos, como o algoritmo de Shor, para resolver o problema do logaritmo discreto em curvas elípticas de maneira eficiente, colocando em risco a segurança desses sistemas criptográficos (LAPIERRE; LAPIERRE, 2021). Portanto, é necessário desenvolver algoritmos criptográficos pós-quânticos que possam resistir a esses ataques.

Para lidar com essa questão, estão sendo desenvolvidos algoritmos de criptografia pós-quântica, que são resistentes aos ataques de computação quântica. Esses algoritmos estão sendo projetados para garantir a segurança das comunicações e das informações mesmo em um cenário em que computadores quânticos avançados estejam disponíveis.

A transição para sistemas criptográficos pós-quânticos é um desafio complexo, pois envolve a atualização de uma ampla gama de infraestruturas e protocolos de segurança. No entanto, é um passo necessário para garantir a segurança das comunicações no futuro da computação quântica.

Os órgãos públicos, assim como as entidades privadas norte-americanas, adotam os algoritmos criptográficos de acordo com os padrões estabelecidos pelo *National Institute of Standards and Technology* (NIST). Devido à significativa influência das entidades norte-americanas na área de segurança da informação, os padrões definidos pelo NIST se tornaram uma referência global. Um exemplo notável é o AES (Advanced Encryption Standard), um padrão de criptografia simétrica estabelecido pelo NIST em 2001, que continua sendo amplamente utilizado e reconhecido internacionalmente até os dias de hoje. Essa adoção e reconhecimento refletem a confiança depositada nos padrões de criptografia definidos pelo NIST, que desempenham um papel fundamental na garantia da segurança das comunicações e na proteção dos dados sensíveis em todo o mundo.

Para estabelecer um novo padrão de algoritmo criptográfico, o NIST abre um processo de seleção de algoritmos, no qual critérios técnicos e de usabilidade devem ser atendidos. Cabe então à comunidade submeter propostas de algoritmos para serem analisadas tanto pelos técnicos do NIST quanto pelo restante da comunidade internacional (academia e setor privado). No caso do AES, o algoritmo vencedor foi o *Rijndael*, criado pelos criptógrafos belgas Joan Daemen e Vincent Rijmen (JOAN; VINCENT, 2002). Já no caso dos algoritmos criptográficos pós quânticos, o NIST iniciou o processo de padronização em 2016, solicitando submissões de propostas. Desde então, foram realizadas três rodadas de análise e eliminação de propostas, sendo que das 69 propostas inicialmente submetidas, apenas 7 delas avançaram até a terceira rodada.

Em julho de 2022, o NIST divulgou a lista dos quatro algoritmos escolhidos como candidatos para padronização, embora tenha anunciado uma quarta rodada de seleção para novos algoritmos alternativos (NIST, 2022). Ou seja, embora já tenham sido realizadas três rodadas de seleção, ainda existe bastante incerteza na comunidade em relação aos algoritmos até aqui propostos. Isso deve-se ao fato de que foram encontradas vulnerabilidades em muitos dos algoritmos propostos que pareciam bastante seguros, como, por exemplo, o Rainbow (BEULLENS, 2022) e o SIKE (CASTRYCK; DECRU, 2022). Além disso, questões de usabilidade, como performance e o tamanho das chaves e das assinaturas digitais geradas, também trazem dúvidas de quão satisfatórios serão esses algoritmos na prática.

Os algoritmos selecionados como candidatos para padronização pelo NIST foram: CRYSTALS-KYBER, CRYSTALS-Dilithium, FALCON e SPHINCS+ e representam propostas promissoras para resistir aos ataques de computação quântica.

O CRYSTALS-KYBER é um algoritmo de criptografia de chave pública pós-quântica,

projetado para resistir a ataques realizados por computadores quânticos. Ele faz parte do projeto CRYSTALS (*Cryptographic Suite for Algebraic Lattices*) e é baseado em problemas matemáticos relacionados a reticulados (lattices). Uma das suas principais vantagens é seu equilíbrio entre segurança e desempenho. Ele possui várias variantes, como Kyber512, Kyber768 e Kyber1024, que diferem em termos de tamanho de chave e nível de segurança (AVANZI et al., 2017). Esta pesquisa limita-se no estudo da variante Kyber512.

O CRYSTALS-Dilithium é um algoritmo de assinatura digital pós-quântica desenvolvido como parte do projeto CRYSTALS. Ele é projetado para resistir a ataques realizados por computadores quânticos e é considerado um dos principais candidatos para substituir os algoritmos de assinatura digital atualmente utilizados. Este algoritmo possui três variantes: Dilithium-2, Dilithium-3 e Dilithium-5, que diferem em termos de segurança e desempenho. A escolha da variante depende do nível de segurança desejado e das restrições de recursos do sistema (LYUBASHEVSKY, et al., 2020). Esta pesquisa limitou-se no estudo do Dilithium-3.

O FALCON é um algoritmo de assinatura digital pós-quântico, projetado para ser resistente a ataques realizados por computadores quânticos. Ele pertence à família de algoritmos de criptografia baseados em reticulados (*lattice-based cryptography*) (SONI et al., 2021).

O SPHINCS+ é projetado para ser altamente seguro e eficiente, mesmo em sistemas com recursos limitados. Ele utiliza uma construção de árvore hash iterada e emprega funções hash criptográficas resistentes a ataques quânticos, garantindo a integridade e a autenticidade das assinaturas digitais (BERNSTEIN et al., 2019).

Em sistemas embarcados, para estabelecer a comunicação segura e confiável entre dispositivos, a forma mais comumente utilizada é através do uso de TLS (Transport Layer Security), protocolo de segurança implementado acima da camada TCP/IP. Para o TLS funcionar de maneira apropriada, utiliza-se criptografia de chaves públicas para verificação de identidade de cliente/servidor e para estabelecimento da chave simétrica de comunicação entre eles. Os algoritmos atualmente utilizados para essas finalidades serão substituídos por algoritmos criptográficos pós quânticos no futuro e, portanto, os dispositivos embarcados deverão ser capazes de executar esses algoritmos de forma satisfatória.

O objetivo deste artigo é medir o consumo de memória e o tempo de execução de quatro algoritmos pós-quânticos: CRYSTALS-KYBER, CRYSTALS-Dilithium, FALCON e SPHINCS+ quando executados em um sistema embarcado (Raspberry Pi Model B). Além desses, o algoritmo RSA foi testado e seus resultados analisados. Foram realizadas medições do consumo de memória e tempo de execução para cada um dos algoritmos analisados em diferentes tarefas criptográficas, como geração de pares de chaves criptográficas, assinatura digital, verificação de assinatura, cifração com chave pública e decifração com chave privada.

2 METODOLOGIA

A fim de verificar o desempenho dos algoritmos criptográficos pós quânticos, foram realizados testes de performance e uso de memória dos quatro algoritmos candidatos da terceira rodada de seleção do NIST, apresentados na Tabela 1.

Tabela 1: Algoritmos pós quânticos analisados neste trabalho.

Cifração com chave pública	Assinatura Digital
CRYSTALS-KYBER	CRYSTALS-Dilithium FALCON SPHINCS+

Fonte: elaborado pelos autores.

Embora todos sejam algoritmos de chave pública, existe uma diferenciação entre algoritmos utilizados para cifrar dados (geralmente uma chave simétrica que será encapsulada) e algoritmos utilizados para assinatura digital/verificação de assinatura (utilizados, por exemplo, na verificação de uma cadeia de certificados de identidade).

O ambiente de execução dos algoritmos foi uma placa Raspberry Pi Model B. Essa placa possui processador ARMv8 Cortex-A53 Quad-core, clock de 1.2 GHz e memória RAM de 1 GB. Nela, utilizou-se sistema operacional Raspbian versão 3.2.

Por tratar-se de um processo público de seleção, os autores dos algoritmos propostos disponibilizam documentação sobre seu funcionamento, embasamento teórico e também implementações de referência. Essas implementações podem ser encontradas nas páginas oficiais dos algoritmos (DUCAS et al., 2021), (AVANZI et al., 2019), (PORNIN, 2019), (BERNSTEIN et al., 2019). Importante salientar que, embora algumas dessas implementações possuam otimizações para execução em ambiente x86, neste trabalho foram utilizadas as implementações sem otimização para arquitetura específica, a fim de evitar que o desempenho de um dado algoritmo seja superior a outro devido a essas otimizações. Além disso, é importante também ressaltar que todas as implementações utilizadas foram escritas em linguagem C e compiladas com a mesma versão de compilador e *flags* de otimização.

A fim de medir o uso de memória dos algoritmos, foi utilizado o programa *Valgrind* (NETHERCOTE; SEWARD, 2007) juntamente com a ferramenta *massif*. Ambas as ferramentas estão disponíveis para instalação no sistema operacional Raspbian por meio do pacote *apt*. Ao serem utilizadas em conjunto, essas ferramentas permitiram medir o consumo de memória *heap* e *stack* de um programa durante a execução dos algoritmos estudados.

Para isso, foram desenvolvidos *scripts* de testes utilizando as APIs disponibilizadas pelos autores dos algoritmos. Os *scripts* (Makefiles) fornecidos por DUCAS et al. (2021) foram adaptados para facilitar compilação dos programas para a geração de pares de chaves criptográficas, assinatura digital, verificação de assinatura, cifração com chave pública e decifração com chave privada.

Para avaliar o desempenho dos algoritmos criptográficos pós-quânticos com um algoritmo padrão amplamente utilizado em sistemas embarcados, foram também realizados testes de desempenho com o algoritmo RSA. Utilizou-se a biblioteca criptográfica mbedTLS (versão 3.2.1) (BAKKER, 2019) para implementar esses testes. O algoritmo RSA foi configurado com uma chave de 3072 bits, enquanto os algoritmos pós-quânticos foram testados com parâmetros que garantem um nível equivalente de segurança criptográfica, ou seja, uma segurança de 128 bits.

3 RESULTADOS E DISCUSSÕES

A seguir são apresentadas as medições do tempo de execução e consumo de memória para cada um dos algoritmos analisados, em diferentes tarefas criptográficas.

3.1 RSA

É relevante destacar que o RSA pode ser utilizado tanto para assinatura quanto para cifração de dados, ao contrário dos algoritmos pós-quânticos analisados.

A Tabela 2 apresenta a Taxa de Tarefas Criptográficas (TTC) a cada um segundo com o algoritmo RSA utilizando uma chave de 3072 bits. Esta taxa corresponde a quantidade máximas de tarefas criptográficas geradas e atendidas pelo dispositivo embarcado utilizado (Raspberry Pi) nas configurações físicas definidas no ensaio.

Tabela 2: Taxa de Tarefas Criptográficas (TTC) por segundo com RSA.

Funcionalidade	TTC (s)
Geração de par de chaves	0,07
Assinaturas	5
Verificações	289
Cifrações	285
Decifrações	5,3

Fonte: elaborado pelos autores.

Os resultados referentes ao consumo de memória (em bytes) das chaves públicas e privadas, assim como o objeto de assinatura e texto cifrado geradas pelo algoritmo RSA estão apresentados na Tabela 3. Observa-se que, no algoritmo RSA, os objetos gerados possuem o mesmo tamanho das chaves (3072 bits).

Isso ocorre devido à própria natureza do algoritmo RSA e sua dependência no uso de chaves com tamanho específico.

No RSA, a segurança criptográfica é baseada na dificuldade de fatorizar números primos muito grandes. O tamanho das chaves utilizadas no algoritmo está diretamente relacionado à segurança oferecida por ele. Chaves maiores são consideradas mais seguras, pois a fatorização de números primos grandes se torna computacionalmente mais difícil.

Tabela 3 – Consumo de memória dos objetos gerados com o algoritmo RSA.

Objeto	bytes
Chave pública	384
Chave privada	384
Assinatura	384
Texto cifrado	384

Fonte: elaborado pelos autores.

No caso específico mencionado, um tamanho de chave de 3072 bits é considerado robusto para garantir a segurança em aplicações criptográficas. Essa escolha de tamanho de chave é resultado de uma análise de segurança e do equilíbrio entre a segurança desejada e a eficiência computacional. Chaves de tamanhos diferentes podem ser utilizadas dependendo dos requisitos de segurança e dos recursos computacionais disponíveis.

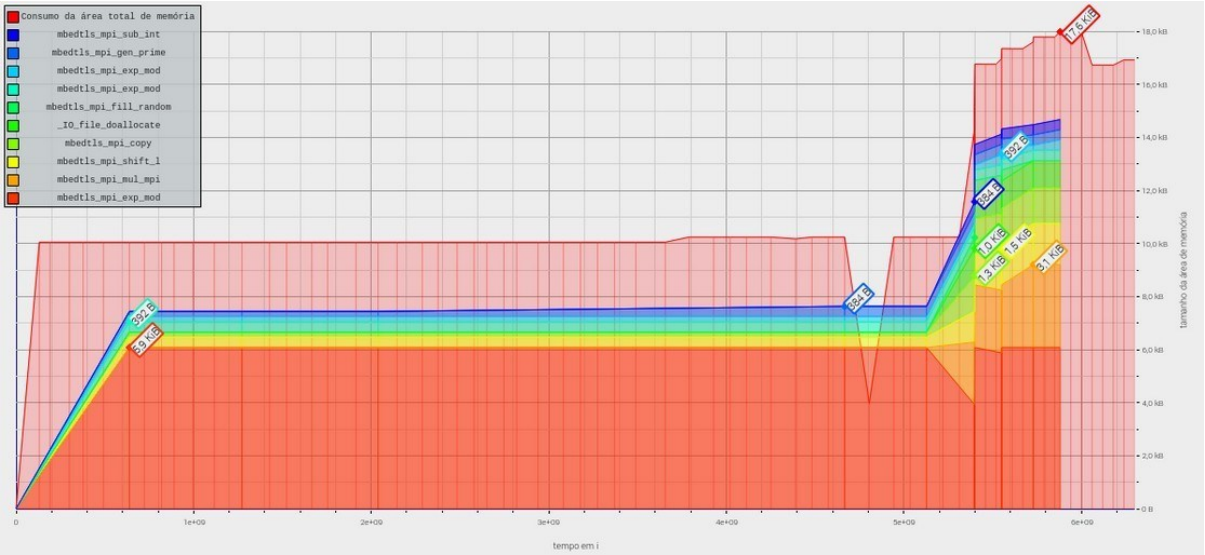
Em relação ao consumo de memória RAM durante a execução do algoritmo RSA, os resultados obtidos são apresentados na Figura 1.

A Figura 1 apresenta o consumo de memória RAM de cada uma das principais funções da biblioteca mbedTLS utilizadas nos testes, as quais são detalhadas a seguir.

A função *mbedtls_mpi_sub_int* é responsável por subtrair um número inteiro de um objeto do tipo *mbedtls_mpi* (*Multiple Precision Integer* – MPI). O MPI é usado para manipular e realizar operações matemáticas com números inteiros que excedem o limite de tamanho suportado pelos tipos de dados inteiros padrão da linguagem de programação.

Já a função *mbedtls_mpi_gen_prime* é responsável pela geração de números primos aleatórios e de tamanho especificado, essenciais para a geração das chaves criptográficas nos algoritmos RSA.

Figura 1 – Consumo de memória RAM utilizada com algoritmo RSA.



Fonte: elaborado pelos autores.

A função *mbedtls_mpi_exp_mod* é responsável por realizar a operação de exponenciação modular entre MPI e é uma operação matemática que envolve elevar um número a uma potência e calcular o resultado utilizando a operação de módulo em relação a um valor especificado. Esta função é usada para na cifração como na decifração dos dados.

Similarmente, a função *mbedtls_mpi_mul_mpi* é responsável por realizar a multiplicação de dois números de precisão múltipla.

Observou-se na Figura 1, que a função *mbedtls_mpi_exp_mod* foi responsável pelo maior consumo de memória durante os testes. Assim, deduz-se que as operações de exponenciação modular entre MPI podem consumir uma quantidade significativa de memória, principalmente quando são realizadas com números inteiros muito grandes.

O máximo consumo de memória RAM utilizado foi de 17,6 KiB.

3.2 CRYSTALS-DILITHIUM

Os resultados apresentados na Tabela 4 indicaram que o desempenho do algoritmo CRYSTALS-Dilithium foi superior ao do RSA em termos da taxa de tarefas criptográficas. A geração de pares de chaves teve um aumento de 467042,86 %, as assinaturas tiveram um aumento de 2480 %, e as verificações apresentaram um aumento de 12,80 %.

Tabela 4: Taxa de Tarefas Criptográficas (TTC) por segundo com CRYSTALS-Dilithium 3.

Funcionalidade	TTC (s)
Geração de par de chaves	327
Assinaturas	129
Verificações	326

Fonte: elaborado pelos autores.

No entanto, como observado na Tabela 5, o consumo de memória dos objetos do CRYSTALS-Dilithium é maior em comparação com os objetos equivalentes do RSA. Por exemplo, o tamanho do objeto de chave privada é aproximadamente 941,67 % maior no CRYSTALS-Dilithium. Esse aumento de tamanho pode representar um desafio caso seja necessário armazenar esses objetos em dispositivos embarcados com restrições de memória.

Tabela 5: Consumo de memória dos objetos gerados com o algoritmo CRYSTALS-Dilithium 3.

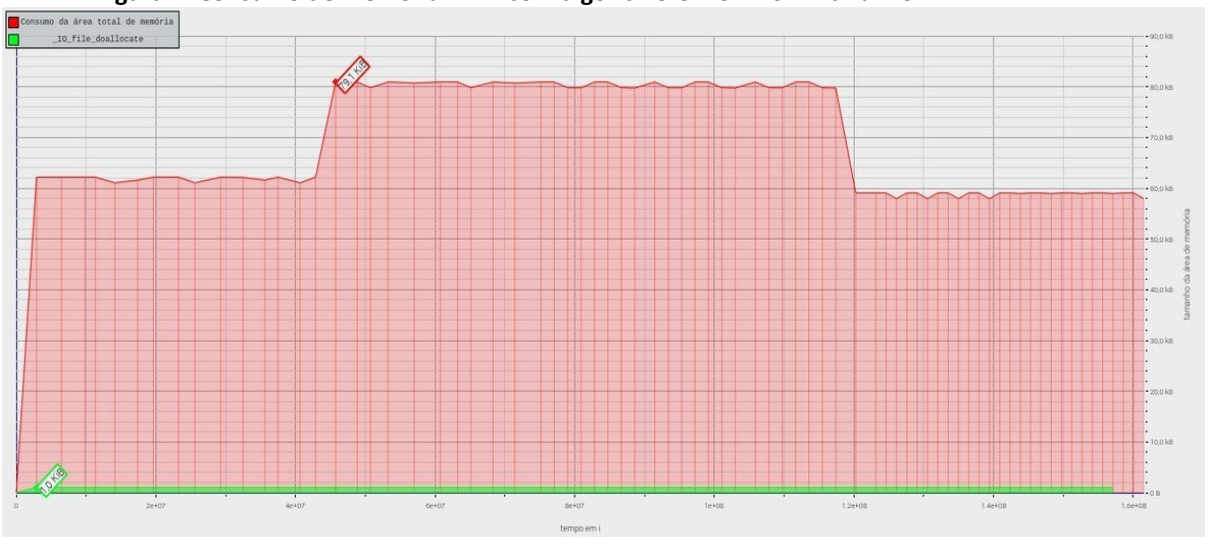
Objeto	bytes
Chave pública	1952
Chave privada	4000
Assinatura	3293

Fonte: elaborado pelos autores.

A comparação entre desempenho e consumo de memória é crucial para avaliar a viabilidade e adequação dos algoritmos criptográficos em diferentes cenários de uso. O CRYSTALS-Dilithium demonstrou um desempenho superior em relação à taxa de tarefas criptográficas, porém, é importante ponderar o consumo de memória para garantir uma implementação eficiente e compatível com as restrições do sistema em questão.

Além disso, o consumo máximo de memória durante a execução do CRYSTALS-Dilithium foi substancialmente maior que a do RSA, conforme apresentado na Figura 2, chegando a 79,1 KiB, resultando em um aumento de 349,43 %.

Figura 2: Consumo de memória RAM com algoritmo CRYSTALS-Dilithium 3.



Fonte: elaborado pelos autores.

3.3 SPHINCS+

A Tabela 6 apresenta a taxa de tarefas criptográficas do algoritmo SPHINCS+. Foi observado que, embora a taxa de geração de pares de chaves seja superior ao RSA, com um aumento de 23185,71 %, o desempenho do SPHINCS+ nas assinaturas e verificações foi inferior ao do RSA.

No entanto, a Tabela 7 revela que o consumo de memória das chaves pública e privada é menor no SPHINCS+, enquanto o consumo de memória na assinatura é maior. Especificamente, o objeto de assinatura consumiu 17152 bytes de memória. Esse consumo de memória pode representar um desafio em dispositivos com limitações de memória, especialmente ao lidar com longa cadeias de certificados digitais que exigem a verificação de várias assinaturas.

Tabela 6: Taxa de Tarefas Criptográficas (TTC) por segundo com algoritmo SPHINCS+.

Funcionalidade	TTC (s)
Geração de par de chaves	16,3
Assinaturas	0,66
Verificações	10,13

Fonte: elaborado pelos autores.

Tabela 7 - Consumo de memória dos objetos gerados com o algoritmo SPHINCS+.

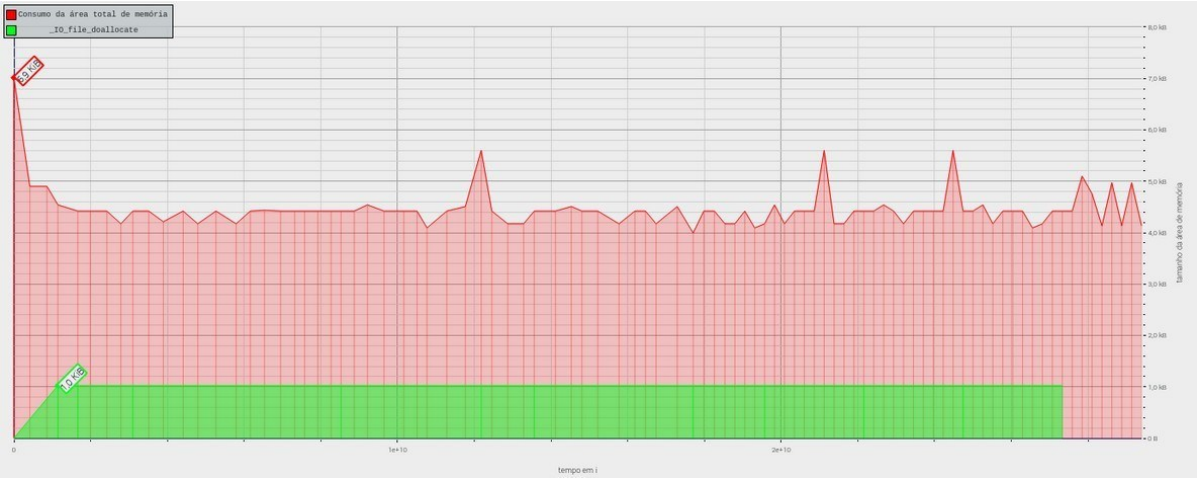
Objeto	bytes
Chave pública	32
Chave privada	64
Assinatura	17152

Fonte: elaborado pelos autores.

Durante a execução do algoritmo SPHINCS+, foi observado que o consumo de memória RAM foi inferior ao do RSA, com uma redução de 60,80 %. O valor máximo de consumo de memória foi de 6,9 KiB. Essa redução no consumo de memória é uma vantagem significativa do SPHINCS+ em comparação com o RSA. Com uma demanda de memória menor, o algoritmo SPHINCS+ se mostra mais eficiente em termos de utilização de recursos de memória.

Essa economia de memória é particularmente relevante em sistemas com restrições de memória, como dispositivos embarcados e sistemas com recursos limitados. O menor consumo de memória do SPHINCS+ permite uma melhor adaptação a esses cenários.

Figura 3 - Consumo de memória RAM com algoritmo SPHINCS+.



Fonte: elaborado pelos autores.

3.4 FALCON-512

A Tabela 8 apresenta a taxa de tarefas criptográficas do algoritmo FALCON-512. Observa-se que o algoritmo FALCON, foi superior ao do RSA em termos da obteve um desempenho melhor principalmente da taxa de tarefas criptográficas. A geração de par de chaves teve um aumento de 17042,86 %, as assinaturas tiveram um aumento de 6220 %, e as verificações apresentaram um aumento de 546,37 %. Esse diferencial é importante para sistemas embarcados que precisam verificar cadeias de certificados digitais (e.g. no fechamento da conexão TLS).

A Tabela 9 apresenta o consumo de memória (em bytes) dos objetos gerados com o algoritmo FALCON-512. Percebeu-se que o consumo de memória dos objetos foi maior que dos gerados no RSA.

Tabela 8: Taxa de Tarefas Criptográficas (TTC) por segundo com algoritmo FALCON-512.

Funcionalidade	TTC (s)
Geração de par de chaves	12
Assinaturas	316
Verificações	1868

Fonte: elaborado pelos autores.

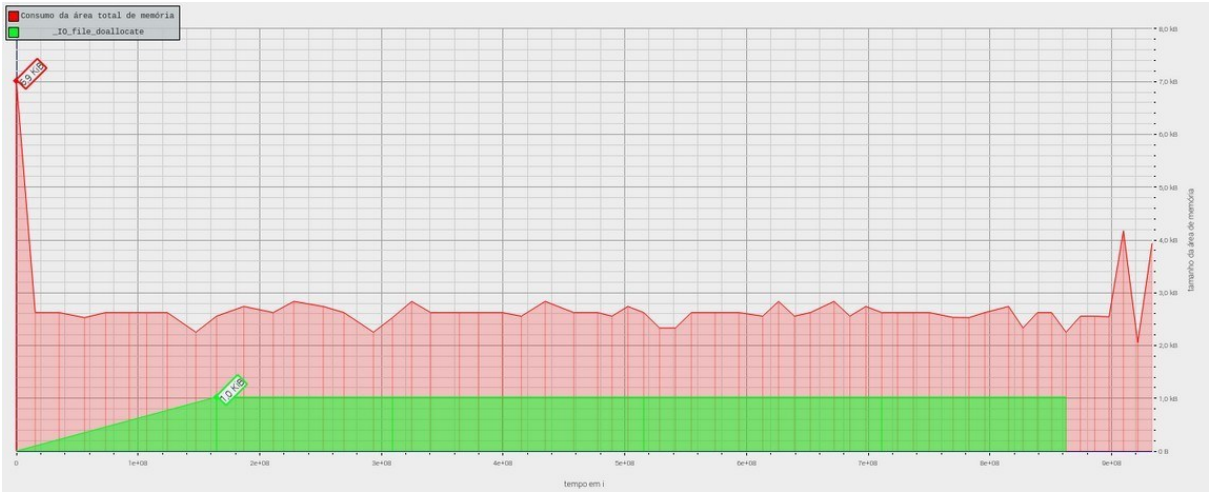
Tabela 9: Consumo de memória dos objetos gerados com o algoritmo FALCON-512.

Objeto	bytes
Chave pública	897
Chave privada	1281
Assinatura	809

Fonte: elaborado pelos autores.

Em relação ao uso de memória, o máximo consumo de memória foi de 6.9 KiB, conforme apresentado na Figura 4.

Figura 4 - Consumo de memória RAM com algoritmo FALCON-512.



Fonte: elaborado pelos autores.

3.5 CRYSTALS-KYBER

Por fim, os resultados do algoritmo CRYSTALS-Kyber, que é utilizado especificamente para cifração/decifração e não para assinatura digital, foram apresentados nas Tabelas 10 e 11, utilizando os parâmetros do Kyber 512.

Tabela 10 - Taxa de Tarefas Criptográficas (TTC) por segundo com CRYSTALS-Kyber 512.

Funcionalidade	TTC (s)
Geração de par de chaves	1748
Cifrações	1300
Decifrações	1222

Fonte: elaborado pelos autores.

Tabela 11 - Consumo de memória dos objetos gerados com CRYSTALS-Kyber 512.

Objeto	Bytes
Chave pública	800
Chave privada	1632
Texto cifrado	768

Fonte: elaborado pelos autores.

A taxa de tarefas criptográficas obtida com o CRYSTALS-Kyber foi superior às obtidas com o RSA. A geração de par de chaves teve um aumento de 2497042,86 %, as cifrações tiveram um aumento de 356,14 %, e as decifrações apresentaram um aumento de 22956,60 %. O máximo consumo de memória RAM foi 10,9 KiB, conforme apresentado na Figura 5.

Figura 5: Consumo de memória RAM com algoritmo CRYSTALS-Kyber 512.

Fonte: elaborado pelos autores.

4 CONCLUSÃO

Em conclusão, este estudo analisou o desempenho de quatro algoritmos criptográficos pós-quânticos: CRYSTALS-Kyber, CRYSTALS-Dilithium, FALCON e SPHINCS+. O objetivo foi medir o consumo de memória e o tempo de processamento desses algoritmos em diferentes tarefas criptográficas.

Os resultados mostraram que o algoritmo CRYSTALS-Dilithium teve um desempenho superior ao RSA em termos da taxa de tarefas criptográficas, apresentando um aumento significativo na geração de pares de chaves, assinaturas e verificações. No entanto, o consumo de memória dos objetos do CRYSTALS-Dilithium foi maior em comparação com o RSA, o que pode ser problemático em dispositivos com restrições de memória.

Por sua vez, o algoritmo SPHINCS+ apresentou uma taxa de geração de pares de chaves superior ao RSA, mas um desempenho inferior nas assinaturas e verificações. No entanto, seu consumo de memória foi menor em algumas áreas, como nas chaves pública e privada. No entanto, a assinatura do SPHINCS+ exigiu um consumo maior de memória.

Em relação ao CRYSTALS-Kyber, o estudo se concentrou na cifração/decifração, e os resultados foram apresentados de acordo com os parâmetros do Kyber 512.

Essas análises forneceram *insights* valiosos sobre o desempenho e o consumo de memória desses algoritmos pós-quânticos, permitindo uma comparação entre eles e o algoritmo RSA, amplamente utilizado em sistemas embarcados. Essas informações são fundamentais para avaliar a viabilidade e a adequação dos algoritmos pós-quânticos em diferentes cenários de uso, considerando as restrições de memória e os requisitos de desempenho.

No contexto da evolução da computação quântica, os resultados deste estudo contribuem para o avanço no desenvolvimento de algoritmos criptográficos seguros e eficientes, capazes de resistir aos ataques realizados por computadores quânticos. Essas informações são relevantes para a pesquisa e implementação de soluções de segurança criptográfica em sistemas embarcados e outros ambientes que requerem comunicações seguras e confiáveis.

REFERÊNCIAS

AVANZI, R. et al. **Crystals-kyber. NIST, Tech**. Rep, 2017. Disponível em: https://csrc.nist.gov/csrc/media/Presentations/2022/crystals-kyber-update/images-media/session-1-schwabe-crystals-kyber-pqc2022.pdf. Acesso em: 17 jul. 2023.

BAKKER, A. P. **mbedtls. tls. mbed. org**. 2019. Disponível em: https://www.trustedfirmware.org/projects/mbed-tls/. Acesso em: 17 jul. 2023.

BEULLENS, W. Breaking rainbow takes a weekend on a laptop. *In*: ANNUAL INTERNATIONAL CRYPTOLOGY CONFERENCE, CRYPTO 2022, 42., 2022. Santa Barbara, CA, USA, August 15–18, 2022, **Proceedings**[…] Part II. 2022. p. 464–479. Disponível em: https://eprint.iacr.org/2022/214.pdf. Acesso em: 17 maio 2023.

BERNSTEIN, D. J. et al. The sphincs+ signature framework. *In*: PROCEEDINGS OF THE 2019 ACM SIGSAC CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY. [s.n.], 2019. p. 2129–2146. Disponível em: https://doi.org/10.1145/3319535.3363229. Acesso em: 17 jul. 2023.

BORGES, F. G.; MOREIRA, M. A. R.; FERREIRA, R. M. ECDSA (Elliptic Curve Digital Signature Algorithm). *In*: CONTECSI-INTERNATIONAL CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGY MANAGEMENT, 6., 2015. Disponível em: https://www.tecsi.org/contecsi/index.php/contecsi/6contecsi/paper/viewFile/ 2784/1606. Acesso em: 17 jul. 2023.

CASTRYCK, W.; DECRU, T. An efficient key recovery attack on sidh (preliminary version). Cryptology ePrint Archive, p. Paper–2022, 2022. Disponível em: https://www.esat.kuleuven.be/cosic/publications/article-3569.pdf. Acesso em: 17 mai. 2023.

DATTANI, N. S.; BRYANS, N. **Quantum factorization of 56153 with only 4 qubits. arXiv preprint arXiv:1411.6758**, 2014. Disponível em: https://arxiv.org/pdf/1411.6758.pdf. Acesso em: 17 jul. 2023.

DUCAS, L. et al. **Crystals-dilithium algorithm specifications and supporting documentation**. 2019. Disponível em: https://pq-crystals.org/dilithium/data/dilithium-specification.pdf. 2021. Acesso em: 17 mai. 2023.

JOAN, D.; VINCENT, R. **The design of rijndael**: Aes-the advanced encryption standard. Information Security and Cryptography. Springer, 2002. Disponível em: https://link.springer.com/book/10.1007/978-3-662-04722-4. Acesso em: 17 maio 2023.

KOBLITZ, N.; MENEZES, A. A riddle wrapped in an enigma. **IEEE Security & Privacy**, v. 14, n. 6, p. 34–42, 2016. Disponível em: https://eprint.iacr.org/2015/1018.pdf. Acesso em: 17 jul. 2023.

LaPIERRE, R. **Introduction to quantum computing**, [s. l.]: Springer, 2021. Disponível em: https://link.springer.com/book/10.1007/978-3-030-69318-3 . Acesso em: 17 jul. 2023.

LYUBASHEVSKY, V. et al. **Crystals-Dilithium**. 2020. Disponível em: https://csrc.nist.gov/CSRC/media/Presentations/crystals-dilithium-round-3-presentation/images-media/session-1-crystals-dilithium-lyubashevsky.pdf. Acesso em: 17 jul. 2023.

NETHERCOTE, N.; SEWARD, J. Valgrind: a framework for heavyweight dynamic binary instrumentation. **ACM Sigplan notices**, ACM New York, NY, USA, v. 42, n. 6, p. 89–100, 2007. Disponível em: http://webcluster.cs.columbia.edu/~junfeng/ 09fa-e6998/papers/valgrind.pdf. Acesso em: 17 jul. 2023.

NIST. **PQC Standardization process**: announcing four candidates to be standardized, plus fourth round candidates. 2022. Disponível em: https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4#standardization. Acesso em: 17 mai. 2023.

PORNIN, T. New efficient, constant-time implementations of falcon**. Cryptology ePrint Archive**, 2019. Disponível em: https://eprint.iacr.org/2019/893.pdf. Acesso em: 17 mai. 2023.

SONI, D. et al. **Falcon**. *In*: Hardware architectures for post-quantum digital signature schemes, Springer, p. 31–41, 2021. Disponível em: https://link.springer.com/book/10.1007/978-3-030-57682-0. Acesso em: 17 jul. 2023.

SOBRE O(S)AUTOR(ES)

[i] **FELIPE KENDI ALVES YAMAMOTO**

Engenheiro de Computação (2013) pela (Unicamp). Possui mais de 10 anos de experiência na área de Defesa e Segurança da Informação, tendo atuado ativamente no desenvolvimento de sistemas embarcados em projetos sensíveis das Forças Armadas do Brasil e do setor privado. Atualmente, trabalha com desenvolvimento de software. Aluno do Curso de Pós-Graduação em Sistemas Embarcados pela Faculdade SENAI .

[ii] **LEANDRO POLONI DANTAS**

Engenheiro (2004) e Doutor (2018) em Engenharia Elétrica pelo Centro Universitário FEI. Atuou por 15 anos na indústria eletrônica no desenvolvimento de novos produtos. Desde 2009, vem lecionando em cursos de pós-graduação, graduação e de nível técnico em diferentes instituições paulistanas. Atualmente é professor na Faculdade de Tecnologia SENAI e no Insper. https://orcid.org/0000-0003-3674-336X

[iii] **MARCONES CLEBER BRITO DA SILVA**

Tecnólogo em Mecatrônica Industrial (2011), Engenheiro Mecatrônico (2013) e Especialista em Engenharia de Manutenção Industrial pela Centro universitário Eniac (2013). Mestre em Tecnologia Nuclear (2020) pela Universidade de São Paulo. Desde 2011, vem lecionando em cursos de nível técnicos e de graduação. Atualmente é professor da Faculdade de Tecnologia SENAI e na FESA. https://orcid.org/0000-0002-3690-1682

[iv] **LUIZ CARLOS CANNO**

Graduado em Tecnologia de Automação Industrial (2009) com Especialização em Gestão Empresarial pela Universidade Nove de Julho (2012), e Especialização em Docência na Educação Profissional e Tecnológica pelo SENAI CETIQT (2015). Professor na Faculdade de Tecnologia SENAI do curso de Tecnologia em Eletrônica Industrial e Pós-graduação em Sistemas Embarcados. https://orcid.org/0000-0001-9331-9309

[v] **FERNANDO SIMPLICIO DE SOUSA**

Professor da Faculdade SENAI no curso de Pós-Graduação em Sistemas Embarcados. Mestre em Engenharia Elétrica pela Universidade Federal do ABC (UFABC) e Pós-Graduado (Lato Sensu) pela Universidade Mackenzie. Graduado em Gestão de Pequenas e Médias Empresas pela UNIP e em Projetos Mecânicos pela Faculdade de Tecnologia de São Paulo (UNESP/FATEC-SP).http://lattes.cnpq.br/4579382987984065
https://orcid.org/0009-0009-5760-4845