Faculdade de Tecnologia de Americana "Ministro Ralph Biasi"

Curso Superior de Tecnologia em Segurança da Informação

Gustavo Vinícius Marques
Thiago Talpo

COMPARAÇÃO DO TEMPO DE *HANDSHAKE* DO ALGORITMO PÓS-QUÂNTICO ML-KEM COM O ECDHE EM VPN

Americana, SP

2025

Gustavo Vinicius Marques
Thiago Talpo

COMPARAÇÃO DO TEMPO DE *HANDSHAKE* DO ALGORITMO PÓS-QUÂNTICO ML-KEM COM O ECDHE EM VPN

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Segurança da Informação sob a orientação Prof.(a) Dra. Mariana Godoy Vazquez Miano.
Área de concentração: Criptografia.

Orientadora: Profª. Drª. Mariana Godoy Vazquez Miano

Este trabalho corresponde à versão final do Trabalho de Conclusão de Curso apresentado por Gustavo Vinícius Marques e Thiago Talpo e orientado pela Profª. Drª. Mariana Godoy Vazquez Miano.

Americana, SP
2025

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana Ministro Ralph Biasi- CEETEPS Dados Internacionais de Catalogação-na-fonte

MARQUES, Gustavo Vinicius

Comparação do tempo de handshake do algoritmo pós-quântico ML-KEM com o ECDHE em VPN. / Gustavo Vinicius Marques, Thiago Talpo – Americana, 2025.

39f.

Monografia (Curso Superior de Tecnologia em Segurança da Informação) - - Faculdade de Tecnologia de Americana Ministro Ralph Biasi – Centro Estadual de Educação Tecnológica Paula Souza

Orientadora: Profa. Dra. Mariana Godoy Vazquez Miano

1. Computação quântica 2. Criptografia 3. VPN – rede de computadores. I. MARQUES, Gustavo Vinicius, II. TALPO, Thiago III. MIANO, Mariana Godoy Vazquez IV. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana Ministro Ralph Biasi

CDU: 681518
681.518.5
681.519VPN

Elaborada pelo autor por meio de sistema automático gerador de ficha catalográfica da Fatec de Americana Ministro Ralph Biasi.

Gustavo Vinícius Marques

Thiago Talpo

Comparação do tempo de handshake do algoritmo pósquântico ml-kem com o ecdhe em vpn

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Curso Superior de Tecnologia em Segurança da Informação pelo Centro Paula Souza – FATEC Faculdade de Tecnologia de Americana Ministro Ralph Biasi.
Área de concentração: Segurança da informação.

Americana, 01 de dezembro de 2025.

Banca Examinadora:

Mariana Godoy Vazquez Miano
Doutora
Fatec Americana "Ministro Ralph Biasi"

José William Pinto Gomes
Especialista
Fatec Americana "Ministro Ralph Biasi"

Luiz Carlos Caetano
Especialista
Fatec Americana "Ministro Ralph Biasi"

RESUMO

O presente trabalho tem como objetivo realizar um estudo comparativo entre o algoritmo de criptografia pós-quântico ML-KEM e todas suas variantes, e os algoritmos de criptografia tradicionais baseados em Curvas Elípticas utilizados atualmente como o protocolo ECDHE. Todos esses algoritmos serão aplicados em ambientes VPNs. O estudo abordou uma coleta de dados referentes ao tempo médio de conexão entre duas partes (*handshake*) a fim de analisar o desempenho de cada algoritmo. Desse modo, por meio de experimentos práticos e pesquisas teóricas, buscou-se avaliar o desempenho do ML-KEM em comparação aos algoritmos de troca de chave que são utilizados em VPN. O objetivo é evidenciar sua viabilidade para a futura implementação em ambientes seguros para a proteção de dados na sociedade, promovendo maior confiança e resiliência para a segurança da informação na era da Computação Quântica. Os experimentos evidenciaram a eficácia do algoritmo pós-quântico ML-KEM no quesito velocidade para o estabelecimento de uma conexão entre duas partes.

Palavras-chave: VPN; ML-KEM; ECDHE; criptografia pós-quântica; computação quântica.

ABSTRACT

The present work aims to conduct a comparative study between the post-quantum cryptography algorithm ML-KEM, in all its variants, and the traditional Elliptical Curve-based cryptography algorithms currently in use, such as the ECDHE protocol. All these algorithms will be applied within VPN environments. The study involved collecting data on the average connection time (handshake) between two parties in order to analyze the performance of each algorithm. Thus, through practical experiments and theoretical research, this work seeks to evaluate the performance of ML-KEM in comparison to the key exchange algorithms used in VPNs. The objective is to highlight its viability for future implementation in secure environments for societal data protection, promoting greater trust and resilience for information security in the Quantum Computing era. The experiments demonstrated the effectiveness of the post-quantum ML-KEM algorithm regarding the speed of establishing a connection between two parties.

Keywords: *VPN; ML-KEM; ECDHE; post-quantum cryptography; quantum computing.*

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC	Autoridade Certificadora
AES	*Advanced Encryption Standard*
APT	*Advanced Package Tool*
CPU	*Central Processing Unit*
CSR	*Certificate Signing Request*
ECC	*Elliptic Curve Cryptography*
ECDHE	*Elliptic Curve Diffie-Hellman Ephemeral*
IP	*Internet Protocol*
ML-KEM	*Module-Lattice-based Key-Encapsulation Mechanism*
NIST	*National Institute of Standards and Technology*
RAM	*Random Access Memory*
RSA	Rivest-Shamir-Adleman
SSL	*Secure Sockets Layer*
TLS	*Transport Layer Protocol*
UDP	*User Datagram Protocol*
VM	*Virtual Machine*
VPN	*Virtual Private Network*

SUMÁRIO

1 INTRODUÇÃO

Conforme a computação quântica avança, os sistemas de criptografia utilizados atualmente tornaram-se vulneráveis, especialmente quando se trata de segurança e eficácia. Alguns mecanismos de segurança utilizados, como o algoritmo *Elliptic Curve Diffie-Hellman Ephemeral* (ECDHE), serão vulneráveis aos ataques realizados por computadores quânticos. Essa ameaça iminente aos mecanismos de proteção utilizados como base da segurança da informação ocorre devido a possibilidade dos computadores quânticos solucionarem problemas matemáticos complexos, visto que para computadores clássicos, a resolução destes problemas se torna difícil, de acordo com as limitações computacionais atuais. Assim, o Instituto Nacional de Padrões e Tecnologia (NIST) tem desenvolvido modelos criptográficos que possam enfrentar essa ameaça, sendo uma alternativa os algoritmos pós-quânticos.

Dentre as novas opções, o *Module-Lattice Key Encapsulation Mechanism* (ML-KEM) aparece como alternativa para as operações que envolvam chaves criptográficas. Esse algoritmo tem como base estruturas matemáticas de reticulados e foi selecionado pelo NIST como modelo de padronização para algoritmos de criptografia pós-quântica. Este estudo visa comparar o algoritmo pós-quântico ML-KEM, com algoritmos tradicionais utilizados na troca de chaves criptográficas, como o ECDHE, analisando como parâmetro comparativo o tempo de duração médio para o início de uma comunicação segura (*handshake*) em uma VPN (*Virtual Private Network*).

Os algoritmos criptográficos, mais especificamente aqueles voltados para operações com chaves criptográficas, são essenciais para a segurança das comunicações realizadas. A escolha deles depende de sua estrutura matemática, da eficiência e da resistência a ataques. Analisando-se este cenário da evolução da computação quântica e através dos dados obtidos, procura-se evidenciar a diferença de mecanismos e ferramentas utilizadas atualmente com as que serão utilizados num futuro próximo e como essas alterações afetarão a segurança da informação.

A questão a ser tratada é o estudo comparativo entre o padrão criptográfico ECDHE e o algoritmo pós-quântico ML-KEM, comparando-se a eficiência destas ferramentas ao se realizar a operação de troca de chaves criptográficas no *handshake*. O foco é comparar os algoritmos, analisando-se estatísticas que

realmente sejam úteis para a área de Segurança da Informação. Como uma meta específica deste trabalho, inclui-se a coleta do tempo médio de *handshake* de cada um dos padrões em questão, para a realização de uma análise de desempenho.

A metodologia da pesquisa é prática e exploratória, consistindo na implementação de máquinas virtuais, com foco em coletar dados para uma análise comparativa entre o algoritmo de criptografia clássica ECDHE e o algoritmo pós-quântico ML-KEM, baseando-se no tempo médio que cada algoritmo utiliza para se comunicar com outra máquina por meio de uma VPN. Outras métricas como utilização de CPU, tamanho de chave criptográfica, tamanho de pacotes, entre outras métricas – embora relevantes para a área de segurança da informação – estão fora do escopo dessa pesquisa.

Para isso, será estruturado um conjunto composto por duas máquinas virtuais, sendo a primeira máquina dedicada à função de servidor VPN e a outra máquina à de cliente. As máquinas utilizarão o sistema Linux Debian na versão 13.1.0 virtualizado no VirtualBox em máquina local, garantindo o isolamento do ambiente e a reprodução dos testes.

A solução adotada será o *OpenVPN*, cuja pilha criptográfica poderá ser adaptada através da compilação de versões customizadas do *OpenSSL*. Com essa configuração, pretende-se estabelecer conexões seguras entre o cliente e o servidor e, durante esse processo, mensurar o tempo total dessa operação.

As medições serão realizadas com o *Wireshark* e, analisando-se os resultados obtidos, conduzir-se-á uma análise quantitativa e qualitativa, possibilitando a avaliação da viabilidade do ML-KEM como modelo criptográfico para a proteção em ambientes com VPN.

Visando os objetivos propostos, este estudo foi dividido em seções, como segue: a seção 2 apresenta o referencial teórico, com explicações de conceitos que serão necessários para o entendimento do trabalho; a seção 3 contém o desenvolvimento prático do trabalho, detalhando todas as ferramentas e métodos para a realização do mesmo; a seção 4 apresenta os resultados que foram obtidos durante o estudo e a seção 5 contém as conclusões finais do trabalho.

2 REFERENCIAL TEÓRICO

Esta seção detalha os assuntos referentes à computação quântica, computação clássica, criptografia clássica, criptografia pós-quântica e a ferramenta *Wireshark.* Primeiramente, aborda-se a computação quântica e clássica e, em seguida, o *bit* quântico e o algoritmo de Shor. Na sequência, a criptografia clássica e seus algoritmos utilizados atualmente. Na sequência, a criptografia pós-quântica e o seu algoritmo para troca de chaves, o ML-KEM. Ao final, são apresentados conceitos de VPN e das ferramentas *OpenSSL, OpenVPN* e *Wireshark*.

2.1 Computação quântica x computação clássica

A computação quântica tem se consolidado como uma área de estudo em constante expansão, principalmente devido ao seu potencial de superar significativamente as limitações da computação clássica quando trata-se de poder computacional. Esse avanço impacta diretamente a forma com que se aborda a proteção dos dados, tendo em vista que diversos mecanismos criptográficos atualmente utilizados tornar-se-ão vulneráveis diante do poder de processamento que os algoritmos quânticos possuem.

Alguns exemplos notórios envolvem a resolução de problemas com números primos, base dos sistemas criptográficos, que fundamentam sua segurança nessa dificuldade, como o RSA, por exemplo. Porém, essa não será uma dificuldade para os computadores quânticos, que podem solucionar problemas desse tipo de maneira exponencialmente mais rápida — até mesmo em questão de segundos. Essa mudança de paradigma fez com que surgisse a necessidade de novos métodos criptográficos resistentes a ataques quânticos (Dylan *et al*., 2023)

2.2 Qubit

O *qubit*, ou *bit* quântico, é a menor unidade de informação na computação quântica, sendo equivalente ao *bit* para a computação clássica. No entanto, diferentemente do bit clássico — que se restringe à representação de apenas um entre dois estados bem definidos, 0 ou 1 — o qubit não se limita a esses estados discretos.

Devido aos princípios da física quântica, como a sobreposição e o emaranhamento, o qubit pode existir simultaneamente em múltiplas combinações de estados. Isso significa que, por meio da sobreposição, um único qubit pode representar infinitas possibilidades entre os estados 0 e 1, ampliando significativamente o potencial computacional diante aos sistemas clássicos (Eunmi; Joonhee; Junki, 2024).

2.3 Algoritmo de Shor

Algoritmo quântico conhecido pela sua capacidade de fatorar números inteiros grandes exigindo um menor tempo e esforço em comparação a qualquer algoritmo clássico conhecido. Desenvolvido por Peter Shor no ano de 1994, utiliza os princípios da mecânica quântica e pode assumir os estados de sobreposição e emaranhamento de um bit quântico, o que permite realizar a fatoração de um número inteiro N em tempo polinomial, utilizando um computador quântico com capacidade suficiente para tal operação (Silva *et al.,* 2023).

Muitos dos sistemas criptográficos utilizados atualmente, como o RSA, dependem da dificuldade desta fatoração para proteger informações, o que o torna vulnerável devido à capacidade do algoritmo de Shor de fatorar rapidamente números inteiros grandes, colocando tais sistemas em risco potencial e, consequentemente, a proteção de dados e informações.

2.4 Criptografia clássica

A criptografia clássica fundamenta-se em problemas matemáticos que são considerados insolúveis com as máquinas atuais, como a dificuldade de fatorar números inteiros grandes, o que se tornou um mecanismo de proteção para os dados e base para algoritmos de criptografia atuais, como o RSA. (Kurose; Ross, 2021).

Atualmente, existem dois modelos criptográficos: simétricos e assimétricos. Na criptografia simétrica existe apenas uma chave para realizar a encriptação dos dados e a descriptografia do conteúdo, sendo assim, um método mais rápido e eficiente, porém não viável no quesito segurança. Na criptografia assimétrica é utilizada uma chave pública para realizar a criptografia do conteúdo e uma chave privada correspondente para a descriptografia. Deste modo, são modelos que requerem maior poder computacional para serem executados, como o RSA.

2.4.1 RSA

Trata-se de um algoritmo de criptografia assimétrica amplamente utilizado na criptografia moderna. Desenvolvido por Rivest, Shamir e Adleman em 1970, o RSA baseia-se no conceito de chave pública e privada, sendo a chave pública utilizada para criptografar e a chave privada utilizada para descriptografar. O RSA possui a sua segurança fundamentada na dificuldade computacional de fatorar grandes números primos, assegurando a proteção dos princípios da confidencialidade, integridade e autenticação na segurança da informação (Campos; Rosa, 2024).

De acordo com Kurose e Ross (2021), a geração de suas chaves se dá com a seguinte fórmula:

1. Escolha dois números primos grandes, p e q;
2. Calcula-se $n = pXq$, onde n é o módulo para criptografia e descriptografia;
3. Calcula-se a função totiente de Euler, sendo ela $\Phi(n) = (p-1)(q-1)$;
4. Seleciona-se um inteiro e, tal que $mdc(\Phi(n), e) = 1$ e $1 < e < \Phi(n)$;
5. Calcula-se d, sendo ele o inverso modular de $e \ mod \ \Phi(n)$;
6. A chave pública é n, e e a chave privada é n, d;

Para cifrar uma mensagem, utiliza-se a fórmula

$$C = M^e mod n,$$

onde M representa a mensagem e C é o texto cifrado. Além disso, a condição $M < n$ deve ser respeitada. O processo de descriptografia, a fim de recuperar a mensagem original, utiliza a seguinte fórmula

$$M = C^d mod n.$$

Além da criptografia, o RSA também é utilizado para garantir a autenticação digital, onde o processo da assinatura é realizado pela chave privada, enquanto a verificação é feita pela chave pública, ambas pertencentes ao remetente. Qualquer entidade pode validar a autenticidade da assinatura com a chave pública correspondente. Dessa forma, o RSA se torna aplicável em sistemas de segurança, como protocolos SSL/TLS, certificados digitais e sistemas de autenticação em redes seguras.

2.4.2 ECC

Segundo Ernesto (2023), a *Ellyptic Curve Cryptography* (ECC) é um algoritmo de criptografia assimétrica baseado na matemática das curvas elípticas sobre corpos finitos. Desenvolvido por Neal Koblitz e Victor Miller em 1985, sua segurança está fundamentada na dificuldade de resolver o problema do logaritmo discreto, sendo necessário encontrar k tal que $Q = kP$, onde P e Q são valores conhecidos. A equação de uma curva elíptica no contexto de corpos finitos é dada por

$$y^2 \equiv x^3 + ax + b \ (mod\ p).$$

No processo de geração de chaves do ECC, a chave privada d trata-se de um valor randômico escolhido entre 1 e $n - 1$, sendo n a ordem do ponto base G. A chave pública é gerada como resultado da equação $D = dG$. Para criptografar uma mensagem, utiliza-se a sua chave privada e pública do receptor por meio da fórmula $C = M + d_e D_r$, sendo *M* a mensagem e *C* o texto cifrado. Para decifrar o texto cifrado, o destinatário utiliza a sua chave privada e a chave pública do emissor, aplicando a fórmula $M = C - d_r D_e$.

Dentro da criptografia de curvas elípticas (ECC) estão os algoritmos específicos como ECDHE para realizar acordo de chaves, ECDSA (*Elliptic Curve Digital Signature Algorithm*) para assinaturas digitais, entre outras funções que são variantes desses algoritmos como o algoritmo X25519, que implementa o algoritmo ECDHE em curvas específicas (Azarderakhsh *et al*., 2021).

2.4.3 *Transport Layer Security*

Lançado no ano de 2018, o *Transport Layer Security* 1.3 (TLS) é um protocolo de criptografia focado em prover segurança em comunicações realizadas na internet. Foi criado como um sucessor de sua antiga versão o TLS 1.2. Essa nova versão do protocolo melhorou tanto em segurança quanto na questão de performance, sendo mais eficiente que seu antecessor, e ganhou novos recursos. Esses benefícios foram possíveis devido a mudanças na estrutura do protocolo: mudanças nos esquemas de segurança utilizados como base no protocolo para a proteção das informações e redução no processo de *handshake,* sendo este o processo que dá início a uma conexão entre duas partes, fazendo com que a eficiência do protocolo aumentasse (Zhou *et al*., 2024).

O aumento na eficiência do protocolo na sua versão 1.3 ocorreu devido à alteração que o protocolo sofreu no início de sua comunicação - no *handshake.* Na sua versão anterio,r o protocolo necessitava de duas rodadas para realizar a conexão: na primeira rodada, o cliente solicitava a conexão e o servidor respondia com o seu certificado digital e os parâmetros de cifra. Na segunda rodada, o cliente (após receber o certificado) enviava as informações de chave criptográfica para o servidor. Já na nova versão do protocolo, este processo foi reduzido para apenas uma rodada, onde juntamente na primeira etapa, o cliente enviava as informações de chaves e logo em seguida o servidor respondia com a outra parte da chave e certificado, não necessitando esperar a resposta do servidor, encurtando o processo e deixando o protocolo muito mais eficiente (Zhou *et al*., 2024).

2.5 Criptografia pós-quântica

Devido à ascensão dos computadores quânticos, o NIST e outras entidades da área de segurança têm se dedicado a estudos para minimizar os impactos deste avanço tecnológico, visto que alguns algoritmos quânticos, como o de Shor e o de Grover, são capazes de resolver problemas matemáticos da criptografia clássica, colocando em risco as informações e dados protegidos pelo sistema de criptografia assimétrica atual. Desde então, o NIST tem realizado chamadas para submissões de diversas implementações de criptografia assimétrica resistente a ataques tradicionais e quânticos com o intuito de padronizar um ou mais algoritmos (Amorim; Henriques, 2020).

O NIST propõe que seja seguido como referência cinco diferentes categorias de segurança conhecidas, como o AES e SHA, conforme a figura 1.

Figura 1: níveis de segurança para avaliação do algoritmo proposto ao NIST.

Nível	Descrição de Segurança
I	Tão difícil de quebrar quanto o AES128 (busca exaustiva)
II	Tão difícil de quebrar quanto o SHA256 (busca de colisão)
III	Tão difícil de quebrar quanto o AES192 (busca exaustiva)
IV	Tão difícil de quebrar quanto o SHA384 (busca de colisão)
V	Tão difícil de quebrar quanto o AES256 (busca exaustiva)

Fonte: Amorim e Henriques (2021).

Dessa forma, o NIST apresenta um passo importante para que existam padrões criptográficos que resistam não somente às ameaças oriundas dos computadores clássicos, mas também às ameaças dos computadores quânticos. A padronização de algoritmos de criptografia pós-quântica será essencial para garantir a proteção de dados de forma confiável e eficiente.

2.5.1 ML-KEM/CRYSTALS-KYBER

Segundo Karatsiolis *et al.* (2025), ML-KEM trata-se de um algoritmo de criptografia pós-quântico padronizado pelo NIST em 2024 para proteger as informações contra os ataques efetuados por computadores quânticos. É fundamentado no *CRYSTALS*-Kyber, um mecanismo de troca de chaves baseado em reticulados que permite o compartilhamento de chaves secretas de forma segura entre duas entidades para o estabelecimento de uma comunicação.

Um reticulado é um conjunto de pontos em um espaço de múltiplas dimensões. Os pontos são formados pela combinação linear de vetores linearmente independentes que compõe a base do reticulado, onde bases diferentes podem formar o mesmo reticulado. Uma base com os vetores pequenos e relativamente ortogonais é classificada como uma base boa; caso contrário, é uma base ruim devido a maior dificuldade em formar o reticulado em comparação a base boa (Ferro; Rampazzo; Henriques, 2021).

Sua segurança está fundamentada no problema *Module-LWE (Module Learning With* Errors) cuja dificuldade reside em resolver equações matriciais com adição de erros com operações modulares em reticulados. Além disso, possui três parâmetros: ML-KEM512, ML-KEM768 e o ML-KEM1024, que se diferenciam pelo tamanho das chaves, sendo elas de 128 bits, 192 bits e 256 bits, respectivamente.

2.6 OPENSSL

O OpenSSL é uma ferramenta de código aberto utilizada para a implementação de protocolos de segurança – como o *Secure Socket Layer* (SSL) e o *Transport Layer Security* (TLS) -, além de gerenciar chaves e certificados (Ricchizzi; Schwinne; Pelzl; 2025). O OpenSSL possui suporte aos algoritmos de criptografia pós-quântico padronizados - como o ML-KEM para troca de chaves, ML-DSA para assinatura digital e o SLH-DSA para assinatura digital baseado em *hash* sem estado – a partir da versão 3.5.

2.7 VPN

Uma VPN (*Virtual Private Network*) é um serviço que cria uma conexão segura e permite uma comunicação segura via rede pública, como a *internet*. A VPN utiliza o IPSec como método de tunelamento para encapsular o datagrama IP e estabelecer um canal de comunicação lógico seguro ao encriptar e autenticar os dados (Firdaouss *et al.,* 2020).

A VPN proporciona confidencialidade nas comunicações via rede pública devido à criptografia utilizada, o que gera a codificação das mensagens. Usuários fisicamente distantes das instalações de uma organização podem acessar recursos da rede interna remotamente.

2.8 OPENVPN

OpenVPN trata-se de uma ferramenta de VPN confiável que utiliza interfaces virtuais TUN/TAP — ou seja, realiza a criação de uma VPN tanto na camada de enlace, trabalhando com quadros, quanto na camada de rede, trabalhando com

pacotes — e suporta conexões ponto a ponto, bem como servidores com múltiplos clientes. A solução utiliza a biblioteca criptográfica *OpenSSL*. Para autenticação, são utilizados certificados, enquanto, para a criptografia, utilizam-se cifras assimétricas e simétricas (Hall, 2025).

2.9 Wireshark

Wireshark é uma ferramenta utilizada para a captura de pacotes que trafegam em uma rede em tempo real e gerar um relatório visual e detalhado sobre o que está passando pela rede, possibilitando a realização de uma análise desses pacotes e garantindo que não há nenhum agente malicioso trafegando pela rede (Wireshark, 2024). Utilizando essa ferramenta, pode-se ver em detalhes os protocolos como SSL/TLS e analisar a troca de certificados digitais, chaves de criptografia e assinaturas digitais.

3 DESENVOLVIMENTO

Nesta seção são exibidas as etapas necessárias para que fosse possível a coleta da duração média de *handshake* do ML-KEM e do ECDHE. Iniciou-se com a configuração do ambiente virtual e sistema operacional utilizado. Na sequência, atualizaram-se os pacotes já pertencentes ao sistema e foram instalados os pacotes do *OpenSSL* e *OpenVPN*. Em seguida, foram gerados os certificados digitais para realizar a conexão entre as máquinas virtuais e a configuração dos arquivos necessários para iniciar essa conexão.

3.1 Ambiente de trabalho

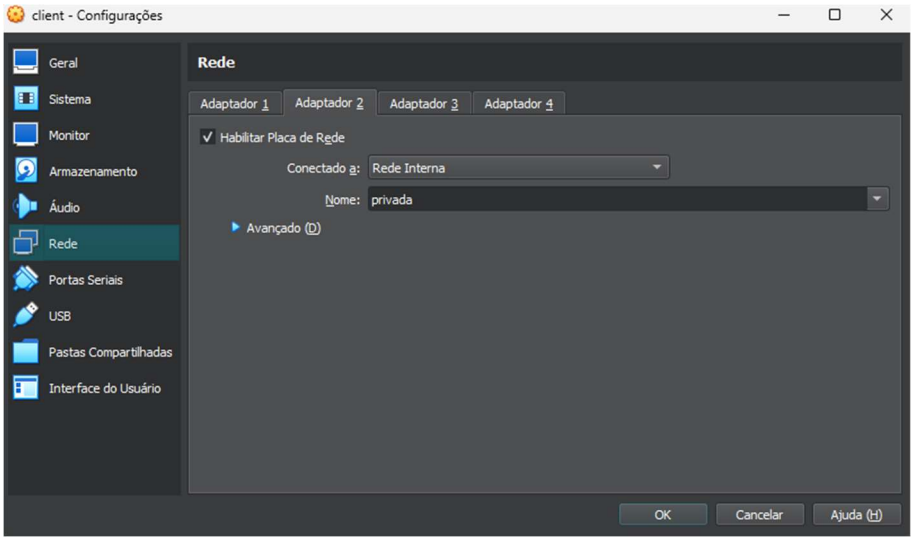
Utilizou-se o VirtualBox para realizar a virtualização de duas máquinas a partir da imagem debian-13.1.0-amd64.iso. Uma máquina realizou a função de um servidor de VPN, chamada servidor, enquanto a outra máquina realizou o papel de cliente, chamada cliente. Nesse cenário, a VM servidor trata-se de um clone da VM cliente. Cada máquina virtual foi provisionada com 4000MB de memória RAM e 2 núcleos de CPU, conforme a figura 2.

Figura 2: definição da memória RAM e CPU.

Fonte: autores (2025).

A figura 3 detalha a configuração das interfaces de redes configuradas nas máquinas virtuais, utilizando-se a placa de rede em modo de rede interna e nomeada como “privada”.

Figura 3: configuração da placa de rede.



Fonte: autores (2025).

O modo de rede interna permite que as máquinas virtuais façam uma comunicação de forma isolada, disponibilizando um ambiente apropriado para a realização de testes e experimentos.

3.1.1 Instalações dos pacotes necessários

Primeiramente, realizou-se a atualização dos pacotes contidos no sistema operacional com o comando *apt update && apt upgrade -y*. Dessa forma, o sistema operacional utilizará os pacotes mais recentes e estáveis, corrigindo falhas e erros de compatibilidade.

3.1.2 Instalação do *OpenSSL*

Para criar e gerenciar certificados digitais, bem como implementar conexões seguras entre duas entidades, precisou-se realizar a instalação do *OpenSSL* no sistema operacional. O *OpenSSL* fornecerá as funcionalidades em relação à criptografia, certificados e protocolos seguros.

A instalação da ferramenta foi efetuada com o gerenciador de pacotes nativo do Debian, o APT (*Advanced Package Tool*), que permite a instalação, atualização e

remoção de pacotes dentro do sistema. Para instalar o *OpenSSL*, utilizou-se o comando *apt install openssl -y*, conforme apresentado na figura 4.

Figura 4: Instalação *OpenSSL*.

```
cliente@cliente:~$ sudo apt install openssl -y
Instalando:
  openssl

Pacotes sugeridos:
  ca-certificates

Resumo:
  Atualizando: 0, Instalando: 1, Removendo: 0, Não atualizando: 0
  Tamanho de download: 1.494 kB
  Espaço necessário: 2.551 kB / 3.194 MB disponível

Obter:1 http://security.debian.org/debian-security trixie-security/main amd64 op
Obtidos 1.494 kB em 4s (353 kB/s)
Selecionando pacote previamente não selecionado openssl.
(Lendo banco de dados ... 131833 arquivos e diretórios atualmente instalados).
Preparando para desempacotar .../openssl_3.5.1-1+deb13u1_amd64.deb ...
Desempacotando openssl (3.5.1-1+deb13u1) ...
Configurando openssl (3.5.1-1+deb13u1) ...
Processando gatilhos para man-db (2.13.1-1) ...
```

Fonte: autores (2025).

Como pode-se observar, o gerenciador de pacotes instalou o *OpenSSL* com sucesso na versão 3.5.1. Esta versão inclui suporte à algoritmos de criptografia pós-quântica, não sendo necessária a instalação ou compilação de qualquer outro pacote para o uso do ML-KEM.

3.1.3 Instalação do OPENVPN

A respeito da instalação do *OpenvVPN,* utilizou-se o mesmo gerenciador de pacote para instalá-lo. O *OpenVPN* será responsável por criar um canal de enlace seguro entres as máquinas em que os dados são transmitidos. Com isso, executou-se o comando *apt install openvpn -y* para a implementação da VPN, conforme ilustrado na figura 5.

Figura 5: Instalação *OpenVPN*.

```
Preparando para desempacotar .../8-openvpn_2.6.14-1_amd64.deb ...
Desempacotando openvpn (2.6.14-1) ...
Configurando runit-helper (2.16.4) ...
Configurando libccid (1.6.2-1) ...
Configurando pcscd (2.3.3-1) ...
Created symlink '/etc/systemd/system/sockets.target.wants/pcscd.socket' → '/u
pcscd.service is a disabled or a static unit, not starting it.
Configurando libeac3:amd64 (1.1.2+ds+git20220117+453c3d6b03a0-1.1+b3) ...
Configurando opensc-pkcs11:amd64 (0.26.1-2) ...
Configurando libpkcs11-helper1t64:amd64 (1.30.0-1+b1) ...
Configurando easy-rsa (3.2.2-1) ...
Configurando openvpn (2.6.14-1) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/openvpn.service'
Configurando opensc (0.26.1-2) ...
Processando gatilhos para man-db (2.13.1-1) ...
Processando gatilhos para mailcap (3.74) ...
Processando gatilhos para desktop-file-utils (0.28-1) ...
Processando gatilhos para gnome-menus (3.36.0-3) ...
Processando gatilhos para libc-bin (2.41-12) ...
```

Fonte: autores (2025).

A figura 5 indica que o comando utilizado está buscando os pacotes do *OpenVPN* presentes nos repositórios do sistema. Além disso, o *OpenVPN* está sendo instalado na versão 2.6.14 e, após a instalação, será possível a criação do túnel virtual entres as máquinas envolvidas.

3.2 Configuração do OPENVPN

Esta seção foi dividida em duas partes onde a primeira está relacionada à criação de chaves e certificados, enquanto a segunda se trata dos arquivos de configurações do *cliente* e *servidor*.

3.2.1 Geração de chaves e certificados

Para configurar o acesso seguro entre o servidor de VPN e o cliente, é necessário realizar a emissão de chaves e certificados para que seja feita autenticidade do cliente ao servidor, dessa forma, o algoritmo RSA foi escolhido para ser utilizado na autenticação dos certificados com todos os algoritmos de troca de chaves. Para gerar a chave da autoridade certificadora (AC), utilizou-se o comando:

openssl genpkey - -algorithm rsa -out ca.key

Na sequência, utilizou-se a chave para gerar o certificado autoassinado da AC com o comando:

openssl req -x509 -new -nodes -key ca.key -days 365 -out ca.crt -subj "/CN=PQC".

O parâmetro *-x509* gera o certificado em formato internacional para autenticação. A opção *-new* requisita um novo certificado, enquanto *-nodes* possibilita o não uso de senha para se autenticar, sendo útil para testes. Em seguida, especifica-se a chave privada da autoridade certificadora (AC) para assinar o certificado com a opção *-key* e o arquivo contendo o conteúdo da chave *ca.key* e, logo após, define-se a validade com *-days,* qual o arquivo que irá armazenar o conteúdo do certificado com *-out* e o nome comum com o parâmetro *-subj*, sendo a identificação “CN=PQC”. A seguir, foi criada a chave privada do *servidor* e *cliente* com os comandos:

openssl genpkey -algorithm rsa -out servidor.key
openssl genpkey -algorithm rsa -out cliente.key

A opção *genpkey* é utilizada para gerar as chaves privadas das máquinas e *-algorithm* define o uso do RSA. As chaves geradas são de caráter sigiloso e exclusivas de cada parte da comunicação. Com posse das chaves, prosseguiu-se para o pedido de assinatura de certificado:

openssl req -new -key servidor.key -out servidor.csr -subj "/CN=PQC"

Neste processo, faz-se o uso da chave privada da entidade para gerar o pedido de assinatura de certificado (CSR), em que será contido a informação da entidade. Após a geração do CSR, a AC o assina para validar sua autenticidade e gerar o certificado digital do *servidor*por meio do comando:

openssl x509 -req -in servidor.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out servidor.crt -days 365

As opções *-CA* e *-CAkey* indicam, respectivamente, o certificado e chave privada da AC. Além disso, utilizou-se *-CAcreateserial* para a criação de um número

de sério exclusivo para o certificado. Em relação ao cliente, foram utilizados os mesmos comandos, porém alterando-se as saídas e entradas de arquivos, resultando na *cliente.key* e nos certificados *cliente.crt* e *cliente.csr.*

3.2.2 Arquivo de configuração *servidor.conf*

O arquivo *servidor.conf* estabelece alguns parâmetros de operação e de segurança, além de definir como a VPN será iniciada e os protocolos que serão usados para criar um canal seguro. A primeira diretiva definida trata-se da porta que a *OpenVPN* será executada, sendo a 1194, que além de utilizar o protocolo de transporte de dados UDP, possui baixa latência e maior desempenho. A *OpenVPN* é uma interface virtual do tipo túnel, cuja função é criar o túnel lógico criptografado e o encapsulamento de datagramas IP para que os dados trafeguem em uma rede de forma segura. A diretiva *tls-groups* é a mais relevante desta configuração diante do cenário apresentado, onde sua função é instruir o servidor a utilizar o ML-KEM-768 como método de acordo de chaves. Para que a negociação de chaves seja feita apenas pelo ML-KEM, utilizou-se o parâmetro *dh none* para desabilitar o uso estático do protocolo vigente *Diffie-Hellman*. A VPN está definida como *subnet* e utilizará a faixa de endereçamento IP 10.10.10.0/24 como rede privada. Essas configurações são demonstradas pela figura 6.

Figura 6: arquivo de configuração *servidor.conf*

```
servidor@servidor:/$ cat /etc/openvpn/server/servidor.conf
port 1194
proto udp
dev tun
user nobody
group nogroup

tls-groups MLKEM768
dh none
topology subnet
server 10.10.10.0 255.255.255.0
reneg-sec 0
cipher AES-256-GCM
auth SHA256
verb 3

ca /etc/openvpn/server/ca.crt
cert /etc/openvpn/server/servidor.crt
key /etc/openvpn/server/servidor.key
```

Fonte: autores (2025)

A diretiva *reneg-sec* é responsável por tratar as renegociações automáticas de chaves e, nesse caso, foi desativada para que fosse possível coletar a duração de *handshake* isoladamente, buscando maior precisão nas medições. A linha *cipher AES-256-GCM* especifica o uso do AES no modo Galois/Counter (GCM) como algoritmo simétrico para a cifragem dos dados dentro do túnel da VPN. No campo *ca* deve ser especificado o arquivo que contém o conteúdo do certificado da autoridade certificadora, enquanto os campos *cert* e *key* devem possuir os arquivos com o conteúdo do certificado e chave do servidor, respectivamente.

3.2.3 Arquivo de configuração *cliente.ovpn*

O arquivo *cliente.ovpn* é utilizado para o cliente fazer a conexão à VPN. É definida a interface virtual do tipo túnel e o protocolo UDP. A linha *remote 10.0.0.20 1194* estabelece a conexão remota no endereço IP 10.0.0.10 e porta 1194 – endereço IP do *servidor* e a porta em que o serviço está sendo executado, respectivamente. Além disso, o campo *resolv-retry 0, reneg-sec 0* e *auth-nocache* garantem que cada conexão realize o processo de *handshake* do ML-KEM sem algum tipo de *cache*. A diretiva *cipher* define o AES-256-GCM como algoritmo de cifra, enquanto a *auth* define o SHA256 como autenticação do canal para evitar alterações de mensagem por pessoas não autorizadas. Este arquivo de configuração pode ser visualizado através da figura 7.

Figura 7: arquivo de configuração *cliente.ovpn.*

```
cliente@cliente:~$ cat /etc/openvpn/client/cliente.ovpn
client
dev tun
proto udp
remote 10.0.0.20 1194
resolv-retry 0
nobind

tls-groups MLKEM768
reneg-sec 0
auth-nocache
cipher AES-256-GCM
auth SHA256
verb 3

<ca>
-----BEGIN CERTIFICATE-----
MIIC/TCCAeWgAwIBAgIUOJ2V9Xip/eHMgBGVYNkE10BCw7wwDQYJKoZIhvcNAQEL
BQAwDjEMMAoGA1UEAwwDUFFDMB4XDTI1MTAyNjE3MjQwOVoXDTI2MTAyNjE3MjQw
OVowDjEMMAoGA1UEAwwDUFFDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKC
AQEAnCIu5YCdR9k7Z0/cTgKYGWBj1wI656ER3aX6eXxtuXjaLPSE6bAWn+saLina
eM8oYO6VW548bl9mxOsLbn/zsXQ6XZG+7pUaOiSmCQHAdRubWF/x+yMGJqoEJHg2
vHMiUK7M7KLdn0rxfB2pg6ijzW6iKRe8CrfKq3olX/Sk4gxx6pdip0RZdGEM0sPW
qAydYgRVXa6iCq7AOcx5Nj9SRvC3als5vZx68mBzFxdAlr++UFkeD2yL5lPGKjLn
FW+AsrC1D3+BtusWIkZJ2DfiMoPDj1FibdYEswcx3IIqc6jjbVcFUXiz7khAz28I
7Savsyu1iHieof18BhO4nDTdZQIDAQABo1MwUTAdBgNVHQ4EFgQUZGS2nCqZfSou
bjdAbIi2ULan7g8wHwYDVR0jBBgwFoAUZGS2nCqZfSoubjdAbIi2ULan7g8wDwYD
VR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAQEAYhZdyb/y6/WzhcjqBKro
8l3NpptokWZRF+9xQucxLC6fdCfmyyOf83+xq8j6eOlBqRUa0u219rY+D1r55Qzz
UlK9DjvK4w4yBclIrMm+GS3F18zO3U00VfTtJAVLlx41hk7AQWMVxbm7ZI4rbtF3
e3ngySLwbjmLeRJSn9LOXH8fSzxyYly99c87Be1gWrsz9KnRaQbBLPAlwIE2pnYk
+0O2Ai9gsvfC5VWVAvRMvD8UUQwBvKFlwsCYl4+NYWZtybJhtjZrYe3F1ZuDTrHy
2+ZownIj9DKY9SHjQ+I8FBHUpYwzcdKi0rDrkx9YMc9QgkoBVyfyzd1Dezf27A9P
nQ==
-----END CERTIFICATE-----
</ca>

<cert>
-----BEGIN CERTIFICATE-----
```

Fonte: autores (2025)

As chaves e os certificados estão configurados dentro do arquivo, devendo os blocos <*ca*>, <*cert*> e <*key*> serem preenchidos com o conteúdo dos arquivos *ca.crt, cliente.crt* e *cliente.key*, respectivamente. A diretiva *tls-groups MLKEM768* permite ao cliente a utilização do *ML-KEM-768* como algoritmo de troca de chaves.

4 RESULTADOS

Após configurados os arquivos *servidor.conf* e *cliente.ovpn*, iniciou-se a VPN na VM *servidor* para receber a conexão com o comando *sudo su openvpn –config /opt/openvpn/server/servidor.conf.* Na figura 8, pode-se observar a mensagem *Initialization Sequence Completed* no servidor de VPN ao iniciar o OpenVPN, indicando uma espera pela conexão.

Figura 8: estado de espera por conexão ao iniciar o servidor de VPN.

```
servidor@servidor:/$ sudo openvpn --config /etc/openvpn/server/servidor.conf --verb 3
2025-10-27 06:49:28 Note: Kernel support for ovpn-dco missing, disabling data channel off
2025-10-27 06:49:28 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL
2025-10-27 06:49:28 library versions: OpenSSL 3.5.1 1 Jul 2025, LZO 2.10
2025-10-27 06:49:28 DCO version: N/A
2025-10-27 06:49:28 WARNING: you are using user/group/chroot/setcon without persist-tun -
2025-10-27 06:49:28 WARNING: you are using user/group/chroot/setcon without persist-key -
2025-10-27 06:49:28 WARNING: --keepalive option is missing from server config
2025-10-27 06:49:28 net_route_v4_best_gw query: dst 0.0.0.0
2025-10-27 06:49:28 net_route_v4_best_gw result: via 192.168.15.1 dev enp0s3
2025-10-27 06:49:28 TUN/TAP device tun0 opened
2025-10-27 06:49:28 net_iface_mtu_set: mtu 1500 for tun0
2025-10-27 06:49:28 net_iface_up: set tun0 up
2025-10-27 06:49:28 net_addr_v4_add: 10.10.10.1/24 dev tun0
2025-10-27 06:49:28 Could not determine IPv4/IPv6 protocol. Using AF_INET
2025-10-27 06:49:28 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-10-27 06:49:28 UDPv4 link local (bound): [AF_INET][undef]:1194
2025-10-27 06:49:28 UDPv4 link remote: [AF_UNSPEC]
2025-10-27 06:49:28 UID set to nobody
2025-10-27 06:49:28 GID set to nogroup
2025-10-27 06:49:28 Capabilities retained: CAP_NET_ADMIN
2025-10-27 06:49:28 MULTI: multi_init called, r=256 v=256
2025-10-27 06:49:28 IFCONFIG POOL IPv4: base=10.10.10.2 size=253
2025-10-27 06:49:28 Initialization Sequence Completed
```

Fonte: autores (2025).

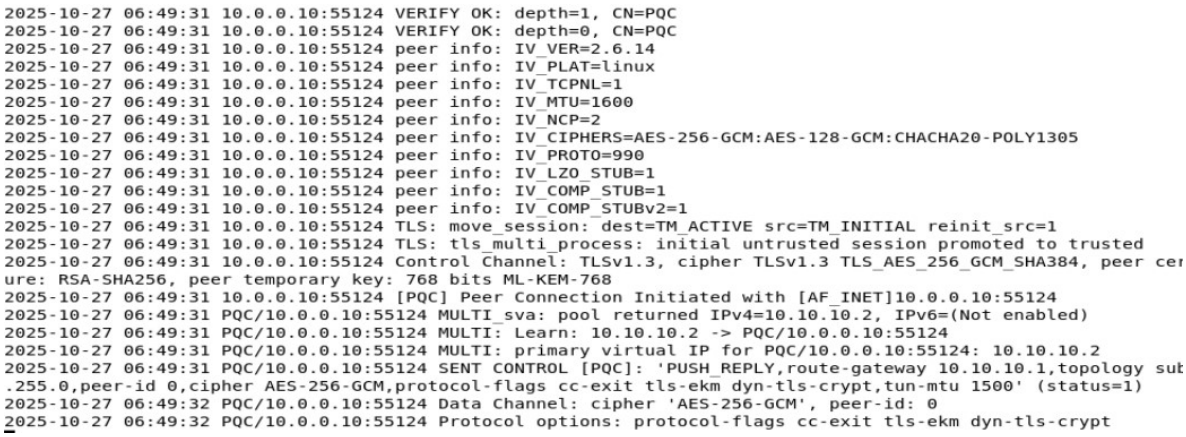
Na VM *cliente,* foi utilizado o comando *sudo openvpn –config /opt/openvpn/client/cliente.ovpn* para realizar o acesso à VPN, devendo os comandos serem executados com o privilégio de super usuário. A figura 9 detalha essa conexão.

Figura 9: conexão da VM cliente à VPN.

Fonte: autores (2025).

Na saída da conexão, é possível observar a atribuição do número de endereço IP 10.10.10.2 à interface dev tun0. A figura 10 apresenta o resultado obtido durante o recebimento da conexão na VM *servidor.*

Figura 10: conexão recebida da VM cliente.

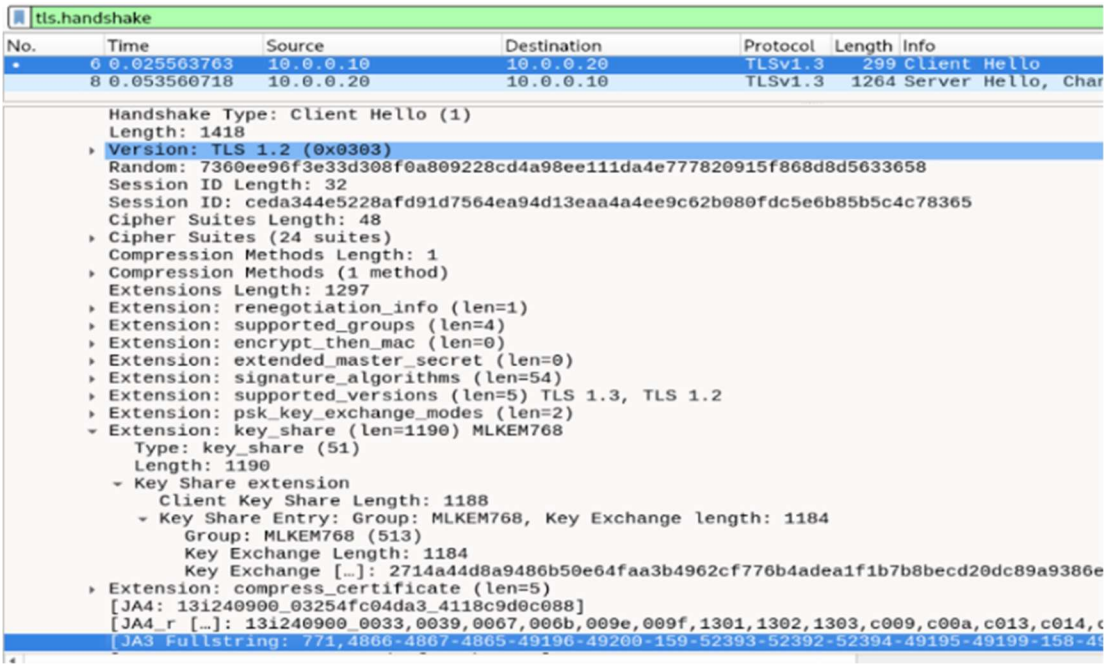


Fonte: autores (2025).

Pode-se observar a utilização do algoritmo para troca de chaves MLKEM768 como um *peer* temporário (um dispositivo temporário) de *handshake.* Além disso, ao capturar os pacotes durante a conexão entre o *cliente* e *servidor* por meio do *Wireshark*, é possível validar o uso do *MLKEM768* como algoritmo de troca de chaves

no pacote nº 21, filtrando com *tls.handshake* para uma melhor verificação, demonstrada na figura 11.

Figura 11: análise do MLKEM768 no *handshake* por meio do Wireshark.



Fonte: autores (2025).

Na figura 12, pode-se observar que a VM *cliente* obteve um endereço IP 10.10.10.2/24 a partir da VM *servidor,* conforme configurado no arquivo *servidor.conf* da VPN onde há o endereço e máscara de rede definidos.

Figura 12: endereço IP da VPN na interface tun0 da VM cliente.

```
8: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNK
    link/none
    inet 10.10.10.2/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::318a:2db8:1fc:9772/64 scope link stable-privacy proto kernel_ll
       valid lft forever preferred lft forever
```

Fonte: autores (2025).

A seguir serão demonstrados todos os resultados obtidos utilizando-se a ferramenta *Wireshark* para realizar a captura dos pacotes frutos da conexão entre a máquina *cliente* e a máquina *servidor*. Foram realizadas 10 conexões entre as partes e a partir desta conexão foi possível obter o tempo de *handshake* para cada uma dessas conexões e através da ferramenta, capturar quanto tempo durou o início de cada conexão.

As tabelas 1 e 2 apresentam os dados obtidos a partir da captura dos pacotes de conexões utilizando os algoritmos de criptografia com 128 bits de segurança. Na tabela 1, o tempo médio de *handshake* do algoritmo *MLKEM512* nas 10 conexões realizadas é de aproximadamente 0,003554765 segundos, ou 3.55 milissegundos.

Tabela 1: número de conexões e tempo de *handshake* do MLKEM512.

N°	TEMPO (S)
1	0.009098789
2	0.005184355
p	0.005636214
4	0.003427032
5	0.003158393
6	0.003437933
7	0.003840658
8	0.003440779
9	0.003668751
10	0.003271621

Fonte: autores (2025).

Na tabela 2, pode-se observar os resultados das 10 conexões realizados com o algoritmo *SECP256R1*, apresentando um tempo médio de *handshake* de aproximadamente 0,003685913 segundos, ou 3.69 milissegundos.

Tabela 2: número de conexões e tempo de *handshake* do SECP256R1.

N°	TEMPO (S)
1	0.013423807
2	0.003835979
3	0.003535847
4	0.004130604
5	0.003300961
6	0.003486394
7	0.003434595
8	0.003452189
9	0.004365444
10	0.004041505

Fonte: autores (2025).

As tabelas 3 e 4 mostram os dados obtidos a partir da captura dos pacotes de conexões dos algoritmos que possuem 192 bits de segurança. Na tabela 3, são apresentados os resultados das 10 conexões realizadas com o *MLKEM768*, possuindo uma média do tempo de *handshake* de aproximadamente 0,004072384 segundos, ou 4.07 milissegundos.

Tabela 3: número de conexões e tempo de *handshake* do MLKEM768.

N°	TEMPO (S)
1	0.003978134
2	0.003372915
3	0.004704294
4	0.005089535
5	0.004204113
6	0.003436765
7	0.003524534
8	0.004166634
9	0.004781570
10	0.003244123

Fonte: autores (2025).

Enquanto isso, a tabela 4 apresenta as conexões realizadas com o algoritmo *SECP384R1,* sendo a sua média do tempo de *handshake* de aproximadamente 0,005118486 segundos, ou 5.12 milissegundos.

Tabela 4: número de conexões e tempo de *handshake* do SECP384R1.

N°	TEMPO (S)
1	0.005092135
2	0.005821796
3	0.005425437
4	0.005144837
5	0.004987691
6	0.006456760
7	0.004874720
8	0.005432736
9	0.004378044
10	0.004735932

Fonte: autores (2025).

As tabelas 5 e 6 apresentam os dados obtidos da captura dos pacotes de conexão dos algoritmos que possuem 256 bits de segurança. Na tabela 5, o MLKEM1024 apresenta um tempo médio de 0.004776525 segundos, ou 4.78 milissegundos, ao se calcular a média das 10 conexões realizadas

Tabela 5: número de conexões e tempo de *handshake* do MLKEM1024.

N°	TEMPO (S)
1	0.004124556
2	0.005115569
3	0.005408094
4	0.003530047
5	0.007231007
6	0.003998707
7	0.003598106
8	0.005681435
9	0.011193378
10	0.004437480

Fonte: autores (2025).

O algoritmo *SECP521R1* apresenta um tempo médio de *handshake* de aproximadamente 0.007335374segundos, ou 7.34 milissegundos, calculado ao se obter a média entre as 10 conexões realizadas, conforme a tabela 6.

Tabela 6: número de conexões e tempo de *handshake* do SECP521R1.

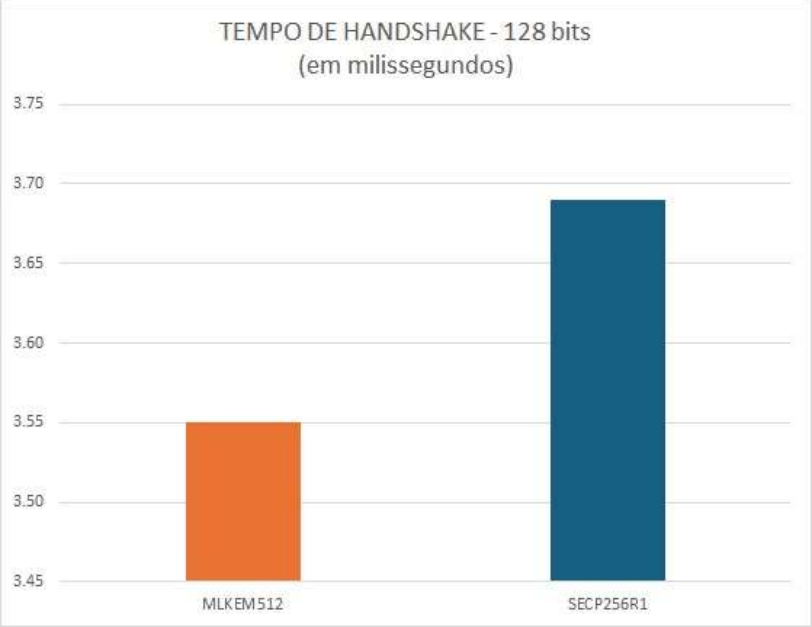
N°	TEMPO (S)
1	0.007445880
2	0.006928169
3	0.007660518
4	0.006844600
5	0.008867455
6	0.007698874
7	0.008761127
8	0.006993056
9	0.006813519
10	0.007224867

Fonte: autores (2025).

A seguir serão apresentadas figuras que contém gráficos demonstrando um comparativo entre a duração média do processo de *handshake* do modelo de

criptografia pós-quântico ML-KEM e do tradicional ECDHE em todos os níveis de segurança (128, 192 e 256 bits), o que possibilita uma análise visual mais detalhada das diferenças de desempenho. Na figura 13, onde se demonstra visualmente a diferença do tempo médio dos dados que estão presentes nas tabelas 1 e 2, pode-se analisar a comparação dos tempos de *handshake* dos algoritmos MLKEM512 e SECP256R1. Ambos possuem uma segurança equivalente a 128 bits. É possível observar que o algoritmo MLKEM512 teve um tempo médio inferior ao do protocolo SECP256R1, o que representa uma maior eficiência no estabelecimento da conexão por parte do algoritmo pós-quântico.

Figura 13: comparativo do tempo de handshake - 128 bits.



Fonte: autores (2025).

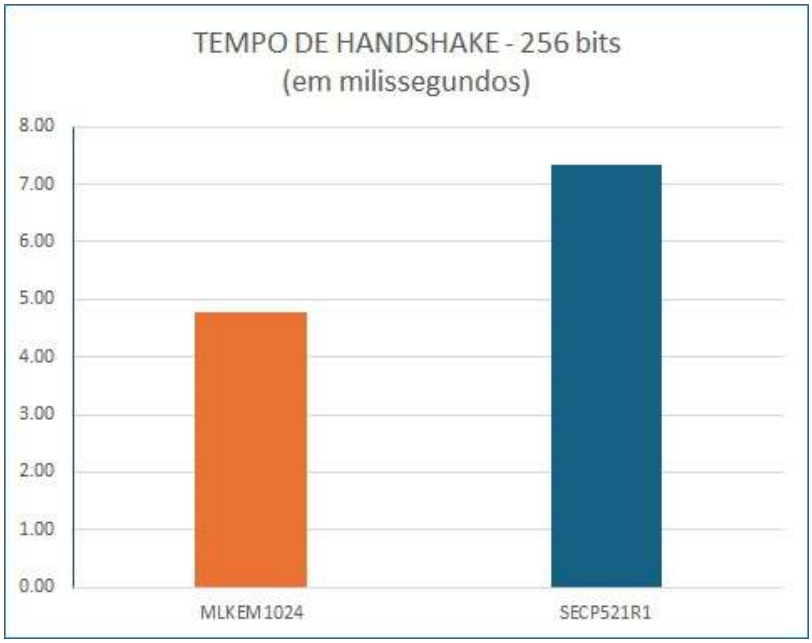
A figura 14 apresenta a comparação dos tempos médios de *handshake* do algoritmo MLKEM768 e do protocolo SECP384R1, utilizando-se os dados presentes nas tabelas 3 e 4. Ambos possuem uma segurança equivalente a 192 bits. É possível observar que apesar da pequena diferença, o algoritmo pós-quântico MLKEM768 mostrou-se mais eficiente no tempo de estabelecimento de conexão.

Figura 14: comparativo do tempo de handshake - 192 bits.

Fonte: Autores (2025).

A figura 15 apresenta a comparação dos tempos médios de *handshake* do algoritmo MLKEM1024 e do protocolo SECP521R1, apresentados nas tabelas 5 e 6. Ambos possuem uma segurança de 256 bits. É possível analisar, a partir dos dados obtidos, que o algoritmo pós-quântico mostrou mais eficiência no estabelecimento da conexão.

Figura 15: comparativo do tempo de handshake - 256 bits.



Fonte: Autores (2025).

Com isso, ao se analisar comparativamente os resultados entre os algoritmos e as suas equivalências de segurança em relação ao AES128, AES192 e o AES256, conclui-se que todas as variantes do ML-KEM apresentaram um bom desempenho no processo de *handshake*, superior em relação ao ECDHE – representado pelas curvas elípticas SECP256R1, SECP384R1 e SECP521R1 -, o que evidencia um resultado positivo em um cenário de migração dos algoritmos de criptografia para troca de chaves clássicos para os pós-quânticos.

Além disso, a análise possibilita inferir que, conforme o nível de segurança do ML-KEM aumenta, é mantido uma escalabilidade mais estável no tempo de *handshake* – o MLKEM768 é cerca de 14,7% mais lento que o MLKEM512, enquanto o MLKEM1024 é aproximadamente 17,45% mais lento que o MLKEM768 e 34,65% mais lento que MLKEM512. No algoritmo clássico, o tamanho do nível de segurança impacta de forma mais acentuada – isto é, o SECP384R1 é cerca de 38,8% mais lento que o SECP256R1, enquanto o SECP521R1 é aproximadamente 43,36% mais lento que o SECP384R1 e 98,65% mais lento que o SECP256R1. Dessa forma, os dados indicam uma melhor relação entre desempenho e segurança por parte do algoritmo baseado em reticulados.

5 CONCLUSÃO

A partir da análise dos dados coletados durante o estudo, pode-se entender que o algoritmo pós-quântico ML-KEM em todas as suas variantes, apresenta vantagem na variável “tempo médio” para a realização de uma conexão (*handshake*). O algoritmo ML-KEM realizou sua conexão mais rapidamente que seus concorrentes SECP256R1, SECP384R1 e SECP521R1, em todos os testes que foram realizados: no primeiro teste, representado através da figura 13, entre ML-KEM512 e o protocolo SECP256R1 percebe-se que o algoritmo pós-quântico foi mais eficiente no tempo de *handshake*. O algoritmo teve como tempo médio calculado (das dez conexões) o valor de 3,55 milissegundos, enquanto sua contraparte teve um valor médio de 3,69 milissegundos. Assim o ML-KEM é cerca de 3,8% mais rápido que o seu concorrente.

No segundo teste, onde foram colocados à prova o ML-KEM768 e o SECP384R1, demonstrado na figura 14, apresentou-se um resultado com um pouco mais de disparidade, no qual o ML-KEM768 continuou sendo mais rápido, tendo como média um valor de 4,07 milissegundos enquanto o outro protocolo teve um tempo médio de 5,12 milissegundos, apresentando, portanto, uma diferença de aproximadamente 21% em relação ao tempo médio calculado do seu concorrente.

Apresentou-se na figura 15, o terceiro e último teste, a maior disparidade em relação ao tempo de *handshake* em que se compararam as variantes ML-KEM1024 e SECP521R1. O algoritmo ML-KEM, foi consistentemente superior em desempenho, apresentando como tempo médio no processo de *handshake* 4,78 milissegundos de duração, enquanto o protocolo obteve 7,34 milissegundos em média. Assim, nessa análise, o ML-KEM mostra-se aproximadamente 35% mais eficiente no estabelecimento de uma conexão**.** Dessa forma, pode-se concluir que o algoritmo ML-KEM é superior que o seu concorrente em todos os testes realizados e, desse modo, evidenciando uma vantagem em relação a esses algoritmos e protocolos mais comuns hoje em dia.

No entanto, embora a questão da eficiência seja um ponto de extrema importância, outro ponto que deve ser considerado além da questão de performance e talvez o maior motivador para uma futura etapa de implementação em massa das tecnologias pós-quânticas no cotidiano, é a questão da segurança que esses novos

mecanismos proporcionem frente as ameaças da expansão da computação quântica. Para um possível uso comum, as máquinas quânticas tem um poder computacional que supera o que se tem como tecnologia atualmente e isso pode acarretar em problemas para a proteção dos dados. Tomando como exemplo o algoritmo de Shor, que consegue resolver problemas matemáticos complexos que são utilizados como base para a segurança da informação e, sendo assim, a implementação de mecanismos que consigam suportar ataques dessa natureza são de extrema importância para que o uso de ambientes como a internet possam continuar a ser viáveis e seguros.

Assim este estudo cumpre com a função de explicitar parte das vantagens que esses algoritmos pós-quânticos trazem em relação as tecnologias que utilizamos atualmente e, assim, proporciona uma visão dos benefícios que estas novas tecnologias têm a oferecer para um futuro próximo, onde o novo paradigma da Computação Quântica tenha se consolidado em massa na vida das pessoas.

REFERÊNCIAS

ABDULRAHMAN, A.; OBERHANSL, F. PHAM, H. N. H.; PHILIPOOM, J.; SCHWABE, P.; STELZER, T.; ZANKL, A. ***Towards ML-KEM & ML-DSA on OpenTitan***. 2025 IEEE Symposium on Security and Privacy, p. 1-26, 16 jun. 2025. DOI 10.1109/SP61157.2025.00220. Disponível em: https://eprint.iacr.org/2024/1192.pdf. Acesso em: 17 ago. 2025.

AMORIM, P. R.; HENRIQUES, M. A. A. **Avaliação de algoritmos de criptografia pós-quânticos empregados em ambientes restritos de computação**. *In:* XVIII Congresso {virtual} de Iniciação Científica da Unicamp, 18, 2020. Disponível em: https://prp.unicamp.br/inscricao-congresso/resumos/2020P17444A35242O3517.pdf. Acesso em: 14 ago. 2025.

AZARDERAKHSH, R.; ELKHATIB, R.; KOZIEL, Br.; LANGENBERG, B. ***Hardware Deployment of Hybrid PQC***. *Security and Privacy in Communication Networks***,** [*s. l.*], 4 nov. 2021. DOI https://doi.org/10.1007/978-3-030-90022-9_26. Disponível em: https://eprint.iacr.org/2021/541.pdf. Acesso em: 5 jul. 2025.

POOVENDRAN, R.; DEBAR, H.; YUNG, M*. Security and privacy in communication networks. SecureComm 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, v. 399. Cham: Springer, 2021. DOI https://doi.org/10.1007/978-3-030-90022-9_26. Disponível em: https://eprint.iacr.org/2021/541.pdf. Acesso em: 7 set. 2025.

CAMPOS, B. A. R.; ROSA, E. C. **A evolução da computação quântica: impacto na criptografia RSA e a segurança nacional**. 2024. 19 p. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) - Instituto Federal Goiano, Catalão, 2024. Disponível em: https://repositorio.ifgoiano.edu.br/bitstream/prefix/4631. Acesso em: 27 maio 2025.

DYLAN, H.; CODY, G.; XIAOYUAN, L.; ALEXEY, G.; YUE, S.; ILYA, S.; MARCO, P.; YURI, A. ***Quantum computing for finance.*** *Nature Reviews Physics*, [*S. l.*], v. 5, n.8, p. 450-465, 2023. DOI: https://doi.org/10.1038/s42254-023-00603-1. Disponível em: https://www.nature.com/articles/s42254-023-00603-1. Acesso em: 5 set. 2025.

ERNESTO, J. A. P. D. M. **Comparação entre RSA e ECC no âmbito da certificação digital**. 2023. 49 p. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Computação) - Pontifícia Universidade Católica de Goiás, Goiânia, 2023. Disponível em: https://repositorio.pucgoias.edu.br/jspui/handle/123456789/6309. Acesso em: 10 jun. 2025.

EUNMI, C.; JOONHEE, C.; JUNKI, K. ***An elementary review on basic principles and developments of qubits for quantum computing***. *Nano Convergence*, [*S. l.*], v. 11, p. 1-13, 2024. DOI https://doi.org/10.1186/s40580-024-00418-5. Disponível em: https://link.springer.com/article/10.1186/s40580-024-00418-5. Acesso em: 7 set. 2025.

FERRO, L. F. C.; RAMPAZZO, F. J. A.; HENRIQUES, M. A. A. **Estudos de otimização do algoritmo de criptografia pós-quântica CRYSTALS-KYBER**. *In*: Workshop de trabalhos de iniciação científica e de graduação - simpósio brasileiro de segurança da informação e de sistemas computacionais (SBSEG), 21., 2021, Evento Online. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 205-218. DOI: <https://doi.org/10.5753/sbseg_estendido.2021.17353>. Acesso em: 2 jun. 2025.

FIRDAOUSS, L.; AYOUB, B.; MANAL B.; IKRAME, Y. ***Automated VPN configuration using DevOps***. *Procedia Computer Science*, [*S. l.*], v. 198, p. 632-637, 2022. DOI https://doi.org/10.1016/j.procs.2021.12.298. Disponível em: https://www.sciencedirect.com/science/article/pii/S1877050921025370?via%3Dihub. Acesso em: 13 ago. 2025.

HALL, M. J. ***Performance analysis of OpenVPN on a consumer grade router***. [*S. l.*], 11 p., 2025. DOI https://doi.org/10.48550/arXiv.2504.19069. Disponível em: https://arxiv.org/abs/2504.19069. Acesso em: 13 ago. 2025.

KARATSIOLIS, E.; KIEFER, F.; KRÄME, J.; MIRJAM, L.; TOBIAS, C.; WEISHÄUPL, M. ***Public key linting for ML-KEM and ML-DSA***. *Cryptology ePrint Archive*, [*S. l*], 25 p., 2025. Disponível em: https://eprint.iacr.org/2025/1241. Acesso em: 17 ago. 2025.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a Internet**: uma abordagem *top-down*. 8. ed. Tradução: Francisco Araújo da Costa. [São Paulo]: Pearson; Porto Alegre: Bookman, 2021, 584 p. ISBN 9788582605592.

NICOLETI, G. A.; OLIVEIRA, R. L. S. ***Virtual Private Networks VPN****:* sua importância no contexto empresarial. 2023. Trabalho de Graduação (Tecnologia em Sistemas para Internet) – Faculdade de Tecnologia Prof. José Camargo, Jales, 2023. Disponível em: <https://ric.cps.sp.gov.br/handle/123456789/16338>. Acesso em: 05 jun. 2025.

RICCHIZZI, N.; SCHWINNE, C.; PELZL, J. ***Applied Post Quantum Cryptography****: A practical approach for generating certificates in industrial environments*, 2025. Disponível em: https://arxiv.org/pdf/2505.04333. Acesso em: 28 set. 2025.

SILVA, E. G.; RIBEIRO, L. R.; EMBOAVA, S. G. R. **Análise comparativa entre o crivo quadrático e algoritmo quântico de Shor aplicado à criptografia**, 2024. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Segurança da Informação) - Faculdade de Tecnologia de Americana “Ministro Ralph Biasi”, Americana, 2024. Disponível em: https://ric.cps.sp.gov.br/handle/123456789/21779. Acesso em: 5 nov. 2025.

WIRESHARK. W*ireshark User's Guide*. v. 4.5.0, 2024. Disponível em: https://www.wireshark.org/docs/wsug_html_chunked/. Acesso em: 17 ago. 2025.

ZHOU, J.; FU, W.i; HU, W.; SUN, Z.; HE, T.; ZHANG, Z. ***Challenges and Advances in Analyzing TLS 1.3-Encrypted Traffic:*** *a comprehensive survey*. *Electronics*, [*s. l.*], v. 13, n. 20, 31 p., 2024. DOI https://doi.org/10.3390/electronics13204000.

Disponível em: https://www.mdpi.com/2079-9292/13/20/4000. Acesso em: 11 nov. 2025.