UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Alexandre Augusto Giron

Hybrid Post-Quantum Cryptography in Network Protocols

Florianópolis
2023

Alexandre Augusto Giron

Hybrid Post-Quantum Cryptography in Network Protocols

Tese submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de doutor em Ciência da Computação.
Orientador: Prof. Ricardo Felipe Custódio, Dr.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Giron, Alexandre Augusto
Hybrid Post-Quantum Cryptography in Network Protocols / Alexandre Augusto Giron ; orientador, Ricardo Felipe Custódio, 2023.
103 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciência da Computação. 2. Criptografia Pós-Quântica Híbrida. 3. Segurança de Redes. 4. Transport Layer Security (TLS). 5. Automatic Certificate Management Environment (ACME). I. Custódio, Ricardo Felipe. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Alexandre Augusto Giron

Hybrid Post-Quantum Cryptography in Network Protocols

O presente trabalho em nível de doutorado foi avaliado e aprovado, em 11 de Agosto 2023, por banca examinadora composta pelos seguintes membros:

Prof. Jean Everson Martina, Dr.
Universidade Federal de Santa Catarina

Prof. Marco Aurélio Amaral Henriques, Dr.
Universidade Estadual de Campinas

Prof. Routo Terada, Dr.
Universidade de São Paulo

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de doutor em Ciência da Computação.

GOV.BR
Documento assinado digitalmente
PATRICIA DELLA MEA PLENTZ
Data: 16/10/2023 09:25:54-0300
CPF: ***.243.700-**
Verifique as assinaturas em https://v.ufsc.br

Profa. Patricia Della Méa Plentz, Dra
Coordenadora do Programa

ICP-Edu
Documento assinado digitalmente
Ricardo Felipe Custodio
Data: 16/10/2023 08:08:16-0300
CPF: ***.981.609-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Ricardo Felipe Custódio, Dr.
Orientador

Florianópolis, 2023.

Para a mamãe, vovós e vovôs da Layla.

AGRADECIMENTOS

Eu sempre tive muitos motivos a agradecer e, durante este doutoramento, não foi diferente. Em primeiro lugar, agradeço a Deus e à minha família, esta que cresceu com a chegada da Layla. Sou muito grato, em especial, à minha esposa Francielle, por ser meu apoio, direção e força para seguir em frente. Sempre tive o apoio da minha família, pais e irmão (agora avós e tio!), portanto, deixo meus sinceros agradecimentos.

Devo agradecimentos especiais ao meu orientador, prof. Ricardo Custódio. Sua orientação – que percebi como motivadora ao crescimento pessoal e profissional – será certamente carregada adiante. Sou muito grato.

Agradeço também aos colegas do LabSEC, Lucas Perin, Frederico Schardong, João Pedro A. do Nascimento, Douglas Martins, Gustavo Zambonin, entre outros. O LabSEC me ajudou de diversas maneiras durante o doutorado. Incluo também um agradecimento especial ao prof. Jean Everson Martina, do LabSEC. Seu apoio e colaboração foram muito importantes. Meu muito obrigado.

Agradeço o apoio e receptividade do pessoal do TII (*Technology Innovation Institute*) dos Emirados Árabes Unidos: Lucas Perin, Victor Mateu, Francisco Rodríguez-Henríquez e aos demais membros do *Cryptography Research Center*. Adquiri muito aprendizado durante a colaboração no projeto de pesquisa. Muito obrigado.

O presente trabalho foi realizado com apoio substancial da Universidade Tecnológica Federal do Paraná (UTFPR). Portanto, deixo os meus sinceros agradecimentos pela oportunidade.

I appreciate the support and receptiveness of the people at the Technology Innovation Institute (TII) in the United Arab Emirates: Lucas Perin, Victor Mateu, Francisco Rodríguez-Henríquez, and the other members of the Cryptography Research Center. I acquired a lot of knowledge during the collaboration on the research project. Thank you very much.

This work was carried out with substantial support from the Federal University of Technology - Paraná (UTFPR). Therefore, I express my sincere thanks for the opportunity.

Trust is a judgement of unquestionable utility
(MARSH, 1994)
Nature is often unpredictable but it is rarely random
(BROWN, 2019)
Take learning in your hands, do not let her go: keep her, for she is your life.
(Proverbs, 4:13)

RESUMO

A segurança de redes é essencial para as comunicações do dia-a-dia. Protocolos como o *Transport Layer Security* (TLS) e o *Automatic Certificate Management Environment* (ACME) possibilitam comunicações seguras para várias aplicações. O TLS fornece canais seguros com autenticação de pares comunicantes, desde que estes pares já tenham um certificado digital para provar sua identidade. Já o protocolo ACME contribui com a adoção de TLS com funcionalidades para emissão e gerenciamento de certificados digitais. Tanto o TLS quanto o ACME depende da Criptografia de Chaves Públicas para autenticação e troca de chaves (*Key Exchange* - KEX). No entanto, o advento do Computador Quântico Criptograficamente Relevante (CQCR) enfraquece os protocolos de KEX e certificados digitais criados com a criptografia clássica usada atualmente, tais como RSA e Diffie-Hellman. Dada a grande adoção do TLS e ACME, esta ameaça alcança uma escala global. Neste contexto, esta tese trata dos desafios da adoção de Criptografia Pós-Quântica (CPQ) no TLS e ACME, focando-se na abordagem recomendada chamada de CPQ híbrida (ou modo híbrido). A CPQ é criada usando suposições matemáticas diferentes das em uso atualmente. Estas suposições são viáveis para construção de esquemas criptográficos resistentes ao computador quântico, pois não se conhece algoritmo (clássico ou quântico) eficiente. Porém, a transição para CPQ é assunto complexo. No modo híbrido, a transição para CPQ é suavizada, pois ela é combinada com a criptografia tradicional. Assim, esta tese defende a estratégia de adoção de CPQ pelo modo híbrido com as seguintes contribuições: um estudo secundário classificando e mostrando a eficiência e segurança do modo híbrido; uma ferramenta para verificar as garantias *quantum-safe* em conexões TLS de usuários; um estudo e uma otimização para a emissão de certificados digitais com CPQ no ACME; o projeto e implementação de uma abordagem híbrida para a alternativa de TLS chamada KEMTLS; e um conceito híbrido inovador, com implementação, para autenticação usando certificados *wrapped*. Na maioria dos cenários de avaliações com modo híbrido propostos neste trabalho, as penalidades de desempenho não são significativas quando comparadas com a implantação de CPQ sem o modo híbrido. O conceito inovador da autenticação híbrida também habilita um plano de contingência para o modo híbrido, contribuindo com a adoção de CPQ. Por meio das propostas e avaliações em diferentes cenários, abordagens e protocolos, esta tese soma esforços em direção ao uso de CPQ híbrida para mitigar os efeitos preocupantes da ameaça quântica à criptografia.

Palavras-chave: Criptografia Pós-Quântica Híbrida. Segurança de redes. TLS. ACME.

RESUMO ESTENDIDO

Introdução

A Criptografia de Chaves Públicas é empregada em larga escala, tanto em sistemas internos de organizações, quanto na Internet de forma global. Por meio de Infraestruturas de Chaves Públicas (ICPs), os protocolos de segurança em rede habilitam uma série de aplicações para os usuários, desde navegação na *web* até transações financeiras. Sem o uso de ICPs, estas aplicações não existiriam ou se tornariam inviáveis de serem utilizadas pela Internet atualmente. Na prática, os casos de uso mais comuns da criptografia de chaves públicas é na autenticação e no provimento de chaves de criptografia simétrica (*Key Exchange* - KEX).

Apesar disso, os esquemas de troca de chaves e de assinaturas digitais usados hoje são considerados inseguros com o advento do Computador Quântico Criptograficamente Relevante (CQCR). Esquemas de chaves públicas amplamente utilizados como o RSA e Diffie-Hellman são vulneráveis ao algoritmo de Shor. Dessa forma, quando a evolução do computador quântico atingir o patamar necessário para efetivamente quebrar os esquemas atuais, os sistemas e aplicações terão propriedades de sigilo e autenticidade passíveis de violação. Em consequência, os esquemas de chaves públicas atuais não poderão mais fazer parte de sistemas do futuro.

A urgência por soluções viáveis a este problema é aumentada sob a face do ataque conhecido como *record-now-decrypt-later*. Neste cenário, os dados de usuário que trafegam pela Internet são suscetíveis à interceptação e coleta, porém a confidencialidade é mantida devido à criptografia. No entanto, o ataque consiste em coletar os dados trafegando hoje com a expectativa de decifrar no futuro, quando o computador quântico estiver disponível e com a capacidade esperada para quebrar a criptografia atual. Isso implica que as conexões na Internet atual, majoritariamente feitas por meio do protocolo *Transport Layer Security* (TLS), não têm mais as garantias de confidencialidade que os esquemas criptográficos têm fornecido. Em outras palavras, havendo um computador quântico com poder suficiente, o conteúdo das comunicações de hoje poderá ser revelado no futuro.

Para solucionar esse problema, especialistas estudam o uso da Criptografia Pós-Quântica (CPQ), feita com base em problemas matemáticos para os quais não se conhece solução eficiente, nem por computadores quânticos, nem por computadores tradicionais. O cenário se baseia em usuários de computadores tradicionais almejando se proteger de atacantes com capacidades da computação quântica. Portanto, os especialistas sugerem uma migração global da criptografia tradicional, em uso hoje, para a CPQ. Porém, essa migração possui uma série de dificuldades. Por exemplo, uma vez que a confiança nos esquemas ditos pós-quânticos ainda não está estabelecida, uma forma de adoção prática sendo recomendada é por meio do "modo híbrido". Nesse modo, existem decisões de projeto a serem tomadas e ele requer análises do impacto no desempenho das aplicações e protocolos de rede. Considerando um algoritmo de assinatura digital para autenticação, o modo híbrido prevê uma assinatura com um algoritmo clássico, já conhecido e estabelecido, e uma segunda assinatura com um algoritmo pós-quântico. No caso dos mecanismos de troca de chaves simétricas, os dois algoritmos executam concomitantemente, e o resultado deles é concatenado para posterior derivação do material de chaves simétricas. Esse uso combinado no modo híbrido garante que a segurança deve se manter enquanto um dos algoritmos continua seguro (mesmo que um deles seja vulnerável ao computador quântico).

Independentemente do modo de adoção da CPQ (híbrido ou não), um problema relacionado a seu uso é em relação ao tamanho em *bytes* dos objetos criptográficos. Assim, a adoção destes novos esquemas pode impactar o desempenho de protocolos de comunicação em rede. Por isso, é fundamental avaliar a transição destes protocolos para se conhecer, de forma antecipada, os requisitos e desempenho esperado com CPQ. Ressalta-se que essa transição é extremamente

complexa, considerando-se os diversos protocolos de rede existentes na Internet. De modo usual, os usuários recorrem ao HTTPS para segurança de suas comunicações na Internet, porém, para permitir conexões seguras com HTTPS, outros protocolos são usados, tais como o TLS e o ACME (*Automatic Certificate Management Protocol*). Enquanto o TLS fornece canais seguros para os usuários, o ACME facilita a configuração de artefatos da infraestrutura de segurança necessária para comunicação com TLS. Tais protocolos são fundamentais para a Internet Segura tal qual é utilizada atualmente.

Objetivos

Os objetivos desta tese convergem em torno da melhoria da adoção da CPQ híbrida em protocolos de rede. Para isso, realiza-se propostas, adaptações, implementações e avaliações de desempenho simulando cenários e planos de transição para criptografia pós-quântica com e sem o modo híbrido. Neste trabalho, os objetivos específicos incluem: a identificação dos desafios da adoção do modo híbrido em protocolos de rede; a realização de propostas e experimentos para mitigação dos desafios identificados previamente, considerando diferentes cenários, tais como ambientes simulados e realísticos; e contribuir com a conscientização das possíveis ameaças da computação quântica sobre os protocolos existentes, dos desafios de adoção de CPQ e das possíveis soluções a estes desafios.

Metodologia

Os algoritmos de criptografia pós-quântica a serem utilizados neste trabalho são os disponibilizados pelo processo de padronização de CPQ do NIST. Tal processo de padronização é um dos esforços mais conhecidos na área, agregando pesquisadores de várias partes do mundo. Dentre os algoritmos, o Dilithium (como escolha primária), Falcon e SPHINCS+ (como escolhas alternativas) foram selecionados pelo NIST para padronização, e assim fazem parte do foco deste trabalho. Adicionalmente, o algoritmo Kyber também foi selecionado para padronização, mas instanciado como um *Key-Encapsulation Mechanism* (KEM). KEMs são normalmente utilizados para a geração e transporte de material de chaves simétricas. No entanto, é importante destacar que o processo de padronização do NIST ainda não foi concluído. Isso é só um dos indicadores de que a adoção de CPQ necessita de pesquisa para uma transição com menos impacto às aplicações.

Em geral, os esforços da literatura sobre a adoção de CPQ têm se concentrado em duas grandes frentes: trabalhos de *benchmark*, onde protocolos e aplicações são avaliados com (e sem) criptografia pós-quântica e trabalhos onde os protocolos são modificados para melhor acomodar a CPQ. Por exemplo, a proposta do KEMTLS modifica o protocolo TLS para uso de KEMs na autenticação, sob o argumento de que KEMs têm menores tamanhos do que as assinaturas pós-quânticas. Sendo o objetivo deste trabalho propor e avaliar o modo híbrido em protocolos de rede, a metodologia desta tese busca oferecer: (i) um rigoroso estudo secundário para classificar e facilitar o entendimento sobre o modo híbrido na troca de chaves simétricas; (ii) uma solução para usuários checarem o uso e a presença de algoritmos de CPQ em conexões TLS 1.3; (iii) fornecer implementações (e avaliações) do modo híbrido, tanto para conexões TLS quanto para a emissão de certificados digitais com o ACME, especialmente onde há carência de avaliações e implementações; e (iv) melhorar as estratégias de transição, por meio de modificações e trazendo uma propriedade inovadora de autenticação pós-quântica híbrida, a qual é projetada para resolver a falta de contingência do modo híbrido (quando é descoberta uma vulnerabilidade no algoritmo CPQ utilizado).

Resultados e Discussão

Os primeiros resultados de contribuição estão relacionados a um Mapeamento Sistemático da Literatura sobre o modo híbrido em mecanismos de KEX. O mapeamento classifica as abordagens híbridas e identifica problemas de pesquisa em aberto, facilitando o entendimento e estudo da área por novos pesquisadores, projetistas e desenvolvedores. Na sequência, deste trabalho resulta uma ferramenta chamada *TLS 1.3 Handshake Analyzer*. Esta ferramenta permite a identificação de algoritmos de CPQ em conexões TLS 1.3 e características de desempenho. Usuários podem verificar o tipo de segurança presente em suas conexões.

Após a identificação dos problemas em aberto, verificou-se que o protocolo ACME carece de avaliações no cenário pós-quântico. Assim, o ACME foi modificado e avaliado na sua emissão e renovação de certificados com CPQ, com objetivo de conhecer os impactos da sua adoção. No lado dos servidores ACME, os impactos foram percebidos com o aumento do consumo de recursos de máquina e diminuição das requisições atendidas por segundo. No lado do cliente, os impactos são menos perceptíveis pela natureza do protocolo. De todo modo, foi proposta uma forma de agilizar a emissão de certificados para o cenário de transição para CPQ. Essa forma modifica o protocolo ACME por meio de um "challenge" alternativo chamado de *PQ-Transition Challenge*. Com ele, foi possível agilizar, em média, 4.22x no tempo de emissão de certificados e diminuir até 35% dos *bytes* transferidos na comunicação. Além disso, observou-se, mais uma vez, que as penalidades do modo híbrido foram baixas, especialmente do lado do cliente ACME. Acredita-se que o ACME será um facilitador da transição para CPQ, uma vez que ele já contribui com a adoção do protocolo TLS na Internet.

Nesta tese, uma implementação e avaliação foi desenvolvida para trazer o modo híbrido para o KEMTLS. O KEMTLS é uma abordagem que modifica o TLS com objetivo de autenticar handshakes com mecanismos de KEM pós-quânticos ao invés de assinaturas digitais. A avaliação do KEMTLS híbrido foi extensa, com experimentos em redes simuladas e "realísticas", incluindo-se diversos algoritmos pós-quânticos e seus níveis de segurança. Observou-se que o modo híbrido não impõe penalidades significativas no desempenho do KEMTLS, o que estimula o seu uso na prática pois mantém a confiabilidade da criptografia tradicional. Observou-se também a superioridade do KEMTLS híbrido contra TLS com assinaturas pós-quânticas híbridas quando os parâmetros de segurança foram configurados em um nível (nível 3 do NIST), e notou-se o aumento da penalidade do híbrido nos níveis mais altos (nível 5 do NIST). Com ênfase em desempenho, portanto, sugerem-se os níveis mais baixos.

Com o estudo do modo híbrido, observou-se uma carência de um plano de contingência caso seja encontrada uma vulnerabilidade no algoritmo pós-quântico. Assim, foi proposta a abordagem chamada PKIELP (*PKI Extended Lifetime Period*), baseada em um conceito inovador de autenticação pós-quântica híbrida. Essa autenticação é feita por meio de certificados *wrapped*, nos quais a chave pública é cifrada, fornecendo contingência para o modo híbrido. Essa proposta permitiu diminuir o tamanho em *bytes* da autenticação pós-quântica, em comparação com algoritmos selecionados pelo NIST. Com tamanhos menores, o desempenho de conexões TLS no contexto da PKIELP foi melhorado.

Outras Contribuições

Além da adoção de criptografia pós-quântica, esta tese também contribuiu com outros temas de pesquisa. A natureza interdisciplinar presente neste trabalho é importante para permitir interação com outros pesquisadores, bem como entender e auxiliar a resolver os desafios e oportunidades de pesquisa em outros temas relacionados à segurança em computação. Dessa forma, a visão dos problemas em segurança computacional se torna mais transversal. Assim, as contribuições adicionais se dividem em duas partes. A primeira sumariza as colaborações (com outros pesquisadores), tanto na adoção de criptografia pós-quântica em protocolos partícipes do

gerenciamento de identidades digitais (como o *OpenID Connect*), quanto na análise de ataques de substituição em algoritmos pós-quânticos. A segunda parte mostra as contribuições adicionais desta tese, com ênfase em propor e avaliar, na prática, um mecanismo de detecção do uso não-autorizado de esteganografia em *blockchains*.

Considerações Finais

A migração da criptografia tradicional para a CPQ é a solução esperada para proteção contra a ameaça do computador quântico. O modo híbrido é uma estratégia de migração que vem sendo recomendada, porém, ela carrega alguns dos desafios inerentes da criptografia pós-quântica. Neste contexto, esta tese tratou dos desafios da adoção da CPQ híbrida em várias perspectivas, focando-se nos protocolos de rede TLS e ACME. Investigaram-se diferentes propostas para facilitar a adoção da CPQ no modo híbrido. Essa investigação é importante, pois dessa forma é possível conhecer os impactos e contribuir com a conscientização de forma antecipada. Além disso, estudos como esse permitem entender outras formas de adoção da CPQ (híbrida ou não), sendo que os exemplos de modificação de protocolo, realizados nesta tese, obtiveram melhores indicadores de performance. Portanto, o estudo antecipado permite avaliar cenários de proteção e escolha da melhor abordagem antes do advento da computação quântica.

Ressalta-se que, apesar da tese recomendar o modo híbrido diante dos resultados obtidos, a migração para CPQ ainda possui desafios futuros. A experiência obtida com o estudo dos protocolos TLS e ACME mostrou que determinados cenários podem favorecer ataques quânticos mesmo após a migração para CPQ. Nesta tese, foi identificada uma ameaça na qual um *record-now-decrypt-later* reusa uma autorização de emissão de certificado no ACME. Generalizando-se para outros protocolos, isso significa que informações de longo prazo, se capturadas hoje e decifradas no futuro, podem permitir interações com o protocolo (ou vazamentos de informação sensível). Dessa forma, apenas substituir algoritmos na infraestrutura de segurança da aplicação pode não ser suficiente. Isso indica que a completa migração para a era pós-quântica requer uma análise completa do protocolo, incluindo-se uma análise de riscos de acordo com as informações sensíveis que são gerenciadas no protocolo ou aplicação. Esse tipo de análise foi deixado como possível trabalho futuro. Assim, as aplicações da Internet terão melhores condições para proteção completa contra a possível ameaça do computador quântico.

Palavras-chave: Criptografia Pós-Quântica Híbrida. Segurança de redes. TLS. ACME.

ABSTRACT

Network security is essential for today's communications. Protocols such as Transport Layer Security (TLS) and Automatic Certificate Management Environment (ACME) enable secure communications for various applications. TLS provides secure channels with peer authentication, given that the peer already has a digital certificate to prove its identity. ACME contributes to TLS adoption with facilities for issuing and managing digital certificates. Both protocols depend on Public-Key Cryptography for authentication and Key Exchange (KEX) of symmetric key material. However, the advent of a Cryptographically Relevant Quantum Computer (CRQC) weakens KEX and digital certificates built with today's classical cryptography (like RSA and Diffie-Hellman). Given the widespread adoption of TLS and ACME, such a threat reaches a global scale. In this context, this thesis aims at the challenges of adopting Post-Quantum Cryptography (PQC) in TLS and ACME, focusing on the recommended approach called Hybrid PQC (or hybrid mode). PQC is created using different mathematical assumptions in which there is no known efficient solution by classical and quantum computers. Hybrids ease the PQC transition by combining it with classical cryptography. This thesis defends the hybrid mode adoption by the following contributions: a secondary study classifying and showing hybrid mode efficiency and security; a tool for users checking their TLS connections for quantum-safe guarantees; a study and an optimized approach for issuance of PQC digital certificates in ACME; a design and implementation of a hybrid approach for the TLS alternative called KEMTLS; and a novel hybrid concept (and implementation) for authentication using wrapped digital certificates. In all proposed hybrid mode evaluations, the penalty in performance was non-significant when compared to PQC-only deployment, except in certain situations. The novel concept for hybrid authentication also allows a contingency plan for hybrids, contributing to the PQC adoption. By proposing and evaluating different scenarios, approaches and protocols, this thesis sums efforts towards using hybrid PQC to mitigate the worrisome effects of the quantum threat to cryptography.

Keywords: Hybrid Post-Quantum Cryptography. Network Security. TLS. ACME.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ACME	Automatic Certificate Management Environment.
ASA	Algorithm Substitution Attack.
CA	Certificate Authority.
CRQC	Cryptographically Relevant Quantum Computer.
DLP	Discrete Logarithm Problem.
DNS	Domain Name System.
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral.
ECDLP	Elliptic Curve Discrete Logarithm Problem.
ECDSA	Elliptic Curve Digital Signature Algorithm.
HKDF	HMAC-based Extract-and-Expand Key Derivation Function.
HMAC	Hash-based Message Authentication Code.
HTTPS	Hypertext Transfer Protocol Secure.
IETF	Internet Engineering Task Force.
IFP	Integer Factorization Problem.
JSON	JavaScript Object Notation.
KAS	Key Authorization String.
KEX	Key Exchange.
LSB	Least-Significant Bit.
NIST	National Institute of Standards and Technology.
OBU	On-Board-Unit.
OQS	Open Quantum Safe.
PKC	Public-Key Cryptography.
PKCS	Public Key Cryptography Standards.
PQ	Post-Quantum.
PQC	Post-Quantum Cryptography.
PSK	Pre-Shared Key.
RFC	Request for Comments.
RSA	Rivest-Shamir-Adleman Algorithm.
RSSI	Received Signal Strength Indicator.
RTT	Round Trip Time.
SDN	Software-Defined Networks.
SMS	Systematic Mapping Study.
SSH	Secure Shell.
TLS	Transport Layer Security.
URL	Uniform Resource Locator.
VN	Vehicular Network.
VPN	Virtual Private Network.

CONTENTS

1 INTRODUCTION

Security in network protocols is a ubiquitous need in today's Internet. Financial transactions, healthcare data services, and regular Internet browsing require secure communications. Implementing such secure communications rely on Public-Key Cryptography (PKC), mainly to provide authentication and employ a Key Exchange (KEX) protocol (also known as Key-Establishment Method) (PAAR; PELZL, 2009). Authentication means proving who is the origin of the information, and it applies to users, servers, or other entities in the network. A KEX, on the other hand, is often an ephemeral process that solves the key-distribution problem in symmetric encryption, i.e., sharing keys before communication. Transport Layer Security (TLS) (RESCORLA, 2018) and Secure Shell (SSH) (YLONEN; LONVICK, 2006) are well-known protocols that employ PKC for KEX and authentication.

TLS is the most popular security protocol available. Several sources indicate the widespread usage of TLS on the Internet (CHAN et al., 2018; PARACHA et al., 2021). TLS can be used in embedded devices, automobile electric charging systems, and microservices. TLS is also suggested for more recent network technologies like 5G and Software-Defined Networks (SDN) (CLANCY; MCGWIER; CHEN, 2019). Undoubtedly, Internet users will continue to rely on TLS in the future.

TLS has been used for many years, but the boom in popularity can be due to the Let's Encrypt initiative (BIRGE-LEE et al., 2021). Let's Encrypt issues free certificates for web servers to secure their communications. Before Let's Encrypt, TLS adoption faced difficulties often related to manually configuring TLS servers. For example, a TLS server requires one (or more) digital certificate(s) for authentication. Such a configuration is now much easier using the Automatic Certificate Management Environment (ACME) protocol (BARNES et al., 2019). ACME is backed by Let's Encrypt, providing means for automating the configuration of HTTPS/TLS servers. ACME can be viewed as a "TLS-enabler protocol", thus helping to build a secure Internet. In numbers, Let's Encrypt has issued over a billion digital certificates (BIRGE-LEE et al., 2021).

However, the mentioned protocols face a security threat imposed by quantum computing on PKC. In the future, a Cryptographically Relevant Quantum Computer (CRQC) (MOSCA; PIANI, 2022) may weaken KEX protocols and digital certificates built with classical cryptography. Here, classical cryptography means PKC schemes based on the difficulty of the Integer Factorization Problem (IFP), Discrete Logarithm Problem (DLP), or Elliptic Curve Discrete Logarithm Problem (ECDLP). Such schemes compose the majority of PKC algorithms being used nowadays, like RSA and ECDSA. This threat is even more worrisome when considering attackers collecting data-in-transfer today expecting to be able to decrypt it in the future (called *store-now-decrypt-later* attacks (BINDEL et al., 2019)). Such an attack threatens the confidentiality of the users' connections. In summary, users of traditional computing environments and networks now require protection against attackers with quantum computing capabilities.

The quantum threat has stimulated the study and development of new cryptography

schemes. These schemes are called by Post-Quantum Cryptography (PQC) or quantum-resistant cryptography (BERNSTEIN; LANGE, 2017) if no known quantum algorithm can "break" them. At some point in the (near) future, the security requirements of network protocols, applications, IoT devices, and others will include a transition between two "worlds": a pre-quantum world, where the classical cryptography methods are used, and a post-quantum world, where PQC will replace the traditional methods. For an easier transition, a hybrid period is established as the first transition phase, thus combining security from alternatives of both pre and post-quantum worlds. Until the confidence in PQC security is fully established, one can expect hybrids to still be in use, even after a CRQC arrives. Figure 1 illustrates the scenario.

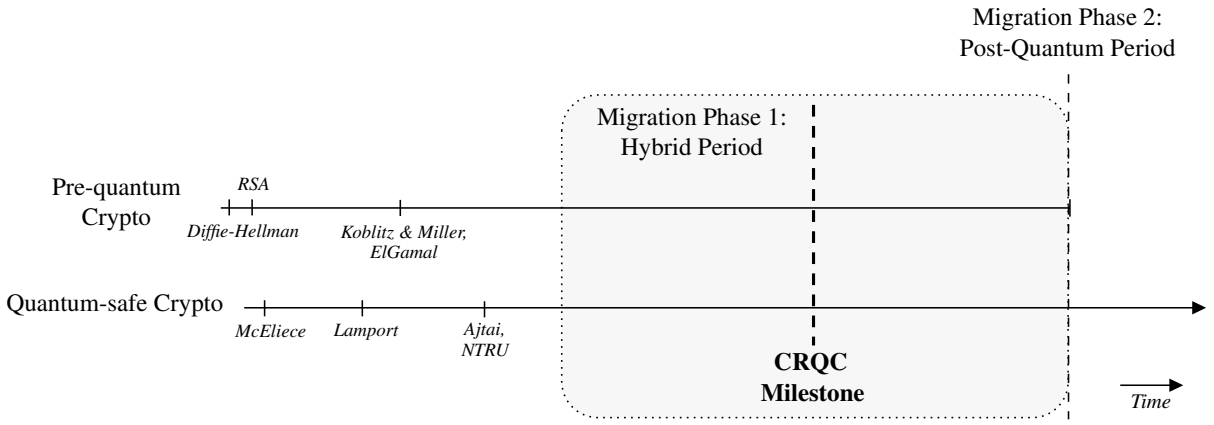

Figure 1 – A transition between two cryptography worlds.

Hybrid PQC is an approach of adopting PQC in implementations, which supports Post-Quantum Cryptography (PQC) but maintains compatibility with the classical cryptography algorithms. In this work, the term Hybrid should not be confused with Hybrid Encryption (KUROSAWA; DESMEDT, 2004), a conjunction of symmetric and asymmetric cryptography. The first reason for selecting the hybrid mode regards confidence in security. Only after the confidence in PQC is fully established abandoning the classical cryptography in use today is recommended. Generally, the classical methods have higher confidence and years of utilization, both academically and by industry standards. Therefore, the hybrid mode is recommended and it means that both PQC and traditional algorithms are used in conjunction. Hence the security of the construction holds until at least one algorithm is not broken.

One problem for migrating to PQC is that it significantly increases the sizes of cryptographic objects, such as public keys and signatures. This challenge bears relevance to many use-case scenarios of cryptography, most notably the network protocols' authentication and KEX mechanisms. For example, some PQC algorithms like Classic McEliece are infeasible for regular TLS handshakes due to the size of their public keys. Network protocols were designed with classical cryptography sizes in mind. Thus they might not be prepared to handle PQC without performance drawbacks. Furthermore, one can expect that updating protocols for PQC may take several years (KAMPANAKIS; LEPOINT, 2023). Hybrids require adding at least one

PQC algorithm (called ingredient) to the classical scheme, it increases the number of cryptographic objects being transmitted between parties, and it requires a cryptographic combiner that has to combine the algorithms securely (i.e., keeping the security properties of the combined ingredients).

1.1 MOTIVATION

In summary, the main question pursued in this work is how to adopt hybrids in network protocols. There are many challenges in this regard. Studies found potential delays caused by PQC in network protocols (such as TLS) caused mainly by authentication components (e.g., PKI certificate chains) (SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020; PAQUIN; STEBILA; TAMVADA, 2020). In order to address those issues, researchers started several benchmarks of PQC in network protocols, while others proposed protocol changes to better fit PQC, such as the KEMTLS approach (SCHWABE; STEBILA; WIGGERS, 2020a; SCHWABE; STEBILA; WIGGERS, 2021; CELI et al., 2021). Such findings, changes, and evaluations are fundamental, so the impacts on performance imposed by PQC – with or without the hybrid mode – can be known and addressed in advance. Consequently, adapting and evaluating protocol changes must be made before the advent of the quantum computer for a smooth PQC transition.

On the other hand, Hybrid PQC has some disadvantages compared to a PQC-only deployment. Hybrids add computational times and sizes of classical cryptography, plus other design decisions to be made in the PQC implementation. Although reports show minor penalties when comparing hybrids to PQC-only deployments, these studies might need to be completed since they do not evaluate different security levels when instantiating cryptographic algorithms. Moreover, some network protocols still need to be extensively evaluated with PQC (or hybrids), such as ACME, and some other protocols' implementations lack support for hybrids, such as KEMTLS. Adding support to hybrids is essential to the PQC adoption in practice.

With the rise of new computing environments, including 5G networks and Vehicular Networks (VNs) (HUSAIN et al., 2019), additional efforts and specific analysis might be required. Noteworthy, TLS is being considered the leading candidate for deployment in these networks, thus augmenting its importance. Furthermore, other challenges may appear when deploying hybrids in TLS for different applications due to their specific requirements. Using VNs as an example, they are critically affected by the cost of communication payload data and message headers (HUSAIN et al., 2019). In short-range VNs, end-to-end delay must be decreased to meet the quality of service (QoS) requirements (KADHIM; SENO, 2019). VNs connect On-Board-Units (OBUs) between cars, where hardware and software updates are complex due to their long-term lifetime. These particularities can impose additional challenges when deploying PQ-TLS (and hybrids) in such environments.

There is also a major issue in hybrids that the literature still needs to address. Consider a hybrid with two ingredients: a PQC and a classical algorithm. The only hope for post-quantum

security in this hybrid relies solely on the PQC ingredient. If the PQC ingredient is broken by a newly-discovered quantum (or classical) attack, the hybrid is not quantum-safe anymore. In such a case, all of the PQC migration efforts will be lost, and the *store-now-decrypt-later* vulnerability remains: the attacker breaks the PQC part with either a classical or quantum computer and then breaks the classical part aided by a quantum computer. There are examples of classical attacks in former PQC candidates like Rainbow (BEULLENS, 2022) and SIKE (CASTRYCK; DECRU, 2022), where classical computer hardware was used. If a hybrid construction with one of these schemes were chosen for the PQC transition, all transition efforts would be lost. Therefore, hybrid modes do not offer a quantum-safe contingency plan for adopters.

However, in preparation for such a complex migration event in cryptography, awareness of the quantum threats is an important concept. The reality is that several users, system administrators, and government agencies are concerned about the security of their applications against quantum computers (MOSCA; PIANI, 2022). However, this reality can only be generalized if awareness is achieved by different efforts, such as in educational contexts and industry standardization. In regards to standardization, there is the notorious (and still ongoing) NIST PQC standardization process (NIST, 2016). In practice, users will require a set of tools to assess and verify the quantum-safe properties of their connections. Given the urgency of *store-now-decrypt-later* attacks, this need requires increased attention.

This thesis deals with the above hybrid adoption problems: lack of tools or mechanisms for users to assess quantum-safe connections, lack of support and evaluations for hybrids in different scenarios, and lack of a method for quantum-safe contingency in hybrids. In this context, the research is initially guided by the following question: "How can network protocols be adapted to adopt and support hybrids as the PQC migration strategy?"

1.2 THESIS GOALS

In light of a global-scale migration towards PQC, this thesis aims to defend the hybrid PQC adoption approach by adapting, implementing, and evaluating it in different scenarios, thus facilitating PQC migration plans. Additionally, the specific thesis' goals are listed below.

- Identify hybrid PQC adoption challenges by performing a rigorous literature search. The objective is to aim at the known challenges and experiment with hybrids in practice, evaluating possible performance penalties.

- Mitigate problems of using hybrids by providing simple solutions whenever possible, followed by security analysis and experimentation in real-world and simulated environments.

- Contributing to the awareness of the quantum threats, PQC adoption challenges, and the corresponding solutions using hybrid PQC, by publicizing them in different venues and providing public hybrid PQC implementations for the community.

Those goals are in the direction of encouraging Hybrid PQC adoption in practice. Indeed, there are some scenarios where hybrids were already available and evaluated. OpenSSH version 9 uses hybrids by default (OPENSSH, 2022), the IETF draft for PQC in TLS provides only hybrid modes for KEX (STEBILA; FLUHRER; GUERON, 2023), and the hybrid mode experiments by Google and Cloudflare (BRAITHWAITE, 2016) are some examples. However, several other scenarios still require attention. Furthermore, after analyzing these other scenarios, new challenges might appear and be worthwhile awareness by the community. Therefore, this work's efforts focus on bringing hybrids in (1) different environments and (2) where they need more complete evaluations than what is available.

1.3 OVERVIEW OF THIS THESIS' CONTRIBUTIONS

All of the contributions of this thesis are related to improving the PQC adoption by using hybrids. This section details the particular PQC adoption problems that this thesis has addressed (labeled with a "P" followed by a number) for better comprehension of the contributions.

P1: Lack of a secondary study in the form of a Systematic Mapping Study (SMS) (KITCHENHAM; BUDGEN; BRERETON, 2011) classifying the literature related to (1) how hybrids are created and evaluated; (2) what is the efficiency and security guarantees of hybrids that were already evaluated; and (3) what are the open challenges that hybrid PQC faces. This problem might not be strictly related to the PQC adoption, but it restricts the awareness of hybrid modes and terminology, considering researchers and newcomers to network protocols.

P2: Lack of a tool for users wanting to know more about their TLS connections, such as if the TLS connection they made is protected against *store-now-decrypt-later* attacks (i.e. if it uses quantum-safe algorithms).

P3: Lack of support and analysis of automatic issuance of certificates with hybrid PQC. There is no knowledge about the impacts of PQC and hybrids in the ACME protocol, commonly used for issuing and renewing certificates for TLS servers.

P4: Lack of implementations with hybrids for a particular PQC adoption approach in TLS, namely KEMTLS (SCHWABE; STEBILA; WIGGERS, 2020a). Consequently, there is a lack of knowledge of how a hybrid version of KEMTLS would perform in practice.

P5: Lack of a quantum-safe contingency plan for hybrids. In long-term products (such as cars), such a contingency might be crucial since providing PQC-optimized hardware and updating software in such products is far from trivial.

Each of the above challenges is addressed in this thesis. First, Chapter 2 starts with the necessary background information for this work. Chapter 3 follows and describes the method-

ology, including a research methodology called Systematic Mapping Study (SMS). The SMS is the first contribution of this thesis. It addresses P1 by providing a secondary study on Hybrid PQC design, efficiency, security, and open challenges.

Chapter 4 details the main contributions of this thesis, divided into several sections. Section 4.1 shows the TLS 1.3 Handshake analyzer tool, addressing P2. The tool allows parsing of TLS 1.3 captured connections to show the user security and performance information. It tells whether the connection was created with quantum-safe mechanisms.

Viewing ACME as a TLS-enabler component, Section 4.2 addresses P3. It shows the initial impacts of PQC in ACME and how the protocol can be modified for faster issuance of TLS certificates with PQC (and hybrids). Our results show speed-ups of 4.22x compared to a non-modified version of ACME.

Given that modifying TLS by KEMTLS can be a promising PQC approach, Section 4.3 shows the design and comparisons of using hybrids in KEMTLS, thus addressing P4. The performance penalties are non-significant for hybrid KEMTLS under certain security levels, thus encouraging its use in practice.

The last contribution of this thesis addresses P5. Section 4.4 details the PKI Extended Lifetime Period (PKIELP) approach, which provides a novel hybrid concept for authentication. PKIELP is designed for specific scenarios, such as those related to VNs. PKIELP's implementation results in smaller sizes and faster TLS handshakes. Figure 2 summarizes all the mentioned approaches for PQC migration and related tools developed in this work.

For a better user perception of the PQC migration efforts, the TLS 1.3 handshake

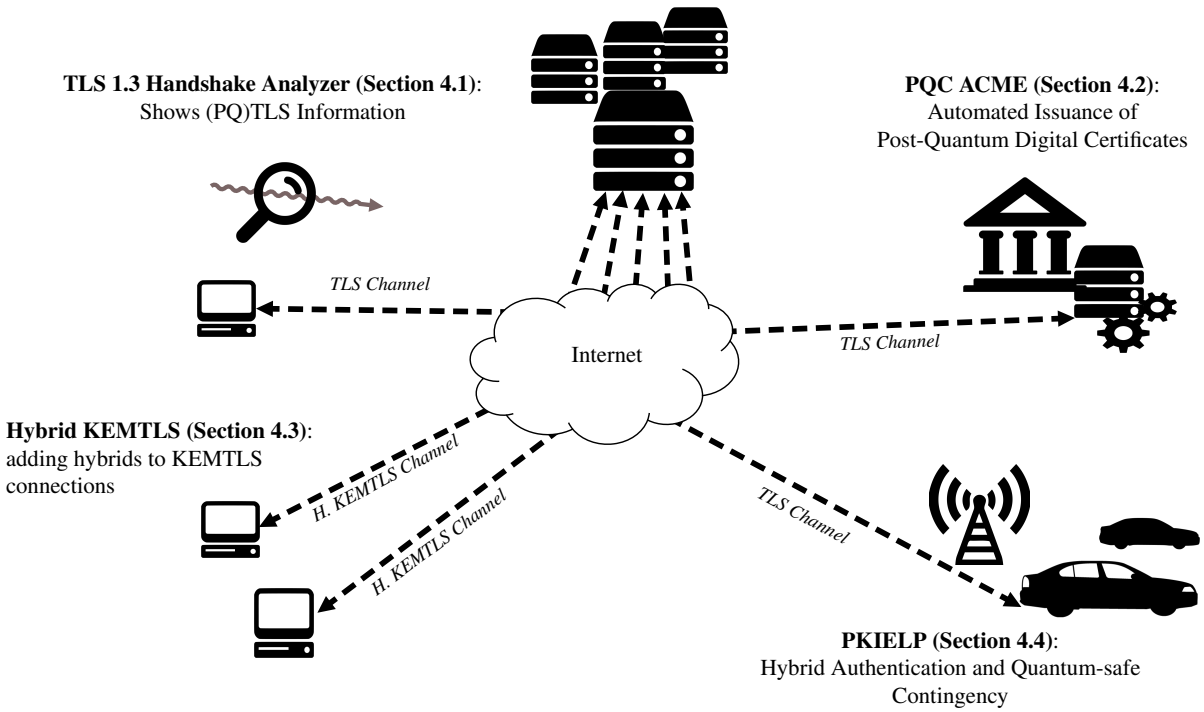

Figure 2 – Overview of this thesis (approaches and tools).

analyzer tool is provided. For easing quantum-safe TLS endpoint configurations, ACME was adapted, modified, and evaluated when issuing PQC and hybrid certificates. When comparing strategies for Internet connections, adding hybrids in the promising KEMTLS shows better confidence in security at small performance costs. Moreover, one can use the proposed PKIELP approach for specific scenarios, enabling quantum-safe TLS connections with a novel hybrid authentication and contingency. All contributions are discussed in Chapter 5. Lastly, Chapter 6 wraps everything up, giving future research directions not covered in this thesis' scope.

Some of the above contributions are already published. Table 1 lists the publications obtained during this Ph.D., including also papers currently under review. Not all publications are detailed in this thesis, such as those unrelated to PQC. This thesis' results that are already published are the PQC adoption problems P1 to P3. Note that there are different types of contributions and venues for better diversity.

Table 1 – List of Publications (including "preprints"). "PQC" means if the publication is related to PQC. ID Labels with a "T" refers to publications detailed in this thesis; "C" refers to collaboration with other author's publications; and "A" refers to additional contributions achieved during this Ph.D. (detailed in the end of this Section).

ID	Reference	PQC?	Type of Contribution	Year	Venue
T1	(GIRON; CUSTÓDIO; RODRÍGUEZ-HENRÍQUEZ, 2023)	✓	Regular Paper	2023	Journal (JCEN)
T2	(GIRON, 2023)	✓	Position Paper	2023	SECrypt 2023
T3	(GIRON et al., 2022) (preprint)	✓	Regular Paper	2023	Accepted at LatinCrypt
T4	(GIRON; NASCIMENTO; CUSTóDIO, 2023) (preprint*)	✓	Regular Paper	2023	-
T5	(GIRON; SCHARDONG; CUSTÓDIO, 2022)	✓	Workshop Paper	2022	SBSeg-SF 2022
T6	(GIRON, 2021)	✓	Ph.D. Poster Paper	2021	SecureComm2021
C1	(SCHARDONG et al., 2022)	✓	Regular Paper	2022	CANS 2022
C2	(MARCHIORI et al., 2021)	✓	Regular Paper	2021	SBSeg 2021
A1	(GIRON; MARTINA; CUSTÓDIO, 2021)	✗	Regular Paper	2021	Journal (Sensors)
A2	(GIRON; MARTINA; CUSTÓDIO, 2020)	✗	Workshop Paper	2020	ACNS-W 2020
A3	(GIRON; CUSTÓDIO, 2020)	✗	Regular Paper	2020	SBSeg 2020

1.4 INTEGRATION AND INTERDISCIPLINARY TOPICS

A great part of this Ph.D. work is in the context of an international collaboration project. The collaboration is registered between the Federal University of Santa Catarina and the Technology Innovation Institute (TII) from the United Arab Emirates (UAE)[1]. The project's main objective was to research the challenges of adopting post-quantum KEX and signatures. Collaborating in this project allowed integration with different researchers and improved the visibility of this research.

During this work, other research topics were addressed, in conjunction with different researchers, resulting in additional contributions. The integration with other topics is important since it allows a holistic view of recent aspects of computer security, not necessarily related to post-quantum cryptography adoption (or applied cryptography). This section summarizes the collaborations (C1-C2 from Table 1) and the additional contributions (A1-A3) below.

[1] Project registration number 202102364 at https://sigpex.sistemas.ufsc.br

One problem in the PQC migration is the complexity due to the variety of frameworks, applications, and network protocols. The OAuth 2.0 is a protocol designed for applications to get authorizations from Identity Providers (IdP), and the OpenID Connect framework extends OAuth 2.0 with mechanisms (such as multi-factor authentication) and policies for accessing user data. In conjunction, these two protocols are widely used by many users on the Internet, exchanging information from IdPs such as Google and Meta. However, OpenID and OAuth 2.0 depend on the TLS protocol and, consequently, Public-Key Cryptography. Therefore, the quantum threats also apply to those protocols. In this context, collaboration C1 (SCHARDONG et al., 2022) fortifies OpenID Connect and related protocols with PQC, providing implementation and experiments. The results showed an evident dependency on TLS timings, as TLS accounted for fifty percent of OIDC duration time but only in the high-latency connection scenarios.

The collaboration C2 (MARCHIORI et al., 2021) dealt with a different perspective on the PQC adoption. The main problem is related to Algorithm Substitution Attacks (ASAs), in which cryptographic implementations are modified (or replaced) to leak users' private information. C2's described two novel ASAs in post-quantum algorithms: the first attack leaks the Kyber's private key seed in the public key, and the second attack leaks a Falcon private key seed using two consecutive signatures. Timing analysis has detected the attack on Falcon but not in Kyber. Besides, it did not detect the presence of the attacked versions in a post-quantum TLS instance. C2's findings highlight the need for auditing efforts in post-quantum implementations to prevent such attacks.

Berndt and Liśkiewicz (BERNDT; LIśKIEWICZ, 2017) formally treat the relationship between a successful ASA and steganography. Steganography is, by definition, a way to hide that secret communication is happening over a legitimate communication channel. ASAs and steganography, when undetected, can be worrisome if used to hide illegal communications.

With the evolution of computing environments, new protocols and applications arise and start being adopted by different entities of society. Typically, these recent computing environments, such as blockchains, can be exploited by steganography in novel ways. A blockchain is a class of distributed databases or ledgers that are immutable and shared by network peers (PARTALA, 2018). At the beginning of this thesis work, it was found that blockchain applications can be an attractive target for steganographic attacks due to the following reasons:

- High availability of the data, since blockchains share it with all network participants.
- Strong integrity, achieved using linked lists indexed by cryptographic hashes, thus makes it difficult to revert a steganographic attack.
- Anonymity level, given that identities in blockchains are often anonymized, helps to hide attackers' identities.
- Detection difficulty, since blockchains exchange a significant amount of legitimate data through several components (such as hashes and digital signatures), that can carry hidden communication data inconspicuously.

One fictitious example of why an attacker would misuse blockchains with steganography is copyright violations. In this example, the attacker wants to disseminate content breaking copyrights. Even if unauthorized, the attacker could use steganography in a blockchain. The broadcasting feature present in blockchains would spread the content amongst the network. Such an attack would make the content available and hard to remove (due to the strong integrity property). Besides, such an attack would also harm legitimate users: they would be disseminating the blocks and therefore infringing copyright without knowing it. That is why developing steganography detection and prevention mechanisms is important to mitigate such threats.

Several types and implementations of blockchains are being used in practice, such as Estonia's KSI blockchain and the Brazillian's blockchain for academic degree certificates (ESI, 2023; RNP, 2023). Having numerous implementations increases the complexity of detecting steganographic attacks in blockchains. Therefore, two additional contributions of this thesis, labeled A1-A2 (GIRON; MARTINA; CUSTÓDIO, 2021; GIRON; MARTINA; CUSTÓDIO, 2020), aim to fill this gap, focusing on steganography detection for real-world blockchains like Bitcoin and Ethereum. Such a practical analysis was not done before, highlighting the gap in experimental studies in detecting blockchain misuse by steganography.

In this context, A1-A2's work proposes a steganographic analysis approach, allowing the processing of LSB data in hashes and block addresses sequentially and in clusters. The clusters are created based on heuristics, aiming to group related transactions in the blockchain. The approach was applied in 253 GiB and 107 GiB of Bitcoin and Ethereum blocks, respectively, and then a forensic tool was integrated to recover potential evidence of steganography use. Moreover, A1's work has analyzed up to 98 million Bitcoin clusters. As a result, bitcoin clusters could carry up to 360 KiB of hidden data if used for such a purpose. On the other hand, no steganography evidence has been found in the clusters or the LSB of hashes and addresses. It is important to provide additional effort in detection (also considering other types of steganography) to prevent such attacks in blockchains.

The contribution A3 (GIRON; CUSTÓDIO, 2020) is the last additional contribution of this thesis. In the context of Internet-of-Things (IoT) devices, their cryptography operations need random number generation processes. The generation of random numbers presupposes the existence of entropy sources. However, there are few entropy sources available. Often, cryptographic modules rely only on a single entropy source (to save energy), and the source is a black box, meaning users have to trust their product manufacturers. Given this scenario, A3's work describes "Bluerandom", an entropy source based on Bluetooth Received Signal Strength Indicator (RSSI), as an additional source for an IoT device. The approach's advantages are mainly lowering hardware costs since it gathers entropy from nearby devices and scalability when increasing the number of nearby devices. Moreover, Bluerandom as an additional source decreases the risk of trusting in a single entropy source device. After extensive statistical testing and comparisons, results show the feasibility of using Bluerandom as an additional source, feeding the device's entropy pool system. It is worth noting that Bluerandom could also be applied in the the PQC scenario.

2 BACKGROUND INFORMATION

This chapter introduces the concepts used in this thesis, plus initial contributions. Section 2.1 defines Post-Quantum Cryptography (PQC) and related concepts. Sections 2.3 and 2.4 describe TLS version 1.3 and ACME 2.0, respectively.

2.1 POST-QUANTUM CRYPTOGRAPHY

The asymmetric cryptography currently used nowadays do not protect against the advent of quantum computers. All of the cryptography based on the Integer Factorization Problem (IFP), Discrete Logarithm Problem (DLP), and Elliptic Curve Discrete Logarithm Problem (ECDLP) is vulnerable to Shor's algorithm (SHOR, 1994). However, powerful-enough quantum computers are not publicly available during this writing. Nevertheless, Post-Quantum Cryptography (PQC) is considered the primary solution for early protection. As a result, researchers foresee a global PQC migration event: several protocols and applications will start adopting PQC. PQC is considered resistant because it is based on different mathematical problems in which classical or quantum computers have no (known) efficient solution. Quantum-safe and quantum-resistant cryptography are similar terms to PQC (BERNSTEIN; LANGE, 2017).

Given that today's public-key cryptosystems are frequently used for authentication and Key Exchange (KEX) over the Internet, the following attacks are possible:

- Break confidentiality: by gaining access to the private key in a KEX process, the attacker learns about the symmetric encryption keys used in the user's communication. As a result, the attacker has access to the contents of the encrypted traffic.

- Impersonation: If the attacker has access to the victim's private key *sk* in a digital signature system, he or she can impersonate the victim by signing messages with *sk*. If a web server's private key is obtained, the attacker can establish a "fake" server, leading every user to believe that their connection is authentic. The impersonation of the server allows for other attacks, such as the disclosure of user data and communications.

In principle, a CRQC runs the Shor algorithm, allowing a quantum attacker to extract a private key from the victim's public key. There is currently no publicly available CRQC. As a result, until a CRQC arrives, public-key cryptosystems used for authentication cannot be utilized for impersonation. Experts believe that a CRQC will be accessible eventually, hence such cryptosystems will need to be changed (MOSCA; PIANI, 2022). Given the complexity of Internet authentication, such as X.509 PKIs, and the uncertainty of when a CRQC would be operational, applications and systems must be built in advance to prevent quantum attacker impersonation.

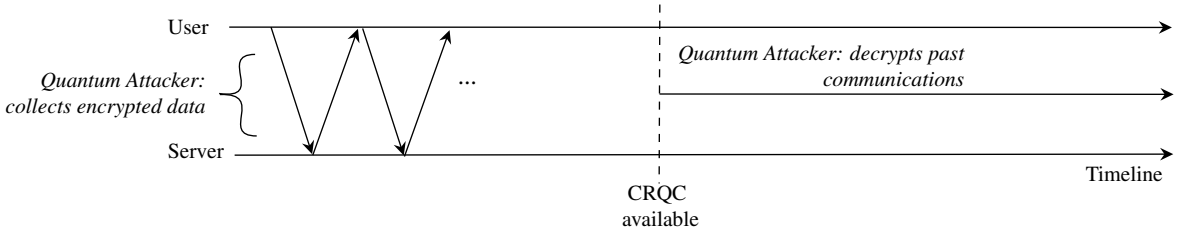

Figure 3 – Record-now-decrypt-later attack timeline

Vulnerable KEX mechanisms are concerning because they are vulnerable to a *record-now-decrypt-later* attack. Figure 3 depicts a situation in which a quantum attacker captures encrypted-communicated data today with the intent of decrypting it later. Given that KEX is commonly employed in network protocols such as TLS and SSH, quantum computers pose a threat to the security of today's communications. Furthermore, Grover's quantum algorithm (GROVER, 1996) poses a threat to confidentiality. It undermines symmetric encryption methods by halving the key-space search time, making brute-force attacks more effective. However, experts think Grover's approach is tough to implement in practice. A simple mitigation to Grover would be to double the security parameters of symmetric primitives while maintaining the original security assumption. As a result, the threat posed by Grover's algorithm is less alarming than the vulnerability of KEX to Shor's algorithm.

The first response to the quantum threat is the use of PQC to replace vulnerable algorithms. Sometimes dubbed "PQC Drop-in replacement" or "PQC-only deployment", the susceptible KEX and authentication procedures are replaced only by PQC alternatives. Applications utilizing PQC can withstand quantum threats, although classical computation still poses risks. SIKE and Rainbow, two promising PQC algorithms, are now regarded vulnerable to traditional attacks (CASTRYCK; DECRU, 2022; BEULLENS, 2022). These examples imply that moving to PQC should be done with caution. In other words, a PQC drop-in replacement should be performed only when confidence in its security has been established. Instead, hybrid mode is advised for early (and smoother) acceptance (STEBILA; MOSCA, 2016).

In TLS, public-key cryptography is often used for Key Exchange (KEX) and peer authentication. ACME also use it for issuing public-key certificates. In order to protect network communications from quantum adversaries, PQC schemes must replace the vulnerable ones. For example, in TLS, a Key-Encapsulation Mechanism (KEM) and a Digital Signature from PQC is required, or at least one PQC KEM (SCHWABE; STEBILA; WIGGERS, 2020a).

Integrating PQC schemes can significantly increase the size of cryptographic objects compared to classical schemes. Table 2 allows comparing sizes of PQC and classical algorithms. Note that the sizes are different for each NIST security level, where each level means that the scheme is as hard to break (using exhaustive key search) as symmetric AES-128 (level one), AES-192 (level three), and AES-256 (level five) (MOODY, 2018).

Table 2 – Comparison of classical and PQC schemes

Algorithm Name	Parameter Set Name	Public Key size (bytes)	Ciphertext or Signature size	(S)ignature or (K)EM/KEX	Quantum-safe?
NIST P256	secp256r1	64	64	K	✗
NIST P384	secp384r1	96	96	K	✗
NIST P521	secp521r1	132	132	K	✗
Kyber	Kyber512	800	768	K	✓
	Kyber768	1184	1088	K	✓
	Kyber1024	1568	1568	K	✓
Saber	LightSaber_KEM	672	736	K	✓
	Saber_KEM	992	1088	K	✓
	FireSaber_KEM	1312	1472	K	✓
NTRU	NTRU-HPS-2048-509	699	699	K	✓
	NTRU-HPS-2048-677	930	930	K	✓
	NTRU-HPS-4096-821	1230	1230	K	✓
ECDSA	ecdsa_secp256r1	64	64	S	✗
	ecdsa_secp384r1	96	96	S	✗
	ecdsa_secp521r1	132	132	S	✗
Dilithium	Dilithium2	1312	2420	S	✓
	Dilithium3	1952	3293	S	✓
	Dilithium5	2592	4595	S	✓
Falcon	Falcon-512	897	690	S	✓
	Falcon-1024	1793	1330	S	✓
Sphincs+	SPHINCS+-SHAKE256-128f-simple	32	17088	S	✓
	SPHINCS+-SHAKE256-128s-simple	32	7856	S	✓
	SPHINCS+-SHAKE256-192f-simple	48	35664	S	✓
	SPHINCS+-SHAKE256-192s-simple	48	16224	S	✓
	SPHINCS+-SHAKE256-256f-simple	64	49856	S	✓
	SPHINCS+-SHAKE256-256s-simple	64	29792	S	✓

2.2 HYBRID PQC

Hybrid PQC is a method of implementing PQC that provides Post-Quantum Cryptography (PQC) while being compatible with traditional cryptography techniques. As previously stated, the first reason for choosing the hybrid option is confidence in PQC security. In general, traditional approaches have more confidence and years of use, both academically and by industrial standards. As a result, the hybrid mode is advised, in which both PQC and classical algorithms are utilized in tandem. As a result, the construction's security is maintained till at least one algorithm is not cracked. Noteworthy, the term Hybrid should not be confused with Hybrid Encryption (KUROSAWA; DESMEDT, 2004), which is a combination of symmetric and asymmetric cryptography.

In practice, hybrids are being proposed as follows:

- Concatenation of KEX objects (STEBILA; FLUHRER; GUERON, 2023): two (or more) KEX mechanisms execute in parallel, but the exchanged public keys (or ciphertexts) of the KEX parties are concatenated before sending. Each KEX will produce a shared secret concatenated prior to symmetric key-derivation. In this way, symmetric keys are produced with seeds from a classical and a PQC algorithm. An attacker would need to break each KEX to obtain the symmetric keys.

- Dual signatures: For authentication with digital signatures, the same data can be signed twice but using different signing keys (a PQC and a classical one). The verifier checks the two signatures for authenticating the data. Legacy implementations can be compatible but will check only the classical signature. Regarding the PKI infrastructure for authentication, there are three possibilities (OUNSWORTH; PALA, 2019; OUNSWORTH, 2023):
  - Composite hybrid: in this strategy, two (or more) cryptographic objects are concatenated, composing the hybrid. For example, a composite instance would concatenate two signatures or two public keys, one from PQC and the other from a classical algorithm. Composite is simple to implement in practice.
  - "Catalyst Hybrid": similar to the composite, but applied to digital certificates, the PQC algorithm objects are added through X.509 extensions. Such extensions can be non-critical to avoid damaging legacy implementations.
  - Parallel PKIs: in this mode, the implementation adds a second PKI (or more), which uses PQC algorithms only. Adding a second PKI probably incurs into a new set of certificates to be handled by the implementation (called certification paths or certificate chains).

The Open Quantum Safe (OQS) Project is a notorious effort to provide a cryptographic library for use by the community. In addition, OQS provides example implementations (such as OQS-OpenSSL) and programming language *bindings* for broad adoption. In their implementations, the hybrid mode is not only recommended but present. Other implementations followed the same strategy, such as the Bouncy Castle (FACTOR, 2023), CIRCL (FAZ-HERNáNDEZ; KWIATKOWSKI, 2019), and OpenSSH (OPENSSH, 2022), in this case, providing hybrid modes for KEX operations. Table 3 summarizes implementations and applications using hybrid modes. The industry's interest in hybrid modes is evident, considering the Google, Cisco, and Cloudflare experiments (BRAITHWAITE, 2016; WESTERBAAN, 2021; KAMPANAKIS, 2020), and Internet Engineering Task Force standardization drafts (STEBILA; FLUHRER; GUERON, 2023). They focus on the hybrid KEX for PQC adoption.

Table 3 – Popular cryptographic implementations and its PQC support features (if present).

Implementation	**Release Version**	**PQC Support?**	**Hybrid mode?**
OQS-OpenSSL	1.1.1	✓	Recommended
OpenSSL	3.1.0	✗	Not present
OpenSSH	9.0	✓	Default
Bouncy Castle	1.73	✓	Present
Wolf SSL	5.6.0	✓	Present
Mbed TLS	v3.4.0	✗	Not present
CIRCL	v1.3.2	✓	Present

Several reports suggest minor performance penalties when comparing hybrids to PQC-only replacements, for example, when benchmarking TLS (PAQUIN; STEBILA; TAMVADA,

2020; BRAITHWAITE, 2016). Such minor penalties encourage the adoption of the hybrid strategy. For more details, refer to Section 3.1, which gives a secondary study regarding the performance and security of hybrids.

2.3 TLS VERSION 1.3

TLS generally provides secure communications for Internet and mail servers and also sets up VPN connections. Transport Layer Security version 1.3 is defined in RFC 8446 of August 2018 (RESCORLA, 2018). An overview of TLS 1.3 is given below, focusing on the technical details regarding public-key cryptography operations.

Figure 4 shows three main components of TLS 1.3: A handshake protocol, a record protocol and an alert protocol. The only requirement of TLS 1.3 is a reliable, in-order data stream channel between communicating parties. The Handshake protocol in TLS 1.3 performs an Authenticated KEX, where a TLS client connects to a TLS server. The typical use case for TLS is server-only authentication, but mutual authentication can also be used. The handshake is responsible for negotiating cryptographic parameters, verifying the authenticated parties and establishing shared keying material for later use. After a successful handshake, the Record protocol protects the traffic between the communicating parties using symmetric encryption. If an error occurs, the Alert protocol is triggered.

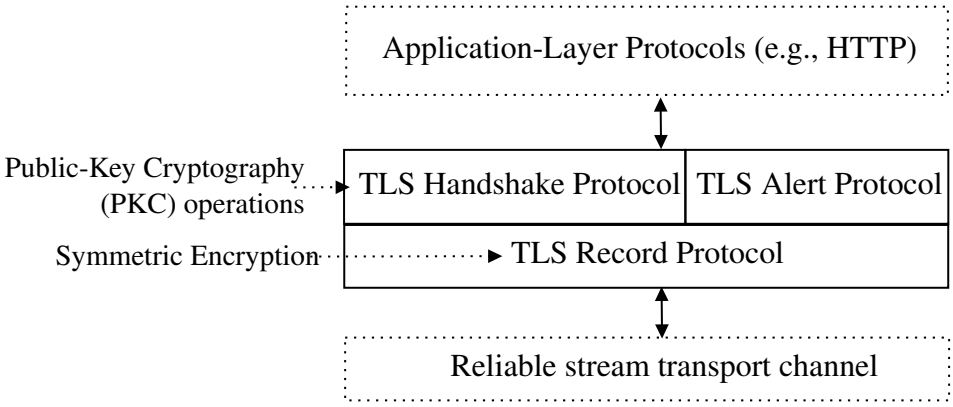

Figure 4 – TLS 1.3 Main Components.

This section focuses on the handshake since TLS handshakes rely on public-key cryptography operations. Each handshake is composed of three phases:

1. Key Exchange (KEX): establishes shared keying material and negotiates cryptographic parameters. TLS 1.3 allows three modes for KEX: ECDHE (Elliptic Curve Diffie-Hellman Ephemeral), Pre-Shared Key (PSK), and PSK with ECDHE. Everything after the KEX phase is encrypted.

2. Server Parameters: TLS servers advertise extensions and cryptography support. Servers can also export information to the application layer if needed (such as extra cryptographic keys).

3. Authentication: performs authentication of the server and, optionally, the client. If PSK is not used, then a set of X.509 digital certificates is used for authentication. PSKs could authenticate handshakes if installed or established in a previous handshake (called TLS Session Resumption). Another objective of this phase is to assure handshake integrity by computing a Message Authentication Code (MAC).

The handshake protocol received two significant improvements in TLS 1.3 compared to TLS 1.2. The first is the 1-RTT mode: the handshake is designed to send the data securely after one round of two messages, i.e. the client can receive application data in 1-RTT. Figure 5 shows this mode. 1-RTT mode provides Forward Secrecy property to the communication if ECDHE is set in the handshake. Another improvement is the 0-RTT mode: application data can be sent at connection setup if a PSK is used with fewer security properties than 1-RTT mode. Although TLS 1.3 is the updated version, reports suggest that TLS version 1.2 is still used (F5 Labs, 2019; PARACHA et al., 2021). One of the reasons found by the study is that clients often encounter a lack of server support for TLS 1.3.

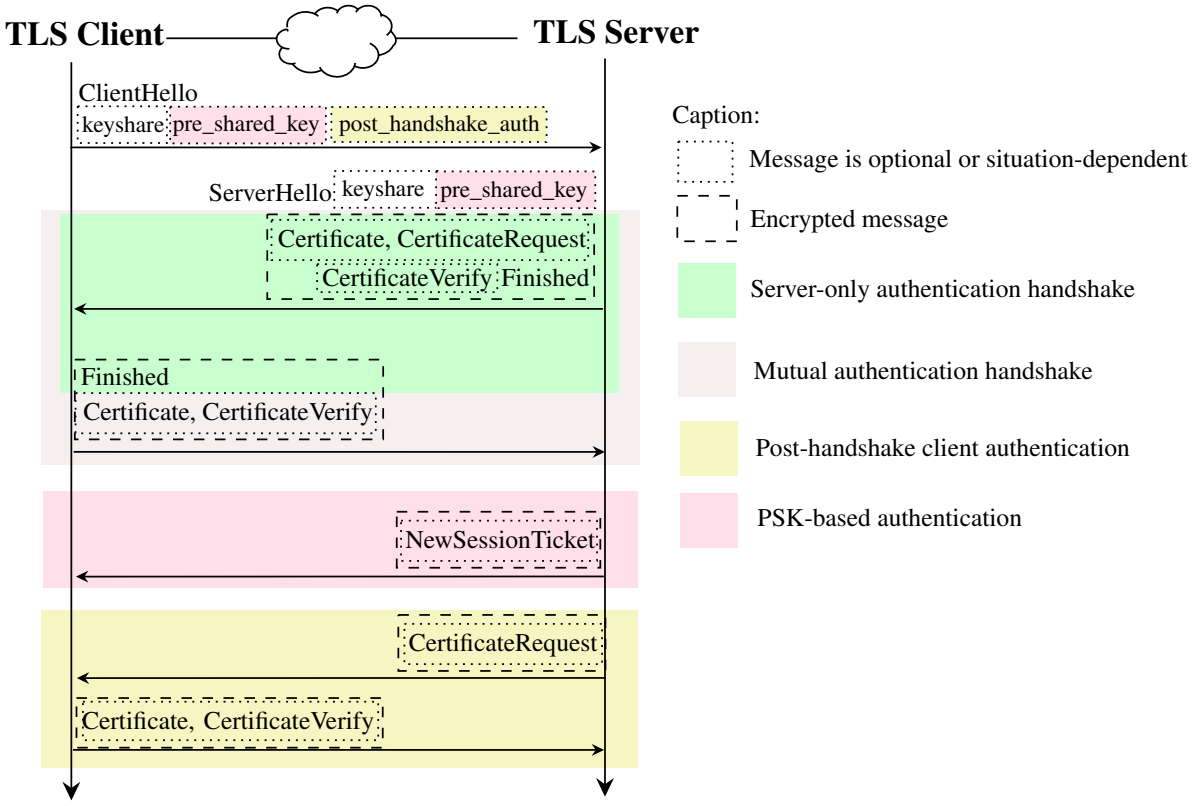

Figure 5 – TLS 1.3 Handshake and available authentication types.

The first message (`ClientHello`) comprises the following information: a random nonce, protocol versions, and a list of algorithms, such as symmetric ciphers and hashes, that are supported by the client. In addition, a `keyshare` contains ECDHE data and `pre_shared_key` contains PSK labels, either one or both of them are sent. In TLS 1.3, the optional messages described by the RFC 8446 can be turned into mandatory depending on the situation, e.g., if

PSK-based authentication, then `pre_shared_key` is sent. After both peers receive a `keyshare`, they can compute a shared secret (named Z) later used to produce encryption keys.

The server then replies with the `ServerHello` message. This message contains a random nonce, the cipher suite selected, and extensions. Depending on the extensions that the client has sent, the server responds with a `keyshare`, with or without PSK-related messages. Additionally, the following messages are responsible for the authentication: `Certificate`, `CertificateVerify` if not PSK-based authentication, `CertificateRequest` if the server's policy requires client authentication, and `Finished`. There is also an encrypted extensions message and the `Finished` message. Application data can be sent now, meaning the client can (securely) receive application data within 1-RTT. Except `ServerHello` and `ClientHello`, all messages are encrypted using keys derived from ephemeral ECDH keys (or PSK), which means that they are used for one particular session (and then discarded). TLS 1.3 does not recommend reusing keys between sessions.

The handshake is concluded when the client sends the `Finished` message, used for integrity check but also for key confirmation. A Finished is sent by both peers, computed by an HMAC of the handshake transcript, using keys derived in this session so that each peer can verify that they have established keys correctly. The finished indicates that the Record protocol can start by encrypting and exchanging application data, such as HTTPS requests.

The most common use case in TLS is to employ server-only authentication. Mutual authentication is also possible; in this case, the client must provide a certificate and a signature. RFC 8446 also allows post-handshake authentication for the clients, but only if advertised by the client at the beginning of the handshake (`post_handshake_auth` extension).

TLS 1.3 implements a key-derivation method called Key Schedule. The key schedule is responsible for deriving and updating encryption keys, and it is based on the OPTLS protocol (KRAWCZYK; WEE, 2016). Basically, it takes as input a shared secret Z after the KEX process and uses a key derivation method to derive a total of 9 types of new secrets that will then be derived into keying material for encryption. Each type is used for a particular purpose and has different inputs (e.g., labels).

TLS derives symmetric keys from a PSK or ECDHE exchange (or both). Since there are no skipping rounds in the key-derivation flow of TLS 1.3, this means that if a mode (e.g. PSK) is not in use, the corresponding parameter for HKDF will be zero. The flow is presented in Figure 6, composed of three main parts and in a tree-based structure. Figure 6 uses the same notation as in RFC 8446. A *salt* argument is at the top, and Z is on the left of each `HKDF-Extract` box. Note that Z is the output of a PKC-based primitive, i.e., ECDHE (vulnerable to quantum computers). The output of each flow is not the keying material yet; additional calls to `HKDF-Expand` are made to generate actual symmetric keys. After all the secret values have been used and no further computations are needed, RFC 8446 recommends that the secret should be securely erased.

The secrets for early data are generated in the first part of Figure 6. The PSK is used as an input with a salt equal to zero. The extracted secret is called "Early Secret" by RFC 8446,

which is expanded to create the *binder_key*, *_early*-prefixed secrets, and the salt for the ECDHE part of the derivation flow. Contextual information (named C_i in the Figure) is given in HKDF-expand. They can be composed of labels and the hash of handshake message transcripts. It is worth noting that the whole process is executed in both machines (client and server) to share the same secrets.

The second part of Figure 6 is where the secrets for the handshake encryption are generated. In this part, the "Handshake Secret" is expanded, generating both *[client or server]_handshake_traffic_secret*, and the salt for the last part of the key-derivation flow. In this part, the contextual information C_5 and C_6 is composed of a specific label and the hash of the transcript of both `ClientHello` and `ServerHello` messages.

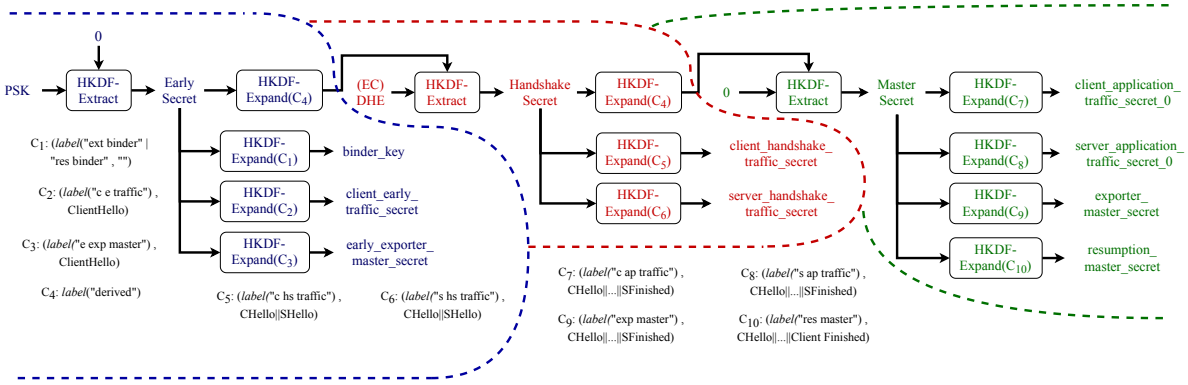

Figure 6 – TLS 1.3 Key-derivation flow (Adapted from RFC 8446)

The keying material used for the symmetric encryption (in the Record protocol) is derived from one of the secrets presented in Figure 6. Noteworthy, the key schedule generates the *[client or server]_handshake_traffic_secret*, which will be used to encrypt handshake data, and *[client or server]_application_traffic_secret_0*, which will be used to encrypt application data. Encrypting handshake data such as server public-key certificates avoids passive attackers. However, it does not prevent active attackers from connecting to the server and obtaining that information. Gaining access to a public certificate is not a problem since it is not enough to enable a Man-in-the-Middle attack: server certificates are issued by Certificate Authorities (CA), organized in hierarchies, which both TLS peers trust before connecting. Therefore, active attackers must first access the server's private information, e.g. private keys. In order to ease the trust configuration in peers, the ACME protocol comes into place.

2.4 ACME VERSION 2.0

The Automatic Certificate Management Environment (ACME) (BARNES et al., 2019) version 2.0 protocol has made a significant contribution to the widespread usage of digital certificates to ensure Internet authenticity. ACME provides services for online identity verification as well as certificate administration. The protocol's primary goal is to reduce the requirement for human intervention in establishing web servers and managing certificates. In practice, ACME is

currently a critical component of Let's Encrypt, one of the major CAs on the Internet (BIRGE-LEE et al., 2021). Furthermore, many certification authorities and PKI suppliers are incorporating the ACME protocol into their products since it simplifies and improves the service they give to their clients.

ACME permits an ACME server (owned by an Issuer CA) to give the ACME client a Domain-Validated (DV) digital certificate. Figure 7 depicts such a scenario, showing the protocols and roles involved in ACME version 2. Typically, there is a "client host" machine where an ACME client and an HTTPS server execute. The client host uses ACME to configure HTTPS/TLS servers and executes one (or more) HTTPS/TLS servers for secure communication with the users or applications. The host has a repository of trusted certificates previously configured to trust the certificate(s) related to the ACME server. When the client connects to the ACME server, the client host creates a TLS client because ACME communications require TLS for security. On top of TLS, the ACME client uses a public-key pair to sign ACME messages called "account keys", e.g. when asking for certificates.

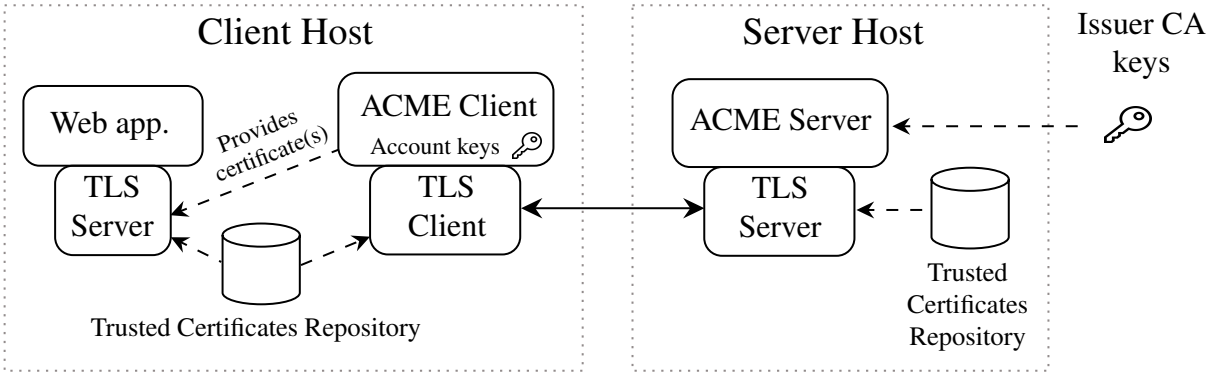

Figure 7 – ACME Typical scenario.

From the ACME server's perspective, there is a "server host" machine, also with a trusted store of certificates, and it is previously configured to serve ACME securely by a TLS server. On the server side, there is one certified public-key pair to establish TLS connections and a second public-key pair, which belongs to the Issuer Certificate Authority (Issuer CA), that will be used to sign certificates issued by the ACME server to ACME clients.

ACME's domain name issuing and validation is completely automated. However, ACME requires two communication channels: (1) the ACME Channel, which is TLS-protected, and (2) the Validation Channel, using protocols like HTTP, DNS or TLS-ALPN (Let's Encrypt, 2020). Although other types of certificates are available on the Internet, such as Organization-Validated (OV) and Extended Validation (DV), they are not permitted in ACME because issuing such certificates is difficult to automate and involves significant human involvement in the protocol.

Using the account keys, ACME clients send messages based on the JSON Web Signature (JWS) standard (JONES; BRADLEY; SAKIMURA, 2015) using HTTPS/TLS requests.

Figure 8 shows the required ACME messages for issuing an X.509 certificate. We divide the issuance process into three steps: (1) account creation; (2) challenge; and (3) issuance. Since the communication uses HTTPS requests, the ACME client must trust the ACME server's certificate. Often, Root CAs are pre-installed and therefore assumed to be trusted by both communicating peers. Note that this trust is established by a chain of certificates for the ACME communicating channel. The chain is not necessarily the same for the certificates issued to the clients.

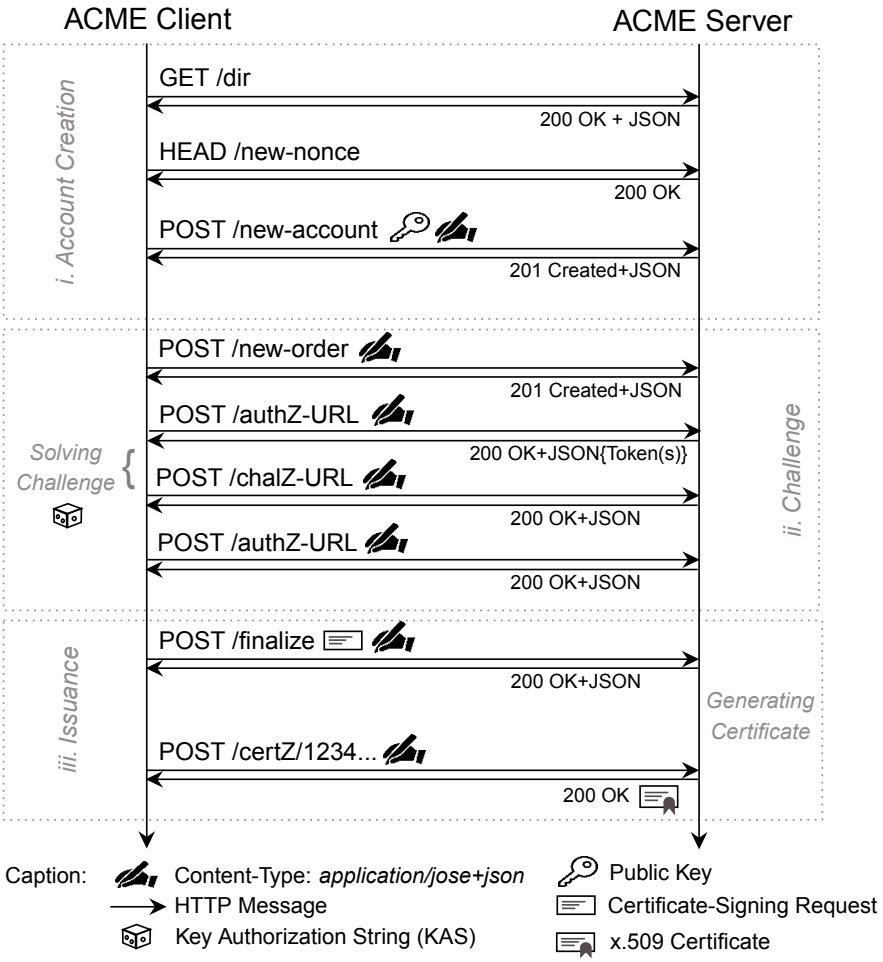

Figure 8 – ACME Issuance Overview

The client requests an account with the ACME server in the first phase. The client account may include contact information and is linked to a key pair generated by the requesting party. To begin the creation process, the client uses `GET /dir` to request server resources, and the server responds with an HTTP code (200 on success) and a JSON payload. The JSON provides the URLs for each required resource as well as the Terms of Service. If this is the client's first connection, it need a new nonce and requests one by sending a `HEAD /new-nonce` message. At this point, however, observe that the ACME server does not have any validation of the claimed identity other the freshly registered authentication key (account key). The client's subsequent HTTP requests must now be signed with the account key. It is important to highlight

that ACME's account is anonymous: the server can not guarantee the client's identity (at this point). That is why an ACME challenge is needed.

The second stage is the Identifier Validation Challenge. The ACME protocol standard solely addresses identifiers associated with domain names. For example, HTTP-01, DNS-01, and TLS-APLN-01 (Let's Encrypt, 2020), with HTTP-01 being the most widely utilized. To complete such a challenge, the client must demonstrate ownership of the account key as well as over the identifier. In HTTP-01, the client must serve a file through HTTP. The file contains the Key Authorization String (KAS). A KAS comprises a 128-bit random token (created previously by the server), a dot separator ('.'), and the base64-encoded key fingerprint. Therefore, a KAS binds the token to the account keys. The ACME server obtains and validates the file using HTTP (shown in Figure 9).

The middle part of Figure 8 shows an abstract challenge-solving step. First, the client orders a new certificate (`POST /new-order`). The reply includes the available challenges, their respective URLs and KAS. Each challenge requires a KAS, which is prepared on-demand, meaning authorization requests can fail, and the client may need to retry the request. Besides, there is a state (e.g., pending, valid, deactivated); therefore, the server can expect several requests using the same KAS until the certificate is issued. Therefore, the client must check the status of the desired KAS (first `POST /authZ/...`) and then execute the desired challenge. Following that, the client sends a `POST /chal/...` to the server, alerting it that the challenge has been completed, and then waits for the server to validate the challenge. The client sends `POST /authZ/...` requests to determine the status. When the challenge is valid, the server considers it to be completed. The authorisation is stored on the server and marked as valid for a length of time (not specified by RFC 8555).

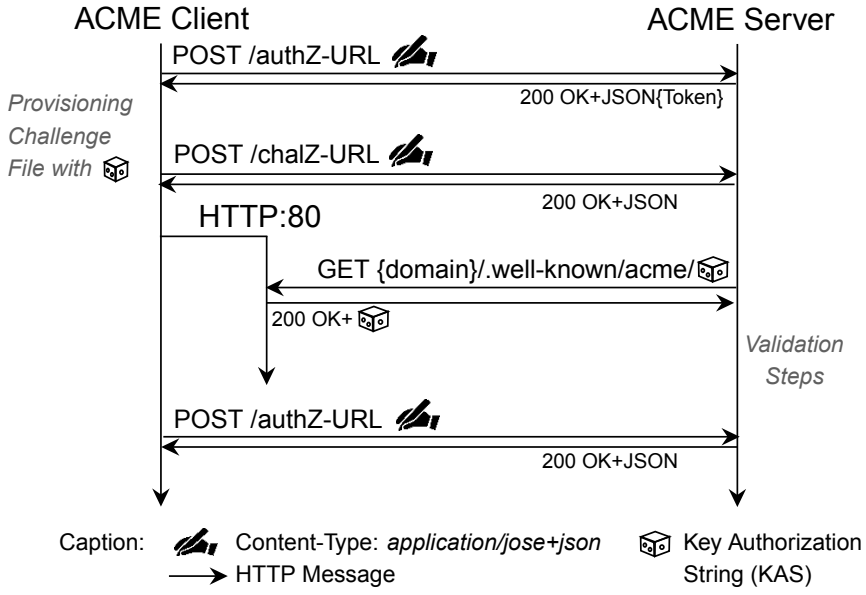

Figure 9 – HTTP-01 challenge flow.

The ACME specification provides challenges for proving identities for Domain Vali-

dated (DV) certificates (BARNES et al., 2019). Regarding the specifics of an ACME challenge, the client must prove to the challenger that: (1) it holds the private key registered to the challenger and (2) controls the identifier under validation. The specification allows other identifiers to be proposed, but the protocol's design focuses on DV certificates. Both HTTP-01 and DNS-01 challenges are described in RFC 8555.

HTTP-01 is the most common challenge (Let's Encrypt, 2020), providing an easy way to secure a web server with HTTPS. Basically, the challenger makes a GET request for a file in one (or more) HTTP servers under control by the challengee. The file contains the KAS for a previously-authorized order (see Figure 8). The certificate can be issued if the KAS downloaded by the challenger matches the KAS of the client's account. Otherwise, the challenge fails. When Figure 8 is combined with Figure 9, they compose a full ACME flow, from account registration to issuing the requested certificate.

With the server's KAS information, the client serves it in its HTTP server at a predefined location. The server seeks the challenge file using a GET request to the desired domain using port 80. The ACME standard obliges to use port 80 in this challenge (which can be problematic if a firewall blocks traffic). Then, the client notifies the ACME server to validate the challenge using the first POST request as shown in Figure 9. The validation steps include checking the response (e.g., if the domain name matches the previous order information) and, most importantly: (i) if the KAS inside the downloaded file matches; and (ii) if the digital signatures (in the POST requests) can be verified using the corresponding account's public key. In practice, the HTTP challenge (and the other types) can consume more POST requests to `authZ` endpoint than shown in Figure 9. The ACME client will repeat such a POST request until the order status is "valid" (or "invalid" in the event of an error). Repeated requests can increase network traffic, in particular when considering multiple clients at the same time.

The issuing procedure can begin once the server has confirmed the challenge, as shown in the final section of Figure 8. A PKCS#10 Certificate-Signing Request (CSR) for the server is sent by the client in a `POST /finalize`. Note that the account key pair that signed the POST request is different from the one that produced the CSR. In instance, no public key for a known account may be found in the CSR. The server then generates the certificate and verifies the CSR. The client can then use a `POST /certZ/...`, sometimes known as "POST-AS-GET," to download the new certificate.

The ACME client terminates after downloading the certificate. ACME client implementations, such as Certbot (FOUNDATION, 2022), often store and automatically setup the certificate(s) in the web server repository. With a few command-line instructions, an HTTPS-secured web server can be started. Certbot also configures automatic renewal, which simplifies certificate administration tasks. It is worth noting that RFC 8555 does not distinguish between certificate issuance and renewal, implying that renewal is the flow from a new post to `/new-order`. Revocation is also possible by using the account or certificate private key, however analyzing such a feature is out of the scope of this work.

3 METHODOLOGY

The methodology of this work divides into two parts. First, and as an initial contribution of this thesis, Section 3.1 summarizes a Systematic Mapping Study (SMS) regarding hybrid PQC in Key Exchange (KEX). An SMS is a methodology of researching, classifying, and producing new insights from the available literature (on a given topic). Secondly, Section 3.2 describes the experimental methodology of this work. It specifies a testbed of experiments to evaluate the proposals presented in Chapter 4.

3.1 POST-QUANTUM HYBRID KEX: A SYSTEMATIC MAPPING STUDY

One of the methodologies used to understand a research topic comprehensively is the Systematic Mapping Study (SMS) (KITCHENHAM; BUDGEN; BRERETON, 2011). Typically, an SMS contains the following activities: (1) definition of a search protocol, specifying the research questions and selection criteria; (2) search execution, obtaining the primary studies; (3) filtering of primary studies, using the criteria; (4) analysis and report the findings. This thesis' section presents an SMS about Hybrid KEX; additional information can be found in the original publication (GIRON; CUSTÓDIO; RODRÍGUEZ-HENRÍQUEZ, 2023). This SMS follows the rigorous research activities proposed by Kitchenham et al. (KITCHENHAM; BUDGEN; BRERETON, 2011) and by Petersen et al. (PETERSEN; VAKKALANKA; KUZNIARZ, 2015). There was no previous systematic investigation addressing Post-Quantum (Hybrid) KEXs.

The meaning of Post-Quantum Hybrid KEX used in this work is presented in Definition 1. As the short-term scenario is being considered, in which the communicating parties cannot access quantum computers, only pre and post-quantum KEX methods compose this hybrid form. The term "hybrid", in short, refers to a combination of pre and post-quantum approaches, as defined below.

Definition 1 (Post-Quantum Hybrid KEX) *Considering a key space $\mathscr{K}$, a Post-Quantum Hybrid Key Exchange is a set of combined KEX methods (called KEX ingredients) $\hat{K} = (K_1, K_2, ..., K_N)$, negotiated by the communicating parties, in order to share a combined secret information $\hat{Z} = Z_1 \diamond Z_2 \diamond ... \diamond Z_N$ with each other, where the following properties hold:*

1. *at least one ingredient is a pre-quantum KEX method, and at least one is a post-quantum KEX; and*

2. *the combiner operation ($\diamond$) to compute $\hat{Z}$ keeps the security of $\hat{K}$ as long as one of the KEX ingredients remains secure.*

3.1.1 Search Protocol

One objective of this Systematic Mapping Study is to identify and classify Hybrid KEXs proposed in the literature. This objective is based on three main Research Questions (RQs) described below:

RQ1: "How the cryptographic data can be **conveyed** in Hybrid KEX?" – The transport of (ephemeral and long-term) public keys can be a problem when instantiating a hybrid mode (i.e., in a network protocol). In addition, such data is used in the input of some key-derivation methods. So, this research question aims to identify how the cryptographic data of classical and post-quantum algorithms can be conveyed in the initial steps of a Hybrid KEX.

RQ2: "What are the methods for **combining** cryptographic data in Hybrid KEXs?" – This research question addresses how the *combiners* can be constructed in a Hybrid KEX. The cryptographic data here is composed of the shared secrets that are input to the key-derivation method.

RQ3: "Which **methods** and functions are proposed in the literature for **key-derivation** in Hybrid KEXs?" – The answer to this research question aims to identify how keying material is derived in Hybrid KEXs that have been proposed in the literature.

The following research sources were queried to gather primary studies: 1. ACM Library, 2. Google Scholar, 3. IEEEXplore, 4. Science Direct, 5. SCOPUS and 6. SpringerLink. In this SMS, primary works accepted are journal manuscripts, theses, technical reports and conference papers, all written in English, but other types of work (e.g. books) are excluded. A candidate study that addresses fully or partially one of the RQs can be selected in the SMS. The query string was designed using the PICOC method (Population, Intervention, Comparison, Outcome, Context) as basis (PETERSEN; VAKKALANKA; KUZNIARZ, 2015). The search included the following key-words, arranged as follows:

(*Key Exchange* **OR** *Key Establishment* **OR** *Key Encapsulation*)
AND *Hybrid*
AND (*Derivation* **OR** *KDF* **OR** *Extract-then-Expansion*)
AND (*Post-Quantum Cryptography* **OR** *PQC*).

The Population part of the PICOC acronym is where the primary studies must be applied (KEX terms, in this case). The "Intervention" part states the method of interest, in this case, Hybrids. A key-derivation method can be in the extract-then-expand fashion, or, at least, it is expected that the studies will specify which KDF they use so that they can be compared (Comparison part). The expected "Outcome" comprises state-of-the-art proposals for Hybrid KEXs, but the string already defines these terms. Therefore, this part is omitted. The last part

contains the Context, where the main interest is in Post-Quantum Cryptography. The string design also considered the limitation of (at most) 8 boolean operators present when searching in some sources.

3.1.2 SMS Results

After selecting, reading and classifying relevant primary studies, this SMS resulted in 29 related studies to hybrid KEX. Table 4 presents a reference list of the selected studies. The list presents an initial set published in the original publication of this SMS. It also includes the papers from the "Snowballing" process (KITCHENHAM; BUDGEN; BRERETON, 2011). However, after 2020, **7** relevant studies appeared, so they were added to Table 4 (except this thesis' hybrid proposals). Those eight primary studies correspond to an update to the published SMS. Therefore, this thesis' SMS has selected **36** primary studies for analysis.

Table 4 – List of references for the selected primary studies

Number	Reference	Number	Reference
1	(GHOSH; KATE, 2015)	2	(BOS et al., 2015)
3	(STEBILA; MOSCA, 2016)	4	(SCHANCK; WHYTE; ZHANG, 2016)
5	(BOS et al., 2016a)	6	(VELÁZQUEZ, 2017)
7	(BINDEL et al., 2017)	8	(HESAMIAN, 2017)
9	(UNGER; GOLDBERG, 2018)	10	(GIACON; HEUER; POETTERING, 2018)
11	(KOCK, 2018)	12	(CHO, 2019)
13	(BRENDEL; FISCHLIN; GÜNTHER, 2019)	14	(BINDEL et al., 2019)
15	(HEESCH et al., 2019)	16	(CROCKETT; PAQUIN; STEBILA, 2019)
17	(HEIDER, 2019)	18	(OTT; PEIKERT; and other workshop participants, 2019)
19	(SCHWABE; STEBILA; WIGGERS, 2020b)	20	(SCHWABE; STEBILA; WIGGERS, 2020a)
21	(XU; GAO; LIM, 2020)	22	(PAQUIN; STEBILA; TAMVADA, 2020)
23	(ALKIM et al., 2020)	24	(TUJNER et al., 2020)
25	(DOWLING; HANSEN; PATERSON, 2020)	26	(PAUL; SCHEIBLE, 2020)
27	(SAARINEN, 2020)	28	(CAMPAGNA; PETCHER, 2020)
29	(SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020)	-	
30	(AZARDERAKHSH et al., 2021)	31	(HOWE; PREST; APON, 2021)
32	(STADLER et al., 2021)	33	(HUGUENIN-DUMITTAN; VAUDENAY, 2021)
34	(AVIRAM et al., 2022)	35	(JOSEPH et al., 2022)
36	(HERZINGER; GAZDAG; LOEBENBERGER, 2021)	-	-

This SMS classifies the state-of-the-art approaches and design challenges of Hybrid KEX. Following the research questions, three types of approaches are highlighted: (i) conveying approaches; (ii) combining approaches; and (iii) key-derivation methods. Figure 10 shows the proposed classification. In addition, hybrids' performance and security aspects are summarized; for a complete discussion, please refer to the original publication. The open problems are discussed at the end of this section.

Each part of the KEX process is classified by existing proposals found in the literature. First, the communicating parties must transmit cryptographic data (e.g., public keys) of all algorithms to start the protocol. This SMS classifies the conveying methods to transmit such data in this step. Secondly, a hybrid KEX must combine the shared secrets of each algorithm, using one combining approach. Finally, the output of the hybrid KEX is often used as input to a Key-Derivation Method such as HKDF (KRAWCZYK, 2010), allowing parties to derive

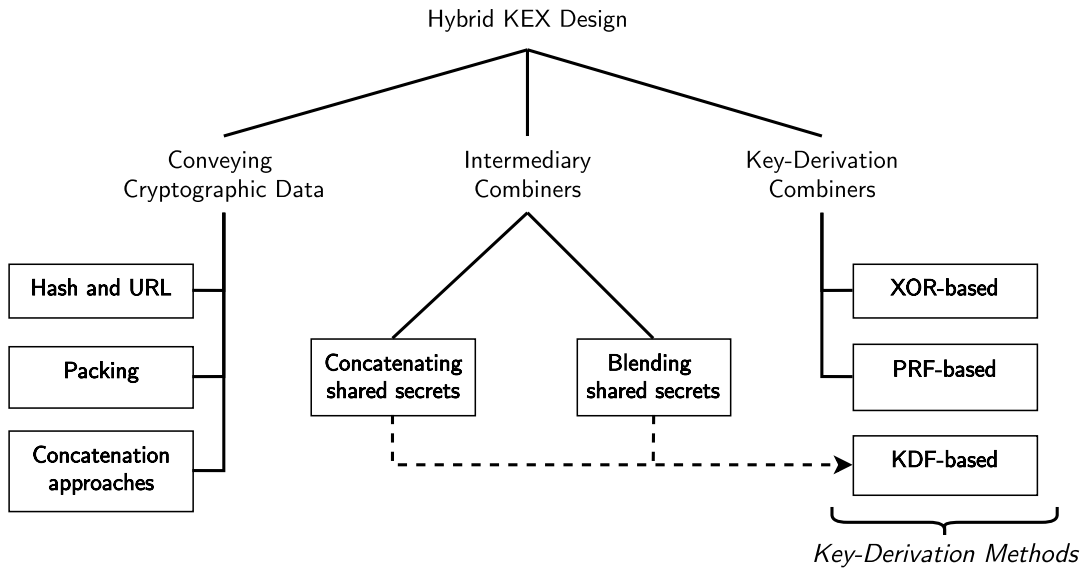

Figure 10 – A Classification of the Design considerations for Hybrid KEXs.

cryptographic keys. Such a classification allows newcomers and implementors to comprehend the hybrid design.

Conveying in Hybrid KEX. In the Hybrid KEX setting, the parties must exchange cryptographic data through the communication channel. Despite the KEX algorithm, the conveying payload in Hybrids is a sum of the payloads of each KEX algorithm (called "ingredients"). For example, the public keys of each algorithm might be sent to the receiver, which replies with the corresponding public keys or with ciphertexts encapsulating shared secrets.

The conveying approaches found in the literature are classified as follows. Concatenation approaches are straightforward and based on sending all of the cryptographic data in a concatenated fashion. Implementations are using this approach (STEBILA; MOSCA, 2016). The second type of approach is based on Packing cryptographic data. Compression is one example, such as compression through rounding, performed in Kyber (BOS et al., 2018); and packing public matrices, performed in Frodo KEX (BOS et al., 2016b; ALKIM et al., 2020). In addition, RFC 8879 (GHEDINI; VASILIEV, 2020) describes how to compress digital certificates for TLS. Hash-and-URL is the third type (HEIDER, 2019; KAUFMAN et al., 2014), where the hash of the data and a URL is sent, delegating the responsibility of obtaining the cryptographic data (using the URL and verifying the hash) to the other communicating party. Changing the protocol's state machine for conveying hybrids is also possible (STADLER et al., 2021; HERZINGER; GAZDAG; LOEBENBERGER, 2021), but it damages compatibility. One finding of this SMS is that the conveying challenge is just one of many focuses of the selected primary studies. Nevertheless, conveying in Hybrid KEXs incurs increased size and computational cost, directly proportional to the number of KEX ingredients.

Cryptographic Combiners. A combiner is a construction that takes as input N cryptographic schemes and combines them into one scheme. In the Hybrid setting, combiners are

used to mix the output of the combined algorithms, which can be shared secrets (when combining KEX) or signatures (when combining Signature schemes) (BINDEL et al., 2017). In this SMS, several works use the concatenation approach for Hybrid KEXs (STEBILA; MOSCA, 2016; UNGER; GOLDBERG, 2018; HEESCH et al., 2019; HEIDER, 2019; PAQUIN; STEBILA; TAMVADA, 2020; SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020). Concatenation is simpler but requires a key-derivation method to mix the outputs and derive keys. More elaborated approaches for combining are built using XOR (XOR-based combiners) or PRF (PRF-based combiners) as the core operation to combine shared secret information (GIACON; HEUER; POETTERING, 2018; BINDEL et al., 2019; CAMPAGNA; PETCHER, 2020).

Key-Derivation Methods. This SMS classifies the Key-Derivation aspect in Hybrid KEX in two ways. First, some works focus on something other than Key-derivation; thus, they rely on known derivation methods, probably to fit the Hybrid easily in network protocols. Examples include HKDF, HMAC or SHA-based derivation (CROCKETT; PAQUIN; STEBILA, 2019; PAQUIN; STEBILA; TAMVADA, 2020; SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020). On the other hand, the XOR-based and PRF-based combiners are designed to derive keying material by themselves, meaning they can generate shared secret information. From a practical perspective, Bindel et al. (BINDEL et al., 2019) and Aviram et al. (AVIRAM et al., 2022) discuss the adoption of combiners in TLS. Aviram's work performs a micro-benchmark of their proposed combiner. As expected, combiners do not harm performance, and their timings are orders of magnitude below network timings in normal conditions.

Performance of Hybrid KEX. This SMS showed primary studies that have instantiated hybrids in real-world protocols: TLS (1.2 and 1.3), SSH, IKEv2/IPSec, Tor, and Signal. The studies suggest minor performance penalties when comparing hybrids to PQC-only replacements. For example, Sikeridis et al. (SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020) experimented with hybrids in TLS and SSH protocols. Their work shows that the average latency of hybrids is less than 2% compared to PQ-only instances. Combining efficient elliptic curve operations with the PQC alternatives in a hybrid mode results in a good performance, considering computation time and byte costs. In summary, lattice constructions with ECDH provided an excellent trade-off between performance and size for the evaluated hybrids (GIRON; CUSTÓDIO; RODRÍGUEZ-HENRÍQUEZ, 2023).

Security of Hybrid KEX. Although several authors used the XOR operation in their constructions, Giacon et al. showed that the XOR combiner is not secure, for instance, to achieve IND-CCA security (GIACON; HEUER; POETTERING, 2018). On the other hand, by using a more elaborated construction, such as the Xor-then-MAC of Bindel et al. (BINDEL et al., 2019), it keeps IND-CCA security. These proofs are essential because the combiner is the core primitive of the hybrid design.

Regarding the attacker model, Bindel et al. (BINDEL et al., 2019) was the first to give security proofs for classical, partial and fully quantum adversaries. Other authors used assumptions to claim PQC security, and Ghosh and Kate used Song's Criterion, that allegedly "shifts" proofs to the quantum attacker scenario (GHOSH; KATE, 2015). In addition, Brendel

et al. (BRENDEL; FISCHLIN; GÜNTHER, 2019) and Dowling et al. (DOWLING; HANSEN; PATERSON, 2020) gave "protocol-level" proofs instead of focusing only on the cryptographic primitives. They used the Bellare-Rogaway Authenticated Key Exchange (AKE) models. Additional theoretical works then were proposed (HUGUENIN-DUMITTAN; VAUDENAY, 2021; AVIRAM et al., 2022).

Open Problems. Reducing communication cost is important for achieving better network performance. This SMS identified open problems regarding the (i) conveying approaches; (2) combiners; and (3) key-derivation methods. Conveying more data in hybrid forms impacts the protocols: in Tor networks because each client will have to communicate with three Tor cells (GHOSH; KATE, 2015). For compatibility purposes, implementations that use the concatenation approach duplicate pre-quantum public keys (CROCKETT; PAQUIN; STEBILA, 2019). Therefore, approaches for better conveying can be helpful, not only for Hybrid KEX but also for Hybrid Certificates (BINDEL et al., 2017).

Recently, the focus on hybrid design for KEX (and for Digital Signatures) puts forward the research on combiners and their security. These combiners are not thoroughly evaluated in practice yet (e.g., deploying in network protocols), except for concatenating prior to derivation (STEBILA; MOSCA, 2016; CROCKETT; PAQUIN; STEBILA, 2019; SCHWABE; STEBILA; WIGGERS, 2020a; SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020) and the XOR-then-MAC (PAUL; SCHEIBLE, 2020). Recently, the dual-PRF was analyzed in practice (AVIRAM et al., 2022), showing minor performance impacts. Only a few key-derivation methods are explicitly designed for Hybrid KEX, and most studies reuse the protocol's KDF to avoid compatibility issues. Key-derivation in hybrids has space for more research.

An observed convention in the primary studies is using only two KEX ingredients. This design is sufficient for security against attackers with different computing capabilities (non-quantum and quantum, under assumptions). In addition, this design incurs less overhead when applying it in a network protocol. Sikeridis et al. (SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020) found out that the overhead of this convention for hybrids is not significant, depending on the post-quantum algorithms used. On the other hand, few studies explore, in practice, the potential of Hybrids with more than two algorithms. There is only one evaluation of a Post-Quantum IKE (HEIDER, 2019), which shows an experiment (locally) using a KEX with three ingredients: `x25519` with `Kyber1024` and `NewHope1024` together. The hybrid cost observed was small using this configuration.

Section 1.3, pointed out research gaps in KEMTLS (SCHWABE; STEBILA; WIGGERS, 2020a) regarding the lack of hybrid support and evaluations; lack of implementations and evaluations for automatic issuing hybrid PQC in digital certificates; lack of quantum-safe contingency in current hybrids; and lack of tools aiding user-awareness about whether their web connections are secured with (hybrid) PQC. These gaps are addressed in Chapter 4.

3.2 EXPERIMENTAL METHODOLOGY

Given the research gaps identified in the SMS and aiming at better comprehending hybrid adoption as the PQC migration strategy, this work performs an empirical study on hybrids. First, the methodology starts with the design and development (or integration) of PQC and hybrids, using Go language based on the PQC algorithms provided by the OQS project (STEBILA; MOSCA, 2016). In this step, several design choices are made, such as the ones classified in Figure 10, and also in regards to what are the algorithms and their corresponding parameter sets to be integrated. Given the emphasis on network protocols, this work focuses on TLS and ACME protocols implementations.

After the development step, this work proceeds to experimentation and collecting empirical results. The experiments consider a TCP/IP network, where a peer (usually the client) connects to another peer of the protocol (i.e., the server). This work studies two types of network environments: simulated and realistic. The first allows modifying network parameters (such as latency). The latter does not control the network, creating a more realistic evaluation scenario. Both consider millisecond measurements, where averages can be shown in decimals. Depending on the protocol under evaluation, there are specific experiment design decisions. Therefore, the particular characteristics for each evaluation are detailed in Chapter 4.

4 CONTRIBUTIONS

This chapter is divided into four parts. First, Section 4.1 presents the TLS 1.3 Handshake Analyzer tool. Section 4.2 shows a performance evaluation of a post-quantum (hybrid) ACME protocol. Section 4.3 adds hybrids to the KEMTLS approach. Lastly, Section 4.4 describes the PKIELP innovative approach applied to TLS connections.

4.1 TLS 1.3 HANDSHAKE ANALYZER

Tracing back to 1986, a joint initiative between US agencies started the development of the Secure Data Network System (SDNS), which is considered one of the precursors of today's Transport Layer Security (TLS) protocol. The popularity of TLS started to grow with a different name, Secure Sockets Layer (SSL), first published in 1995. After 23 years, TLS 1.3 (RESCORLA, 2018) came to the public and currently it is the most-recent version of Internet's most-used security protocol.

TLS's continuous improvement has fixed security problems such as the Bleichenbacher attack (resolved in TLS 1.3). According to security experts, TLS 1.3 does not effectively safeguard user privacy and is exposed to the threat of quantum computers. Even if no known quantum computer is capable of breaching today's security parameters, an attacker may record TLS packets today to break TLS confidentiality in the future. In addition to these issues, non-expert users may need to be informed about whether their connections leak information (such as the websites they visit) or use quantum-vulnerable algorithms.

Fortunately, initiatives exist to address the quantum threat, such as the OQS project. However, users might want more knowledge about the security of their TLS connections. Moreover, PQC deployment approaches may harm network performance. The impact is mainly driven by the larger amount of data that must be sent due to PQC modes. Even though these modes are not yet standardized in TLS, they will likely be used soon.

This section introduces the TLS 1.3 Handshake Analyzer tool (GIRON; SCHARDONG; CUSTÓDIO, 2022). This tool is designed to help a variety of users to better understand or assess the security and performance of TLS connections. The tool's key features are listed below.

- Supports analysis of TLS 1.3 handshakes, with or without TLS keylog files.
- Allows users to check the presence of quantum-safe algorithms for their (early) protection.
- Displays what is the symmetric ciphersuites negotiated between the TLS peers. By using the CIPHERSUITE.INFO webservice (RUDOLPH; GRUNDMANN, 2022), the tool inspect and highlights insecure symmetric ciphersuites.
- Displays handshake statistics such as average handshake completion time and the size of handshake messages.

It is worth noting that there are numerous tools available to evaluate TLS connections. Some are open source, and the majority display security information (at least ciphersuites). The majority of the tools do not display performance data. Table 5 compares available solutions on a qualitative level.

Table 5 – Comparison of existing SSL/TLS tools.

Tool	Open source?	Shows security information?	Checks PQC Support?	Shows performance information?	User Interface
ssl-handshake	✓	✗	✗	✓	CLI
howsmyssl	✓	✓	✗	✗	Web
SSL Labs	✗	✓	✗	✗	Web
SSL Checker	✗	✓	✗	✗	Web
TLS Scanner	✗	✓	✗	✗	Web
SSL Tester	✗	✓	✗	✓(server-to-server)	Web
Observatory	✓	✓	✗	✗	Web/CLI
CryptCheck	✗	✓	✗	✗	Web
SSLChecker.com	✗	✓	✗	✗	Web
OQS-OpenSSL	✓	✓	✓	✓	CLI
TLS 1.3 Handshake Analyzer (This thesis' proposal)	✓	✓	✓	✓	Web/CLI

The OQS-OpenSSL fork (STEBILA; MOSCA, 2016) is a CLI tool that allows users to review security information and measure performance. The SSL Tester (WORMLY, 2022) has a friendly interface providing security and performance information by obtaining handshake times for each ciphersuite supported by a target server. It also displays the handshake sizes when different ciphersuites are selected. All timings, however, are collected from where the tool is hosted to the target website. As a result, it does not reflect the handshake time when the user's geographical location is considered. As detailed in the following section, TLS 1.3 Handshake Analyzer aims to fill those shortcomings.

4.1.1 Design

Figure 11 depicts a high-level overview of the instrument. The user interacts via a web interface, although a CLI is also offered. The tool begins parsing the input capture file (`.pcap` or `.pcapng` files), with or without the matching TLS keylog file, with the help of the `pyshark` module (GREEN, 2022). The tool retrieves handshake information and compares it to a local Object ID (OID) database, which contains a list of quantum-safe algorithm OIDs based on the OQS project. The OQS project is one of the most notorious PQC effort, which justifies integrating their proposed algorithm's OIDs.

TLS 1.3 Handshake Analyzer optionally checks for the Encrypted Client Hello (ECH) (RESCORLA et al., 2022) extension and requests an HTTP POST to CIPHERSUITE.INFO

(RUDOLPH; GRUNDMANN, 2022) to receive security information about the ciphersuites in use: recommended, secure, weak, or insecure. The distinction between weak and insecure is that the latter refers to a ciphersuite that is known to be vulnerable or "broken with minimal effort", whilst the former refers to an old ciphersuite that is no longer recommended by the IETF. After gathering all of the necessary data, the tool computes handshake statistics (performance data) and displays them to the user.

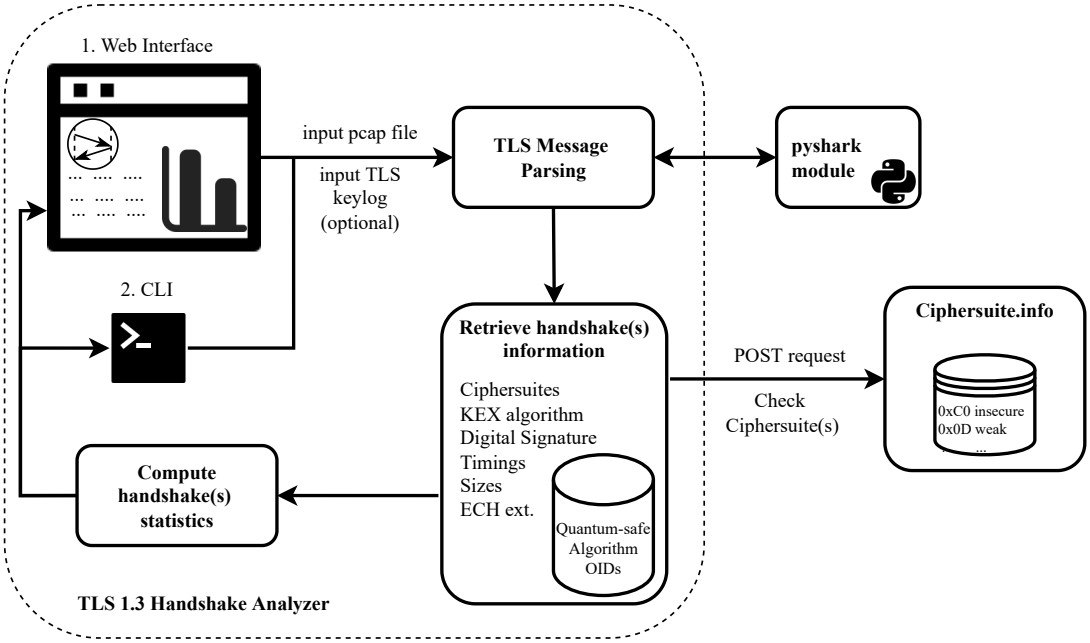

Figure 11 – High-level overview of the tool design

The tool parses each TLS 1.3 complete handshake in the capture file provided by the user. Each handshake message size is computed and used to retrieve the relevant information. For a friendly presentation to the user, the tool shows a plot of the relevant handshake message's size, followed by security and performance information, listed below.

- Security Information:
  - Cryptographic algorithms used: the handshake metadata contains the KEX, signature and symmetric algorithms negotiated between the parties. If the user provides the TLS keylog file, the tool is able to decrypt `CertificateVerify` and `Certificate` messages, recovering information the digital signature (and certificates) used for authentication. After retrieving what are the algorithms in use, the tool matches OIDs and names looking for quantum-safe algorithms, indicating if KEX and authentication used PQC, for each handshake found in the capture file.
  - Ciphersuite check: optionally, the tool checks if the selected ciphersuites are secure, based on the information provided by CIPHERSUITE.INFO (RUDOLPH; GRUNDMANN, 2022).

  - ECH check: optionally, the tool parses the handshake looking for the ECH extension message in `ClientHello`. The presence of the extension indicates that this privacy-preserving method is supported.

- Performance information: handshake timings and sizes. Timings are calculated based on the data in the capture file by subtracting the timestamps of the server's `Finished` message from the `ClientHello` message. When more than one handshake is discovered, average and standard deviation statistics are displayed. The tool calculates and graphs the sizes of all handshake messages (in bytes).

The TLS 1.3 Handshake Analyzer was created in Python (version 3.9 or higher is necessary). `Dash` (PLOTLY, 2022) is based on the `Flask` python's framework, so it simplifies creating a web interface for this tool. The tool can be run directly or built using a Docker file, making deployment and usage easier. Facilitating deployment and testing is critical for users like sysadmins who wish to check their connections or servers.

4.1.2 Demonstration

This tool's demonstration uses two capture files: handshake using PQC and hybrid mode, using the same algorithms: Kyber and Dilithium. Both captures were performed using the service provided by TEST.OPENQUANTUMSAFE.ORG web server, called here "OQS server", which allows one to benchmark post-quantum TLS connections. Table 6 summarizes the handshake characteristics, and Figure 12 shows the per-message size graph created by the proposed tool.

Table 6 – Handshake information retrieved from the sample capture file.

HS	Ciphersuite	KEX Algo.	Auth algo.	HS Time (ms)
1	TLS_AES256_GCM_SHA384	kyber512	dilithium2	306.9
2	TLS_AES256_GCM_SHA384	p256_kyber512	p256_dilithium2	308

The tool indicates to the user whether the handshake used post-quantum algorithms. In this demonstration, HS #1 is quantum-safe, and HS #2 uses the same algorithms but in hybrid mode. The tool shows that both connections are quantum-safe and give the performance based on the capture file with packet timestamps. Moreover, no ciphersuites in use were considered insecure. According to CIPHERSUITE.INFO, the ciphersuite `TLS_AES256_GCM_SHA384` is recommended for use.

The tool shows the handshake time (in milliseconds) and sizes regarding performance information. The sizes are close between the sampled handshakes. Note that both handshakes have similar performance, thus without significant drawbacks using the hybrid. However, the OQS server configuration does not consider some realistic TLS features, such as OCSP stapling. Moreover, the certificate chains used in practice differ from those used by the OQS server.

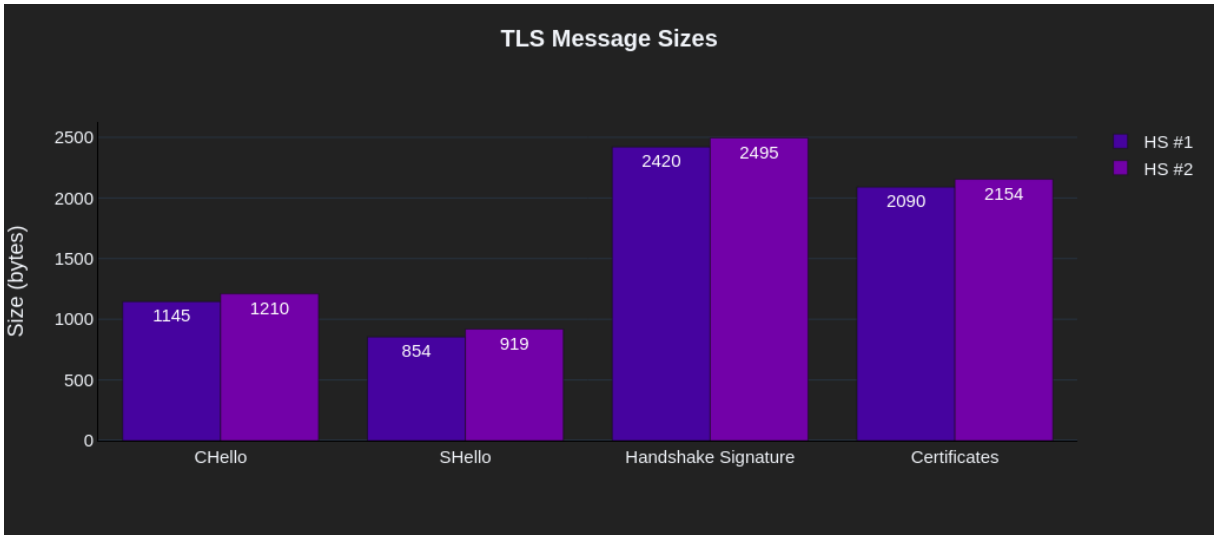

Figure 12 – Screenshot of the size graph generated by TLS 1.3 Handshake Analyzer tool.

The tool's repository provides additional sample capture files and a video demonstration of the tool (GIRON; SCHARDONG; CUSTÓDIO, 2022). The tool supports the analysis of captured files with multiple handshakes (and provides additional statistics). No average or standard deviation is computed since this demonstration's capture file has only two handshakes. The sample file used in this thesis is named `oqs-hybrid-cmp.pcapng`, available in the repository's `captures/` folder.

TLS 1.3 Handshake Analyzer allows web application managers to assess the impact of quantum-safe algorithms on the connection performance, not to mention assessing the presence of hybrid PQC for early protection. Moreover, users examining TLS packets with this tool can determine whether or not their connections employ quantum-safe techniques.

Due to the transparency of cryptography usage in TLS, users will likely be unaware of whether their connections have quantum-safe mechanisms. TLS 1.3 Handshake Analyzer is in the vanguard in this regard, considering quantum-safe awareness for users. The tool reaches different types of HTTPS/TLS users by providing CLI and Web interfaces.

4.2 POST-QUANTUM CRYPTOGRAPHY IN ACME

ACME is considered an Internet security enabler since it provides facilities to manage certificates for web servers. One problem dealt with by this thesis is that ACME is not prepared yet for a post-quantum world. Therefore, this section presents a performance evaluation after migrating ACME to quantum-safe algorithms. The migration considers the ACME protocol, where the communication channel and messages use PQC, and the ACME's output, i.e., quantum-safe digital certificates for the clients.

4.2.1 Quantum threats in ACME

Section 2.4 describes the requirements for using ACME. First, ACME needs a confidential channel for exchanging messages. TLS is the standard option; thus, a quantum-safe TLS must be used in the ACME channel. In order to do so, the ACME server host must be previously provisioned with a quantum-safe certificate and with the corresponding PQC algorithm implementations, both for signing TLS handshakes and for KEX. While quantum-safe signing avoids quantum attacks on authentication, quantum-safe KEX protects the ACME channel against *record-now-decrypt-later* attacks. On the other hand, when ACME servers validate challenges (the validation channel), they might not need quantum-safe algorithms, for instance, when using the HTTP-01 challenge. The HTTP-01 is executed over plain HTTP. Therefore, no need for PQC in that case.

The ACME channel between the client and server should be quantum-safe, meaning both implement PQC in TLS. Moreover, ACME clients also deal with two key pairs: the account keys and the to-be-issued certificate keys. Clients use account keys to sign requests and require a Certificate-Signing Request (CSR) when requesting a new certificate. Note that CSRs can be created outside of ACME but still require a different key pair from the account keys. In order to avoid quantum attacks, those keys should be created with PQC, as well as the keys on the server side (e.g., Issuer CA keys).

In summary, using a quantum-safe TLS avoids quantum attacks, and using quantum-safe keys in ACME allows the issuance of quantum-safe certificates for clients (e.g., web applications), thus helping to build a quantum-safe Internet. Without quantum-safe algorithms, however, ACME faces the quantum threat depicted in Figure 13. The lack of a quantum-safe TLS channel and keys in ACME enables a record-now-decrypt-later attack on sensitive information, such as account data and authorizations. Having account keys broken by quantum computers, the attacker then signs a request to the server on behalf of the client. The server will issue the certificate to the attacker if the client's authorization is still valid. Having the authorization information is not enough to issue a certificate in behalf of the client. However, a quantum attacker could recover the client's private key and then attacking ACME becomes possible (if classical cryptography is in use).

The attack is theoretically possible by exploiting valid authorizations and due to the use of classical cryptography in ACME components (TLS channel and account keys). In practice, authorizations can be valid for 30 days. However, RFC 8555 does not impose an expiration time, leaving it up to implementations to deactivate authorizations (BARNES et al., 2019). If not deactivated or within a valid period, quantum attackers can exploit authorizations issued by servers. Therefore such information should be protected using quantum-safe algorithms. In order to be successful, ACME's PQC migration requires not only PQC algorithms but a risk mitigation procedure: quantum-safe ACME servers should deactivate account authorizations created with classical cryptography. Such a procedure is necessary since ACME servers have no guarantee (or log) of the TLS connection established with ACME clients in the past.

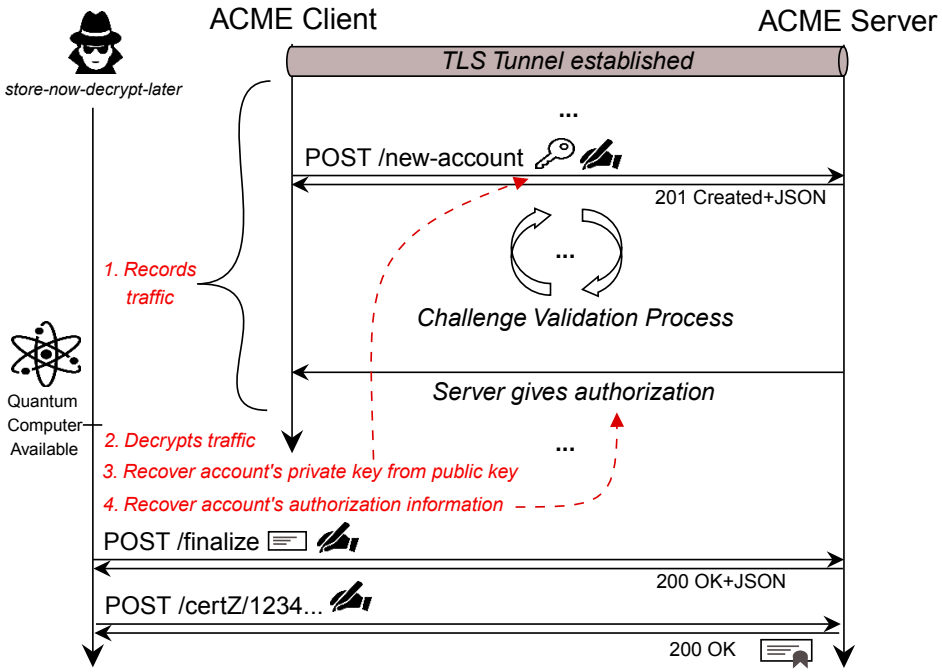

Figure 13 – Unauthorized issuance of a certificate with the help of a quantum computer.

4.2.2 PQC impacts in ACME

The methodology used in this thesis to evaluate the PQC impacts in ACME is described below.

- ACME implementations: several open source software for ACME clients and servers exist. Pebble is an ACME test server, and LEGO provides a library and a client implementation, both endorsed by Let's Encrypt (Let's Encrypt, 2022) and written in Go programming language. Together with Go-JOSE and Go standard library, they can be used for experimenting with ACME.

- PQC source implementations: the OQS project `liboqs` provided the PQC algorithms (STEBILA; MOSCA, 2016). Integrating Golang-based software requires a binding, in this case, provided by liboqs-go (PROJECT, 2022).

- PQC algorithms: this thesis' ACME implementation is integrated with PQC following NIST recommendations. **Kyber** is used for Key Exchange at the TLS level, using security level 1 parameters (i.e., `Kyber512`). For authentication, it uses **Dilithium**, **Falcon** and **Sphincs+**, where the same algorithm and security level parameters are used in all required cryptographic objects. Namely: ACME client account keys and CSR; ACME server digital certificate (TLS level); issued certificates; and the certificate chains of issued certificates (Root CA certificate and Intermediate CA certificate). For simplicity, we did not change Pebble's certificate chain for TLS. We only alter Pebble's TLS chain to

use PQC algorithms without adding a new Intermediate CA certificate. A restriction is that Sphincs+ is only used for the Root CA certificate, based on the following parameters: `SHAKE` for the hash function, "s" for compact signatures and improved verification timings, and "simple" for performance.

- PQC adoption strategy: hybrids and PQC-only deployment. For hybrids, the integration happens with NIST P-curves, namely `P256`, combined by concatenating with Kyber, Dilithium, and Falcon cryptographic objects. The implementation uses concatenation for the classical and PQC cryptographic objects (public keys and signatures) for simplicity. The hybrid mode is referred to by the 'H' letter (e.g., "Dilithium H.").

The metrics used for the performance experiments are based on timings and byte costs. They are described below:

- At the server side, the number of successful requests per second, processing time and memory peak usage. For the number of successful requests, the client creates 1024 threads to make several issuance requests to the server's `/finalize` endpoint (during 6 minutes). Each thread simulates a different client sending CSRs, thus increasing the server's load when issuing certificates.

- Timings: At the client, certificate issuance time and renewal time after 500 executions. The issuance time encompasses the renewal time since renewals perform the same requests except for the account creation requests. Therefore, the renewal time counts the time from the `/new-order` POST request until the client receives the certificate (see Figure 8).

- Byte costs: this metric is analyzed on a per-algorithm and per-message basis by summing up the size of each cryptographic object used in each ACME message.

The experiments were conducted using two geographically distant Google N2 Virtual Machines (VMs) having 157 ms of average RTT. Their configuration is the same (8 GB memory, two vCPUs), except that the VM executing the ACME client is hosted at Osasco, São Paulo, Brazil. In contrast, the ACME server location is based on one of Let's Encrypt's data centers (Salt Lake City, Utah, USA). The source code for the ACME implementations and test scripts are publicly available[1].

Figure 14 shows the PQC impacts on the server side. Given the possibility of clients using an Out-of-Band (OOB) CSR, the experiment analyzes two approaches: "CSR-on-the-fly" and "pre-computed CSR" tests. The first incurs in each client thread generating a new CSR, thus including key generation and signing computational times. As a result, clients are delayed, resulting in fewer successful server requests. Using a pre-computed CSR reduces the PQC impacts but is still noticeable.

[1] https://github.com/AAGiron/acme-newchallenge

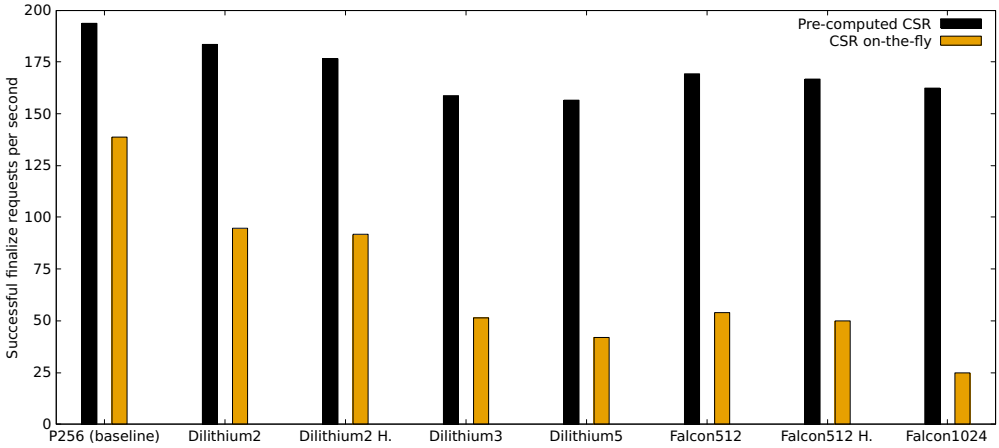

Figure 14 – Load test experiment with and without CSR cryptographic operations.

Table 7 shows the average and 95th percentile of computational timings (rounded to one decimal), measured for each `/finalize` request during the load test. The measurements exclude network time, only the server's load is considered. The measurement includes allocated memory peak, in MiB. Dilitihum2 instances (PQC and hybrid) incurred a higher load, both memory and computing costs. Falcon key-generation bottleneck penalizes the client, which explains the smaller server's metrics in contrast to Figure 14.

Table 7 – Server's metrics during the load test.

Algorithm Instance	**Processing time (ms)**		**Allocated Memory Peak (MiB)**
	Avg.	**95-th percentile**	
P256 (baseline)	8.3	49.4	336
Dilithium2	45.2	240.3	1166
Dilithium2 H.	47.7	277.0	1132
Falcon512	16.6	46.7	597
Falcon512 H.	19.7	59.0	668

When comparing hybrids to PQC-only deployment, the average computational times show non-significant hybrid penalties. The penalties increase at the 95th percentile but are limited to 13 ms in `Falcon512 H.` and 37 ms in `Dilithium2 H`. Given that both hybrid PQC algorithms use the identical classical counterpart (P256), the penalties between Falcon and Dilithium should be equivalent. However, load testing suffers from variations on the host system (VMs in this case), such as processor scheduling. Nevertheless, such a practical experiment allows one to expect minor penalties on average and limited penalties on the 95th percentile.

Additionally, the load testing shows insights into the Issuer CA's perspective. The PQC impacts are noticeable far from baseline, as shown in Figure 14 and Table 7. At the Issuer CA, fewer requests successfully handled by the server means fewer certificates being generated and issued. As a result, the PQC migration for ACME as-is can impact CA's business model requirements. A threat to validity is that the experiment does not measure how many certificates per second can be issued. However, the load test is still representative since it includes handling

several signed requests by the server, the CSR generation (by the client threads) and verification (by the server).

As described in Section 2.4, the issuance flow has several digitally-signed requests between peers. Using PQC signatures in such requests would increase protocol communication costs and impact the interaction between those peers. This thesis proposes a new ACME challenge, *PQ-Transition Challenge*. The new challenge aims to reduce communication costs and speed up issuance. Therefore, the byte cost analysis compares standard ACME sizes to this thesis proposal (presented at the end of the next section).

4.2.3 Faster Issuance with the PQ-Transition Challenge

The alternate ACME challenge proposed in this thesis aims to ease the transition to PQC (not necessarily using hybrids). The proposal's main requirement is that the ACME client already has a previously issued certificate, typically with classical cryptography. Having a certificate means that these ACME peers have a relationship that could be used for an optimized PQC transition: from classical to (hybrid) PQC. The *PQ-Transition Challenge* is depicted in Figure 15.

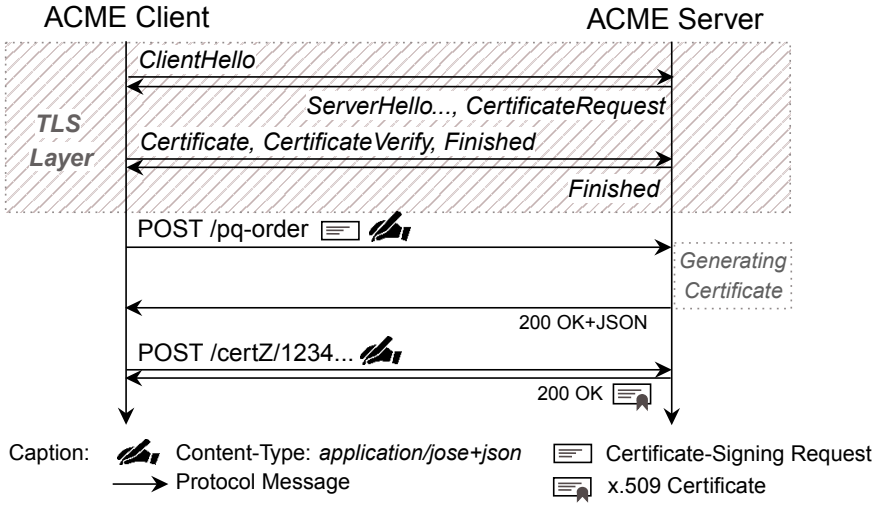

Figure 15 – Proposed ACME Challenge.

The *PQ-Transition Challenge* provides an alternative to the `/new-order` ACME server standard endpoint, called `/pq-order`. As with the traditional `/finalize` endpoint, the new endpoint (on the server) expects a CSR in an HTTP POST message. The primary distinction is that it necessitates a mutually authenticated TLS handshake. Mutual authentication implies that the ACME client authenticates directly in the TLS layer, demonstrating ownership of the certificate's private key. Control of the certificate that verifies an Internet domain name demonstrates control over the web server under that domain by transitivity. If the client successfully authenticates to the server, the server can generate a new (PQC or hybrid) certificate and respond with a URL where the certificate can be downloaded.

The fact that the ACME client already possesses a certificate plays a crucial role in this approach. Let $C_{classic-cert}$ be the certificate that the client is willing to use in the TLS authentication layer, and $C_{pqc-cert}$ the certificate the client requests. If $C_{classic-cert}$ was issued by the same ACME server where $C_{pqc-cert}$ will be requested, then the peer trust relationship is already established. The ACME server will trust $C_{classic-cert}$ (in the pre-quantum scenario), so additional configuration or protocol messages are unnecessary. Comparatively, it removes (at least) 4 signed requests from the ACME flow and replace the challenge by TLS client authentication using the $C_{classic-cert}$.

Figure 16 shows the issuance and renewal times for ACME with PQC compared to our proposed challenge. The bars correspond to average timings in standard ACME protocol compared to those obtained using the PQ-Transition challenge. The graph also includes standard deviation information (above the bars) in standard ACME. All standard deviations obtained from the PQ-Transition challenge executions are below 10 ms, whereas in standard ACME, it reaches 1.4 seconds. All bars are below the baseline standard deviation (using P256), which suggests no PQC transition impact in the timings perceived by the ACME client.

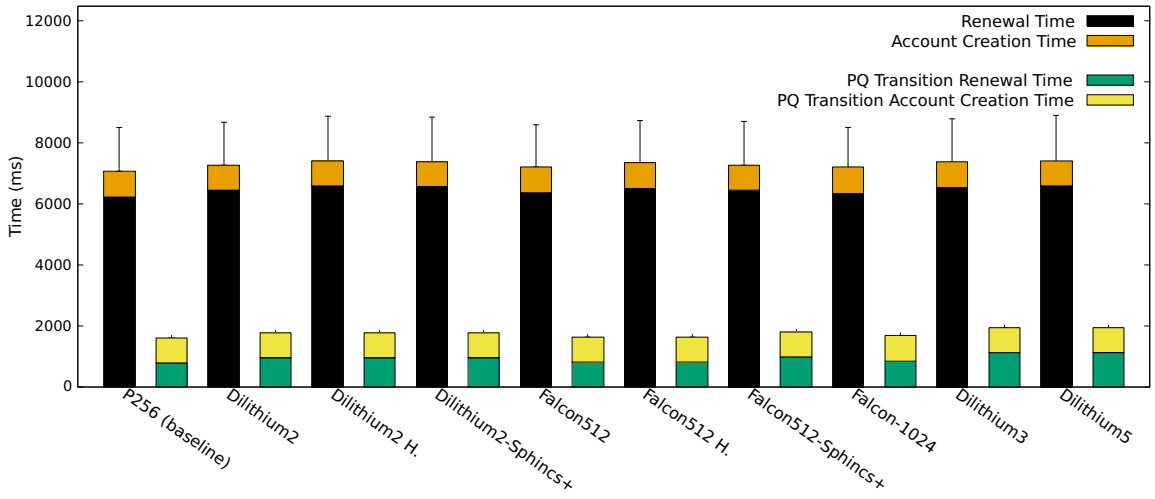

Figure 16 – Issuance and renewal timings for different PQC algorithm instantiations. Note: Issuance time is the sum of Account Creation and Renewal time.

Given the increased sizes in PQC, one can expect a significant slowdown in PQC instances when analyzing the issuance and renewal times. However, this was different. Even though they impose additional bytes, this byte amount is divided between several request messages (please refer to Figure 8). Therefore, keeping the sizes beneath the TCP/IP network stack limits, such as (such as the TCP window size) should not impose additional RTTs on the communication. Additionally, the configuration allows not to transmit the Sphincs+ certificate, saving bytes.

The obtained issuance times do not depend on RTTs but also on the (variable) number of requests and the client's waiting times. For example, clients must wait until the server generates the certificate. This wait affects the timings, explaining why the results show an average issuance time for PQC close to the baseline with classical cryptography.

In summary, the average issuance time is near 7.5 seconds to issue a classical, PQC, or hybrid certificate. Sphincs+ (at the root CA certificate) does not influence the timings, so that configuration is also viable. Once again, no significant penalties for using hybrids, thus encouraging its use as the PQC adoption strategy.

Although the PQC influences on timing are not visible to the clients, the PQ-Transition challenge is considerably faster. Compared to the commonly used HTTP challenge, the issuance times are 4.22x faster on average. Renewals are similarly faster: without account creation, the renewals take under 1 second (on average), independent of the PQC algorithm or mode used for the new certificate. However, the PQ-Transition challenge is not designed to replace the existing ACME challenges. Therefore, for the scenarios where the proposed challenge is unsuitable, ACME clients can fall back to the standard challenges (e.g., HTTP-01).

Table 8 compares the byte costs of ACME without modification and using the PQ-Transition challenge proposal. The proposal simplifies ACME; thus, some requests are not necessary anymore (last column in Table 8). When compared to the original ACME message flow, the PQ-Transition challenge saves $35,39\%$ and $32,14\%$ when analyzing `Dilithium2` and `Falcon-512` example instantiations.

Table 8 – Comparison of sizes of ACME client requests, sampled from a pcapng capture file. Note: "Total (ACME)" excludes repeated requests; therefore, in practice, more bytes are transmitted.

Request	Dilithium2 size (bytes)	Dilithium2 H. (size penalty)	Falcon-512 size (bytes)	Falcon-512 H. (size penalty)	PQ-Transition challenge?
GET /dir	223	-	223	-	✓
HEAD /new-nonce	207	-	207	-	✓
POST /new-account	6097	4.3%	3010	8.7%	✓
POST /new-order	3747	2.7%	1399	7.4%	✗
POST /pq-order	10558	3.3%	4336	7.9%	✓
POST /authZ	3776	2.8%	1423	7.3%	✗
POST /challZ	3779	2.8%	1425	7.4%	✗
POST /finalize	10648	3.3%	4417	8.2%	✗
POST-AS-GET /certZ	3713	2.8%	1361	7.5%	✓
Total (ACME*)	**32190**	**3.2%**	**13465**	**7.7%**	-
Total (PQ-Transition Challenge)	**20798**	**3.4%**	**9137**	**7.8%**	-

Table 8 also shows the hybrid penalties in terms of byte costs. When selecting hybrid modes, one can expect less than 10% increase in payload sizes, more specifically, 3.2% with `Dilithium2 H.` and 7.7% with `Falcon-512 H.`, considering standard ACME messages. Moreover, the penalties of selecting hybrids in the PQ-Transition challenge proposal are similar: 3.4% and 7.8% for `Dilithium2 H.` and `Falcon-512 H.`, respectively.

Finally, it should be highlighted that the suggested challenge can be improved by making additional changes to the TLS layer. For example, reducing the TLS mutual authentication payload size is possible by using caching mechanisms (RFC 7924 (SANTESSON; TSCHOFENIG, 2016)). As a result, smaller TLS messages avoids additional RTTs which can improve performance.

4.3 ADDING HYBRIDS TO KEMTLS

A different approach for the PQC adoption in TLS is called KEMTLS (SCHWABE; STEBILA; WIGGERS, 2020a). It uses KEMs for end-entity authentication as they might be smaller than PQC signatures, aiming for better protocol performance. KEMTLS was evaluated with different PQC algorithms in practical experiments. Afterwards, authors proposed KEMTLS-PDK (SCHWABE; STEBILA; WIGGERS, 2021), which optimizes KEMTLS in specific scenarios. On the other hand, one gap observed is that neither KEMTLS nor KEMTLS-PDK were evaluated in the hybrid mode. This work fills this gap by providing an implementation and extensive evaluation of a hybrid version of KEMTLS and KEMTLS-PDK.

4.3.1 KEMTLS and its variants

KEMTLS is built utilizing PQC KEMs, which are generally lower in size than Digital Signatures. The design incorporates an ephemeral KEM as well as a second KEM for long-term use. KEMs replace the digital signature in every (complete) TLS 1.3 handshake in this manner. TLS handshake is modified in three stages: ephemeral key exchange using KEMs, implicitly authenticated key exchange using KEMs, and confirmation with explicit authentication. After one cycle, during the second phase, when the server is implicitly authenticated, the client can submit application data. However, only after the third phase, when the server is explicitly authenticated, is the handshake completed in two rounds.

KEMTLS-PDK is a KEMTLS version with pre-distributed keys, making it appropriate for cases in which the client knows the server's long-term public key before to communication (SCHWABE; STEBILA; WIGGERS, 2021). As a result, in his first round of messages, the client can execute an encapsulation against the server's long-term public key, boosting performance. The KEMTLS-PDK's 1-RTT time makes it more competitive with TLS in terms of time to complete the handshake completely. KEMTLS, on the other hand, can be utilized only when the server's public key is known (e.g., cached).

Celi et al. (CELI et al., 2021) supplied a Go-lang implementation and evaluation of KEMTLS, KEMTLS-PDK, and PQTLS through a branch of the Go Standard Library. In this implementation, the server's certificate includes a delegated credential and uses the PQC signature algorithm for PQTLS or the PQC KEM algorithm for KEMTLS and KEMTLS-PDK. This solution decouples the handshake authentication algorithm from the authentication algorithms used in the certificate chain, simplifying the cryptographic agility principle in protocol design.

The hybrid PQC design is not considered in any of the suggested KEMTLS variations. A Hybrid PQC architecture consists of (at least) two algorithms, one PQC and one classical, with the security features holding as long as one of the elements remains unbroken (BINDEL et al., 2019). Some works, on the other hand, evaluate hybrid designs in PQTLS (PAQUIN; STEBILA; TAMVADA, 2020; SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020), but none use KEMTLS as the PQC adoption approach.

4.3.2 Hybrid KEMTLS Design

Since KEMTLS initially follows a PQC-only deployment strategy, this section aims to modify KEMTLS as minimally as possible to add the hybrid mode. Therefore, classical KEMs must be added to execute parallel to the PQC KEMs used in the original KEMTLS. Figure 17 shows the proposed design, considering the server-only authentication scenario. It shows the cryptographic operations by the client and server (squared boxes), the required protocol messages (with arrows) and the key derivation steps (in the middle) performed by both peers.

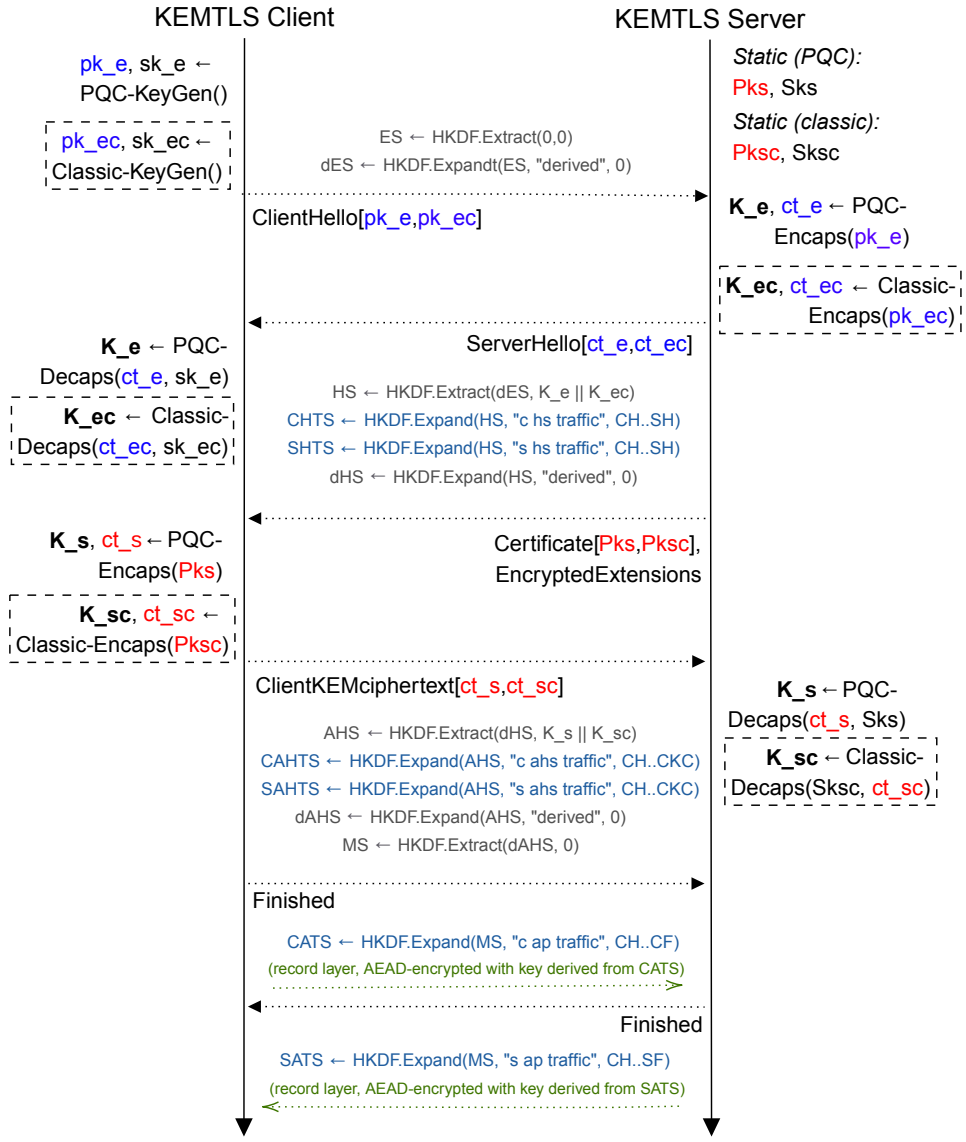

Figure 17 – Proposed Hybrid KEMTLS Handshake (Server-only authentication).

The KEMTLS protocol is based on two KEM keypairs, an ephemeral and a static one. Therefore, to "hybridize" it, classic KEM keypairs are needed. The Hybrid KEMTLS are instantiated with the following classic KEM's: `KEM_P256_HKDF_SHA256`, a KEM using `P256` curve and HKDF with `SHA-256`; `KEM_P384_HKDF_SHA384`, a KEM using `P384` curve and HKDF with `SHA-384`; and `KEM_P521_HKDF_SHA512`, a KEM using `P521` curve and HKDF

with SHA-512. All classical KEMs are obtained from CIRCL library (FAZ-HERNáNDEZ; KWIATKOWSKI, 2019).

The Hybrid KEMTLS naming convention in the implementation is composed of two parts. The first part is the classic KEM curve (p256, p384 or p521), and the second part of is the PQC KEM algorithm name (and parameter set). For example, P256_Kyber512 stands for a Hybrid KEMTLS instance with the KEM_P256_HKDF_SHA256 classic KEM and Kyber512 PQC KEM. For an extensive evaluation, the implementation supports all NIST security levels available for the PQC implementations (levels one to five).

The Hybrid KEMTLS protocol works similarly to the KEMTLS, but now each cryptographic operation has its classic counterpart, as the square dotted boxes highlight it. It starts by clients generating key pairs for both the PQC and Classic ephemeral KEMs. The resultant public keys *pk_e* and *pk_ec* are concatenated and transmitted to the server through the key_share extension from the ClientHello TLS message. The server performs an encapsulation against each key provided by the client, replying the corresponding ciphertexts concatenated in the ServerHello.

The server authenticates by means of a long-term KEM key. This key is comprised in the Certificate message, but in the proposed hybrid now the certificate has two static KEM public keys concatenated: from a classic KEM and from a PQC KEM. The server sends its certificate so that the client can encapsulate using long-term keys. Besides, the client decapsulates the ephemeral ciphertexts to obtain *K_e* and *K_s*. The client replies to the server ciphertexts *ct_s* and *ct_sc*, which are concatenated. Lastly, the server decapsulates the received ciphertexts obtaining the same static shared secrets *K_s* and *K_sc* as the client. At this point, both the client and the server can derive the symmetric keys. The server can send the Finished message back to the client, followed by a reply from the client, finishing the Hybrid KEMTLS Handshake. Further implementation details can be found in the full-version paper (GIRON et al., 2022).

The cryptographic combiner is an essential part of the hybrid design. This proposal's design instantiates the dualPRF combiner, originally proposed by Bindel et al. (BINDEL et al., 2019). The difference is that hybrid KEMTLS as designed here requires two combinations: the ephemeral and static (long-term) part. The shared secrets of both ephemeral (*K_e*,*K_ec*) and static parts (*K_s*, *K_sc*) are combined using concatenation (prior to KDF). This design decision is based on the NIST standards for key derivation that allows such a construction (BARKER; CHEN; DAVIS, 2020). For a security analysis, please refer to the full-version paper.

Since the hybrid KEMTLS makes each *K* contribute to generating the traffic keys, requiring both parties (i.e., client and server) to support all the algorithms of the selected hybrid mode. However, this is necessary for an extensive evaluation, but it should not be a significant issue in practice since we add one algorithm (pre-quantum) cryptography to KEMTLS, which is commonly used nowadays. As KEMTLS, our design allows us to select the same algorithm for KEX and authentication, simplifying the protocol's negotiation procedure.

The hybrid KEMTLS implementation is an adaptation from a Golang repository from Celi et al. (CELI et al., 2021). It "hybridizes" their KEMTLS and PQTLS implementation

and a new testbed to support different PQC algorithms. Moreover, Celi et al. focus on Delegated Credentials (DC) as a migration strategy, which is not used for this work's experiments. The liboqs-go wrapper (PROJECT, 2022) from the OQS project was used to integrate all available PQC algorithms. The tests are based on the NIST Round 3 Finalists regarding the KEM algorithms: Kyber, NTRU and Saber.

In the context of hybrid adoption and aiming at a fair comparison, our evaluation has the following characteristics and metrics:

1. Practical scenario: uses geographically distant connections through two Google N2 VMs (Intel(R) Xeon(R) 2.80GHz CPU and 8 GiB RAM). The TLS client uses a VM located at `southamerica-east1-a` (São Paulo, Brazil), and the server is at `europe-central2-a` (Warsaw, Poland). The average network latency is 108ms.

2. Simulated scenario: uses NetEm (HEMMINGER, 2011) with two parameters: latencies (2ms, 10ms, 100ms, 300ms); and packet loss probabilities (1, 2, 3, 5%). The simulations were executed in an Intel i5-8250U 1.60GHz, with turbo boost technology disabled. The simulated scenario allows a fine-grained analysis and minimizes standard deviations inherent in practical networks and VMs.

3. HTTPS/TLS Load test scenario: this practical test gives insight into the performance of a web server using the proposed approach. It uses 128, 256, 512, and 1024 concurrent client threads and executes for 6 minutes for each hybrid instantiation for comparison.

4. Evaluation metrics: the result statistics are based on 1000 executions (handshakes) of the protocol, and the metrics are: (i) Handshake Completion Time, at the client, it initiates from the start of the protocol until it receives the `Finished` message; (ii) *Time-to-Send-App-Data*: the required time for the client to send application data to the server, since both KEMTLS and TLS clients can send application data in one RTT (for a fair timing comparison); (iii) Hybrid Penalty: if H is the measured time in a hybrid instantiation and P is the PQC-only time, we subtract $H - P$, resulting in the hybrid penalty; and (iv) HTTPS/TLS request successes and failures, the number of successful requests and failures for the server's load test.

4.3.3 Results and Discussion

The Hybrid Penalty, or the cost of adding a hybrid mode over the PQC-only configuration, is the first measure considered in the proposed hybrid KEMTLS. Figures 18 to 22 show the box plots for the KEMTLS timings considering server-only authentication based on the geographically distant connections scenario. Figures 19 to 23 show the KEMTLS-PDK results. They include the results for the three NIST security levels (L1, L3, and L5).

In this case, the hybrid penalties (i.e., the cost of adopting our hybrid) have distinct outcomes. The box plots for security level 1 exhibit overlapping timings, indicating that the

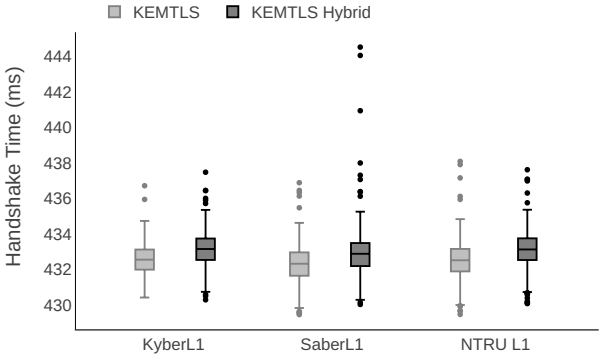

Figure 18 – PQC-Only and Hybrids (L1)

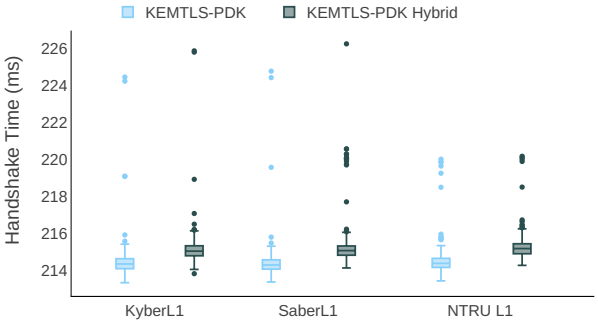

Figure 19 – PDK Instantiations (L1)

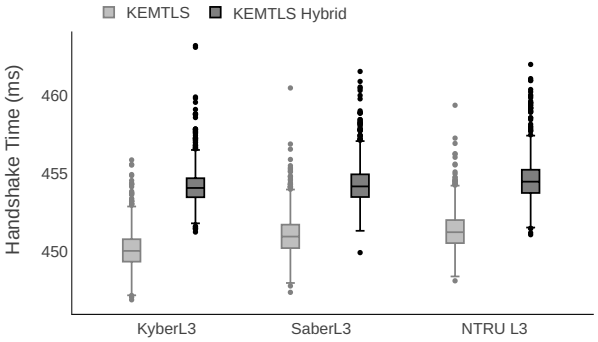

Figure 20 – PQC-Only and Hybrids (L3)

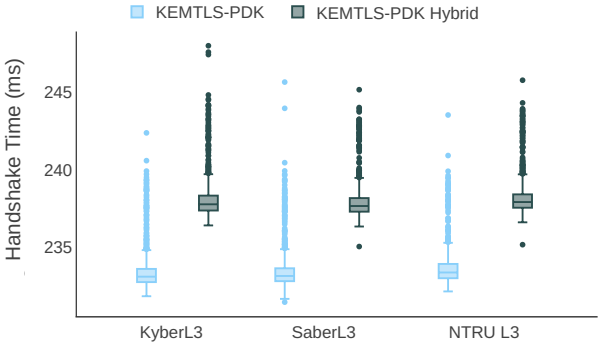

Figure 21 – PDK Instantiations (L3)

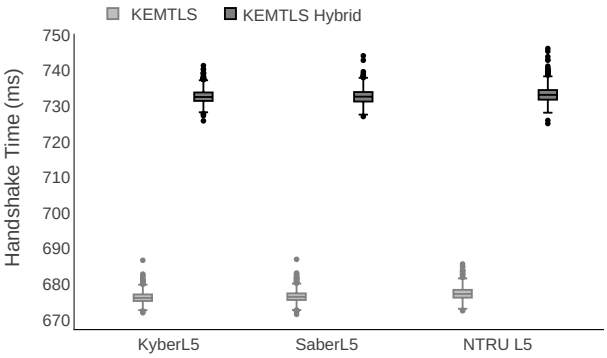

Figure 22 – PQC-Only and Hybrids (L5)

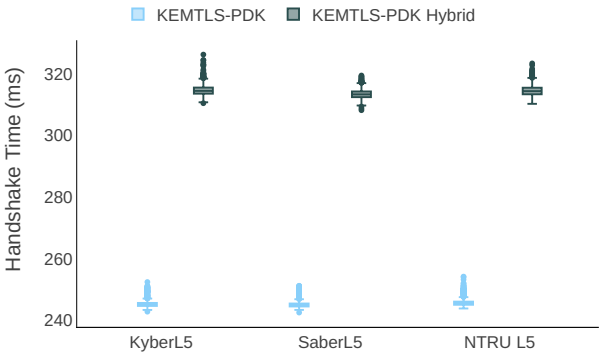

Figure 23 – PDK Instantiations (L5)

penalties, independent of the tested algorithms, are modest. However, as the security level is increased, the boxes no longer overlap, increasing the penalty. As seen in Figure 22, the hybrid penalties are substantially more significant at level 5 than the other levels caused by the classical method. When compared to KEMTLS, KEMTLS-PDK acquired improved timings due to a lower number of RTTs. In such a geographically distant connection, the RTT conceals most cryptographic operation timings. When examining hybrid penalties, KEMTLS and KEMTLS-PDK produce similar behavior. These findings encourage hybrid versions, but not at higher security levels.

The simulated environment allows controlling the effect of network variations. Table 9 highlights the hybrid penalties in the simulated scenario using the average handshake time (HS) metric. The penalties are minor and concealed by the network latency. For example,

configuring 1 ms of link latency means that the client and the server will be delayed by 2 ms, and since KEMTLS requires two round trips to complete the handshake, it doubles this number, reaching 4 ms. Therefore, increasing the latency has a greater impact than the hybrid penalty. Similar behavior happens in KEMTLS mutual authentication (Table 10), requiring three instead of two round trips. Moreover, minor penalties were also found when comparing different packet loss probabilities for hybrids. Those experiment results are available in the full version, corroborating the results presented here.

Table 9 – Average Handshake time (HS, in ms) for PQC-Only and Hybrid KEMTLS under different simulated latencies ('L' means security level and 'H' if in hybrid mode).

Algorithm and Security Level	**Latency: 1 ms**			**Latency: 5 ms**			**Latency: 50 ms**			**Latency: 150 ms**		
	HS Time	**Penalty**	**St. Dev.**	**HS Time**	**Penalty**	**St. Dev.**	**HS Time**	**Penalty**	**St. Dev.**	**HS Time**	**Penalty**	**St. Dev.**
KyberL1	6.0	-	0.4	22.3	-	0.3	202.8	-	0.2	602.9	-	0.2
KyberL1 H.	7.0	**1.0**	0.4	23.2	**0.9**	0.3	203.6	**0.9**	0.3	603.7	**0.8**	0.4
KyberL3	38.5	-	0.8	54.8	-	0.8	236.3	-	1.0	636.6	-	1.0
KyberL3 H.	46.8	**8.3**	0.9	62.9	**8.1**	2.3	243.2	**6.9**	1.2	643.9	**7.3**	1.6
KyberL5	63.0	-	0.8	78.4	-	0.8	261.1	-	6.0	659.9	-	1.0
KyberL5 H.	194.6	**131.6**	2.4	211.4	**133.0**	3.7	393.0	**132.0**	4.5	791.6	**131.7**	3.2
SaberL1	6.1	-	0.5	22.3	-	0.3	202.8	-	0.2	602.9	-	0.3
SaberL1 H.	7.1	**1.0**	0.5	23.3	**0.9**	0.4	203.7	**0.9**	0.4	603.8	**0.8**	0.4
SaberL3	38.9	-	0.7	55.5	-	0.8	236.0	-	1.0	637.1	-	2.2
SaberL3 H.	46.1	**7.2**	0.8	62.7	**7.1**	0.8	243.9	**7.9**	1.0	644.6	**7.5**	1.8
SaberL5	62.6	-	1.7	79.0	-	0.8	260.3	-	0.9	661.1	-	2.5
SaberL5 H.	196.5	**133.8**	2.4	212.2	**133.2**	3.2	391.9	**131.6**	3.3	792.7	**131.6**	3.4
NTRU L1	6.1	-	0.3	22.3	-	0.3	202.8	-	0.2	602.9	-	0.2
NTRU L1 H.	7.1	**1.0**	0.3	23.3	**1.0**	0.4	203.6	**0.8**	0.3	603.7	**0.8**	0.3
NTRU L3	39.2	-	0.8	55.8	-	0.8	236.5	-	2.4	636.9	-	1.0
NTRU L3 H.	46.5	**7.3**	0.8	63.0	**7.2**	0.8	243.6	**7.0**	1.0	644.1	**7.2**	1.0
NTRU L5	62.7	-	0.8	78.7	-	0.8	259.4	-	1.0	659.3	-	1.6
NTRU L5 H.	197.1	**134.4**	2.8	211.1	**132.3**	4.5	393.7	**134.3**	2.9	793.3	**134.0**	3.3

Regarding mutual authentication, the main change is the `CertificateRequest` message sent by the server, which triggers the client to send its certificate containing two KEM public keys (from a PQC and classical algorithm). The server encapsulates under the client's keys and replies the `ServerKEMCiphertext` message. The new shared secrets K_c (PQC) and K_cc (classical) are used in the key derivation process.

On the other hand, about the overall performance, the number of round-trips of mutual authentication design is different, which adds one RTT to the network latency. Using the 1 ms simulated latency as example, in server-only authentication the link latency will be 2 ms (client side and server side). Since KEMTLS has two RTTs, the network latency will be 4 ms, and in the mutual authentication case it will be 6 ms (three RTTs). KyberL1 H. performance is 11 ms in average (Table 10), which means 5 ms of computational cost and 6 ms of network latency (3 RTTs times 2 ms). Now looking at 150 ms simulated latency, KyberL1 H. gets an average handshake time of 906.3 ms, which means approximately 6 ms of computational cost and 900 ms of network latency (3 RTTs times 300 ms). Given a standard deviation near to 1 ms, we can observe very close computational cost and, consequently, the penalties are

Table 10 – Average handshake time (HS, in ms) for PQC-only and Hybrid KEMTLS in mutually-authenticated connections.

Algorithm	Latency: 1ms			Latency: 5ms			Latency: 50ms			Latency: 150ms		
	HS Time	Penalty	St. Dev.	HS Time	Penalty	St. Dev.	HS Time	Penalty	St. Dev.	HS Time	Penalty	St. Dev.
KyberL1	9.5		0.7	33.8		0.7	304.8		0.6	905.0		0.9
KyberL1 H.	11.0	**1.5**	0.8	35.2	**1.4**	0.8	306.1	**1.3**	0.8	906.3	**1.3**	0.9
KyberL3	79.4		1.5	102.7		1.4	374.3		1.9	974.2		1.8
KyberL3 H.	89.6	**10.2**	1.3	114.8	**12.1**	1.3	384.0	**9.7**	1.5	984.6	**10.4**	1.3
KyberL5	128.7		1.7	153.2		2.5	422.9		2.3	1021.7		2.5
KyberL5 H.	319.6	**190.9**	3.3	344.0	**190.9**	3.1	615.5	**192.6**	3.3	1216.4	**194.7**	6.1
SaberL1	9.5		0.7	33.8		0.7	304.9		0.9	905.0		0.7
SaberL1 H.	11.0	**1.5**	0.8	35.2	**1.4**	0.8	306.2	**1.3**	0.8	906.3	**1.3**	0.9
SaberL3	79.0		1.4	103.3		1.6	374.0		1.8	973.6		1.5
SaberL3 H.	89.4	**10.4**	1.3	114.1	**10.8**	1.4	384.3	**10.3**	1.5	984.5	**10.9**	1.2
SaberL5	128.9		1.7	152.6		1.7	422.8		1.9	1023.3		1.8
SaberL5 H.	320.8	**192.0**	4.1	344.0	**191.5**	3.9	616.0	**193.2**	3.4	1214.0	**190.7**	3.4
NTRU L1	9.5		0.8	33.8		0.8	304.8		0.7	904.9		0.8
NTRU L1 H.	11.0	**1.5**	0.7	35.2	**1.4**	0.8	306.1	**1.3**	0.8	906.3	**1.3**	0.9
NTRU L3	79.5		1.5	105.4		6.6	373.9		1.8	974.1		1.7
NTRU L3 H.	90.4	**10.9**	1.3	114.1	**8.6**	1.8	384.3	**10.4**	1.4	985.0	**10.9**	1.3
NTRU L5	129.8		1.5	152.6		1.8	422.3		1.8	1025.2		1.7
NTRU L5 H.	325.0	**195.2**	26.3	345.0	**192.4**	2.9	614.5	**192.2**	4.2	1214.6	**189.4**	4.0

also similar. Although client authentication is not widely used in common TLS connections, analyzing Hybrid KEMTLS with client authentication is important for completeness.

The performance of different hybrid approaches (in KEMTLS and PQTLS) was also evaluated. The main difference is that hybrid PQTLS has a dual-signature operation for the handshake transcript data. Hybrid KEMTLS replaces it with two KEM encapsulations (using a classical and a PQC algorithm). For a fair comparison, we fixed the certificate chain to be hybrid and only one hybrid algorithm, chosen to be Dilithium in both hybrids. Fixing the chain allows one to compare KEMTLS and PQTLS at the protocol level only (regarding handshake operations). Furthermore, RFC 8446 allows peers to omit sending Root CAs certificates (RESCORLA, 2018), so peers exchange only the Intermediate CA certificate. Fewer certificates exchanged minimize the impact of the PQC adoption, though it assumes that both peers have (and trust) the Root CA certificate pre-installed.

The metric for a fair comparison of different hybrid approaches is the time-to-send-application-data. Figures 24 to 26 compare hybrids (KEMTLS and PQTLS), considering the algorithms at each NIST security level. At level 1, the hybrid's boxes overlap, meaning similar timings. However, hybrid KEMTLS is faster at security level 3, due to the sizes below the TCP Maximum Segment Size. Considering the TCP window standard size (10 MSS), the hybrid KEMTLS using KyberL3 H. resulted in 15033 bytes in this experiment. Although the sizes can be different, in this case, the size is 16.57% smaller than hybrid PQTLS using KyberL3 and DilithiumL3, both in hybrid mode (18019 bytes). This difference incurs an additional round-trip at the TCP level. Therefore, hybrid PQTLS is slower in this experiment's configuration. Lastly, level 5 instantiations also exhibit a performance difference that favors the deployment of hybrid KEMTLS rather than PQTLS.

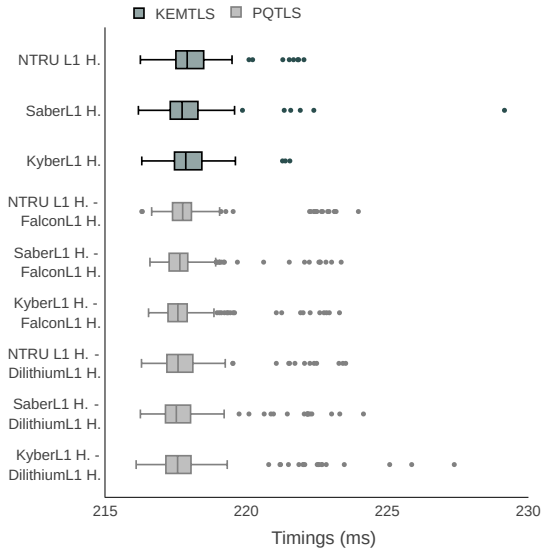

Figure 24 – Hybrids Comparison (L1)

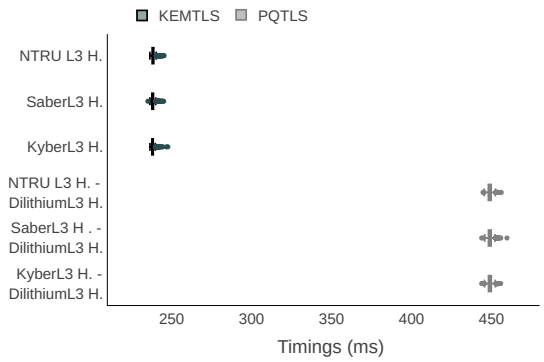

Figure 25 – Hybrids Comparison (L3)

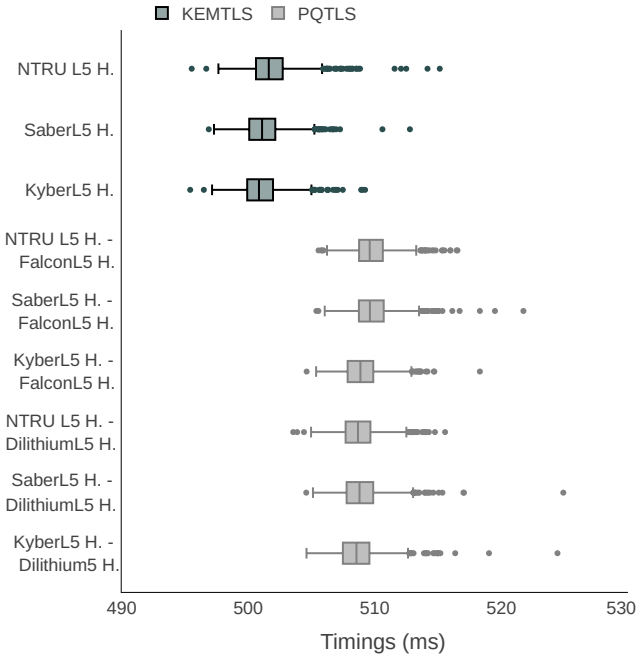

Figure 26 – Hybrids Comparison (L5)

The last experiment is the HTTPS/TLS load test. Figure 27 shows the number of successful requests for algorithms in hybrid modes using the security level 1 parameter set. The hybrids used are KyberL1 H. (`P256_Kyber512`) for KEMTLS KEX and Authentication; KyberL1 H. (KEX) and DilithiumL1 H. (`P256_Dilithium2`) (Auth) for PQTLS; KyberL1 H. (KEX) and Classic-McElieceL1 H. (`P256_Classic-McEliece-348864`) (Au-th) in KEMTLS-PDK; and KyberL1 H. and DilithiumL1 H. for PQTLS Cached-cert configuration. McEliece is selected in PDK since it optimizes size (as shown in (SCHWABE; STEBILA; WIGGERS, 2021)). RFC 7924 (SANTESSON; TSCHOFENIG, 2016) specifies caching of certificates, so KEMTLS-PDK was compared to a hybrid PQTLS instance with a Dilithium cached certificate.

Noteworthy, all configurations achieved a low error rate (mostly zero) so this information is not shown in Figure 27.

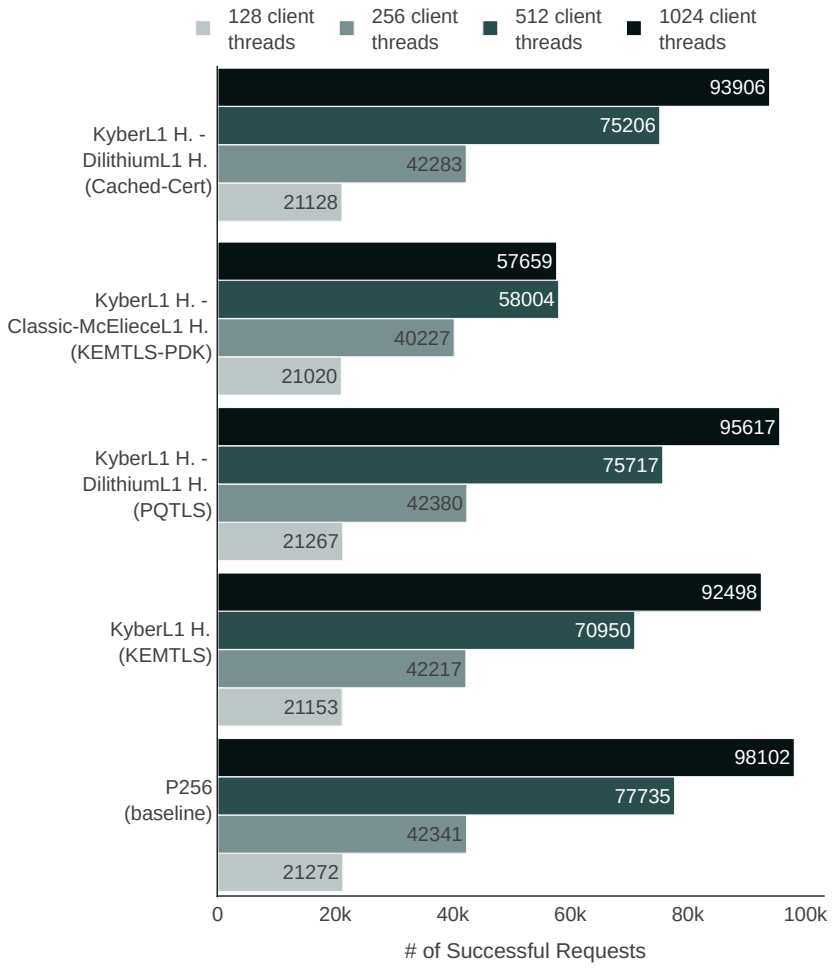

Figure 27 – Load testing with hybrids at level 1

The four configurations achieved similar number of requests when 128 client threads were used. Despite KEMTLS additional RTT for the handshake completion, KEMTLS performed similarly to PQTLS at 128 and 256 client threads. This similarity can be explained by the fact that both protocols allow the client to send application data in the same RTT, and the server's resources were not exhausted. On the other hand, when increasing the number of client threads, the differences between the configurations grow significantly. For example, hybrid KEMTLS connections contain an additional RTT, which may cause the network to get overloaded early, especially if there are more threads, resulting in fewer successful requests (as compared to PQTLS). Furthermore, the web page content size (2MB) contributes to network congestion because each client thread requests such data. Since McEliece has more significant memory requirements, the size gain in KEMTLS did not lead to better performance in this case. After 512 threads, the McEliece test reaches the computational resource limits early when compared to other alternatives. The load test did not include KEMTLS with mutual authentication scenario, which is left for future work.

Figure 28 summarizes the results, focusing on instantiations at security level one and compared to the baseline configuration. The performance penalties incurred by the hybrid approach are small. The handshake sizes increased by approximately 7x in our hybrid implementation at level 1, and the average handshake time nearly doubled due to the additional round-trip time (introduced by KEMTLS' original design). However, there is no significant penalty observed for the time-to-send-application-data at this level. The other metrics are also relevant, so for the number of requests, it may not be evident in the scale of Figure 28, but the maximum number of successful requests decreased by 3% and 6% for hybrids (PQTLS and KEMTLS, respectively, at level 1). Finally, the memory requirements have increased but possibly with minimal impact in current server configurations.

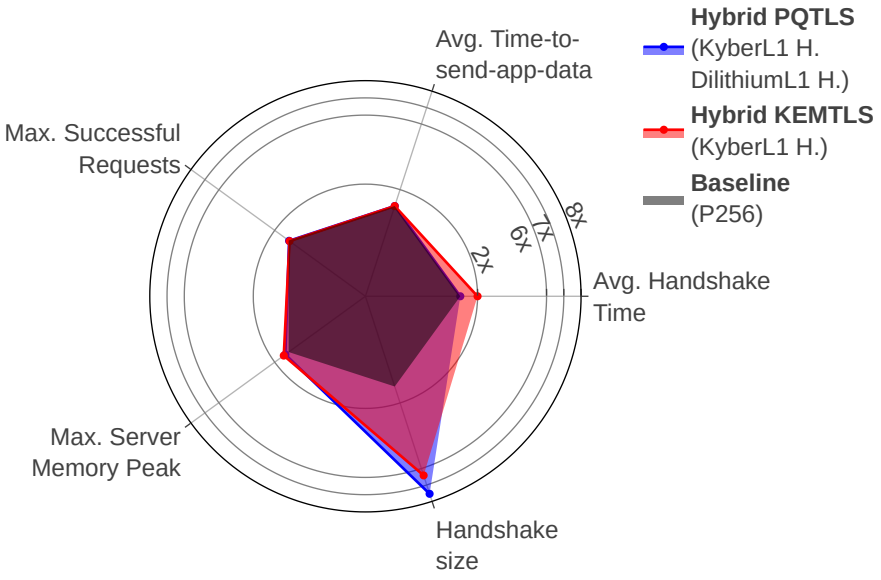

Figure 28 – Summary of performance of hybrids at level 1

Overall, the Hybrid Penalties associated with including classical algorithms into the KEMTLS design are low, especially when considering instantiations based on performance rather than higher security characteristics. The findings were validated in both simulated and geographically distant connections. As a result, hybrids are appropriate for KEMTLS (including the PDK version) since they enhance confidence from classical algorithms while incurring minor performance penalty. At security level 3, hybrid KEMTLS achieved a better fit in the TCP congestion window due to byte savings. Consequently, hybrid KEMTLS performed better than hybrid PQTLS in this work's experiments. Lastly, the implementation and all results are publicly available for the community[2].

[2] https://github.com/AAGiron/hybrid-kemtls-tests

4.4 THE PKI EXTENDED LIFETIME PERIOD (PKIELP)

Hybrid PQC is considered the recommended path to migrating applications. As a result, applications have quantum-safe security properties (from PQC) and classical security properties (from the classical part). Therefore, if a new classical attack is found against the PQC algorithm (as in SIKE (CASTRYCK; DECRU, 2022)), the hybrid still provides classical security, buying time for the application. However, if the PQC part fails to a classical attack, the hybrid modes, as the literature presents, are no longer quantum-safe.

Consider the scenario presented in Figure 29. An institution, company, or entity transits its application or software to a hybrid mode. For instance, consider that the entity has selected and migrated to a hybrid instantiated using ECDSA and a fictitious PQC signature algorithm named "X". Therefore, the application is secure against classical attacks due to ECDSA and X; and quantum-safe due to X. However, in this scenario, a vulnerability is discovered in X after the migration. In this case, the application is still secure against classical attacks (due to ECDSA) but not against the advent of quantum computers.

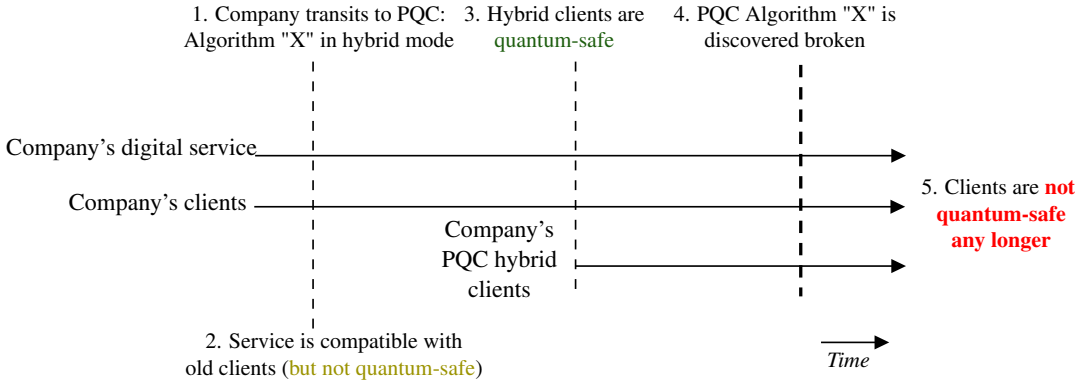

Figure 29 – PQC migration scenario where algorithm "X" has a vulnerability.

As expected, the quantum-safe properties of the hybrid mode rely solely on the PQC algorithm. Although the hybrid mode provides a contingency against classical attacks, it does not provide quantum-safe contingency: if the PQC part fails, the hybrid is no longer quantum-safe.

In this context, this section proposes an approach for the PQC transition named PKI Extended Lifetime Period (PKIELP). PKIELP is designed to solve some of the hybrid mode issues, for example, the lack of quantum-safe contingency. PKIELP's main features are listed below.

- A novel *Hybrid Quantum-safe Authentication*: Definition 2 describes this new concept. It allows both PQC signature certificates and wrapped certificates to be used. Wrapped certificates are X.509 certificates that contain an encrypted public key. Therefore, wrapped certificates are quantum-safe under Grover's algorithm (GROVER, 1996).

- Quantum-safe contingency: applications can authenticate using PQC signature certificates or wrapped certificates while maintaining quantum-safe assurances. Wrapped certificates can be used as a fail-safe authentication solution if the PQC component fails.
- An automated approach from certificate issuance to usage: by relying on ACME protocol, PKIELP allows automating the issuance of wrapped certificates, easing the adoption in TLS servers.
- Lower communication costs: Because PKIELP uses traditional public-key cryptography techniques (e.g., ECDSA) in wrapped certificates, the sizes remain close to those used today. As a result of PKIELP's hybrid quantum-safe authentication, it provides quantum-safe assurances without the requirement for PQC signatures (and their larger length).

4.4.1 PKIELP Design: a Novel Hybrid Concept

PKIELP relies on a set of assets, assumptions and roles. The main asset PKIELP relies on is the Wrapped Certificate, i.e., a certificate where the public key is encrypted. The wrapped certificate changes the public property of the public key to protect it against quantum attackers. Considering that the communication parties have a previously-shared key, they can encrypt and decrypt public keys inside certificates, thus removing Shor's threat (against authentication) and now being secure under Grover's algorithm assumption. Due to wrapped certificates, Definition 2 shows PKIELP's main property: Hybrid Quantum-safe Authentication.

Definition 2 (Hybrid Quantum-safe Authentication) *Let C_{PQC} be an X.509 digital certificate with a public key Pk_{PQC} and a digital signature from a Digital Signature scheme based on Post-Quantum Cryptography (PQC), and $C_{wrapped}$ be an X.509 digital certificate with a public key $Pk_{wrapped}$, which is symmetrically encrypted under* `cert_psk` *key, and with a digital signature from a Classical Digital ornature scheme, where Classical refers to public-key cryptography based on Integer Factorization Problem (IFP) or Elliptic Curve Discrete Logarithm Problem (ECDLP). Let H_s be a digital signature that authenticates a message m, from the private key corresponding to $C_{wrapped}$ or C_{PQC}. A Hybrid Quantum-safe Authentication scheme is an authentication mechanism for communicating peers, where the following properties hold:*

1. *quantum-safe authentication based on H_s verification using Pk_{PQC}; or*
2. *quantum-safe authentication based on H_s verification using $Pk_{wrapped}$.*

In this work, PKIELP is applied to a TLS-like scenario. PKIELP has three communicating parties: a TLS client, a TLS server, and an ACME server. In TLS, most connections have server-only authentication guarantees, whereas the client (as the user) authenticates at the application level. We do not consider TLS mutual authentication in PKIELP. TLS clients start the communication, and TLS servers change their role to ACME client when they ask for a

wrapped certificate. The first assumption is that the TLS client authenticates the TLS server in a handshake that occurs in a pre-quantum period. Therefore, it is safe to authenticate using a regular digital certificate (i.e., classical cryptography) in the first handshake. The main difference is that PKIELP adds a `cert_psk` asset, which will be later used but has to be agreed upon between the parties. So, we modify the TLS handshake to include the derivation of `cert_psk` for the TLS parties, aided by a new message `NewCertPSK`. In this way, parties can exchange a shared secret which is later derived to the wrapped certificate symmetric keys, shown in Figure 30.

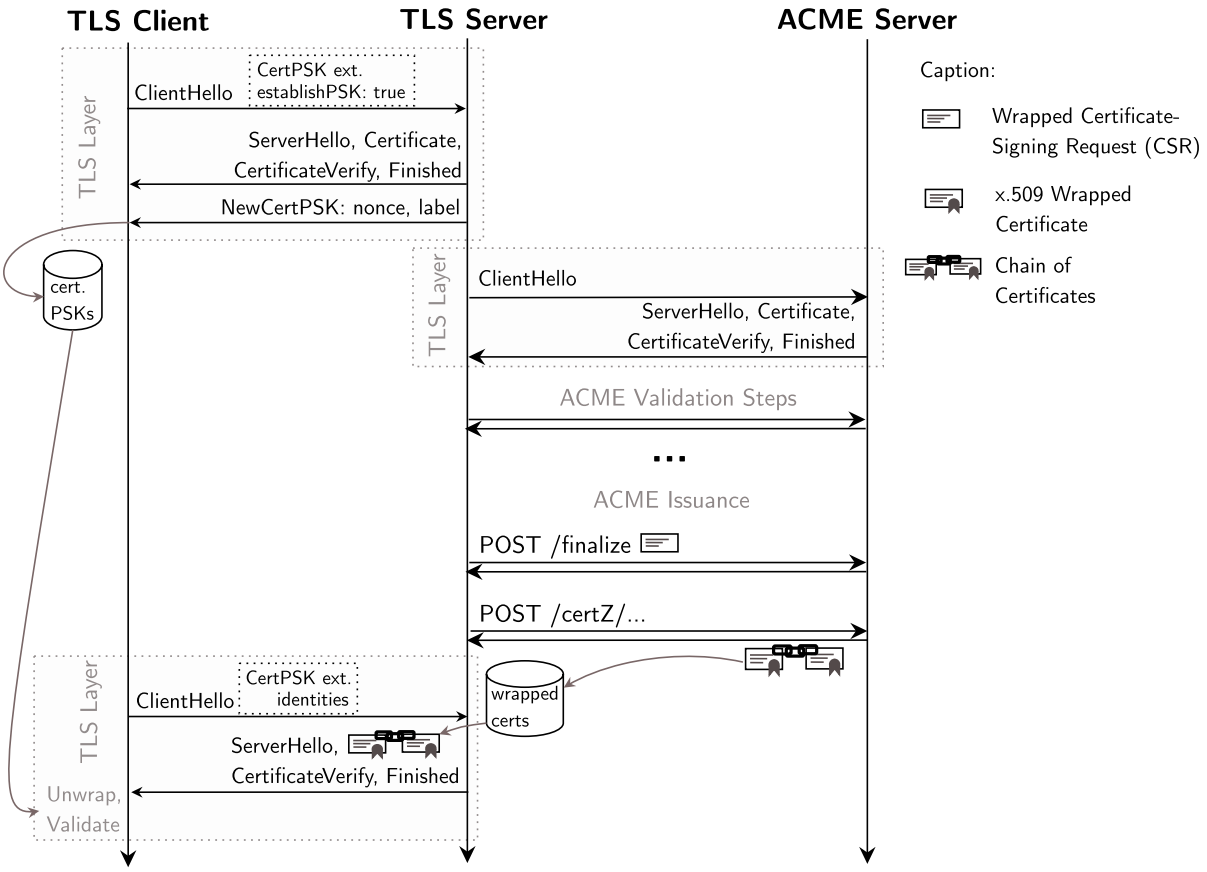

Figure 30 – Overview of PKIELP messages and roles.

In the first handshake (pre-quantum period), TLS clients set `establishPSK` in a new `CertPSK` extension when they are willing to authenticate servers using wrapped certificates. TLS servers reply as usual, except that they reply with a `NewCertPSK` message, with the required information to derive `cert_PSK` key. Note, however, that store-now-decrypt-later attacks can capture `NewCertPSK`. Thus, PKIELP requires a PQC KEX mechanism (or an Out-Of-Band PSK). In TLS, the byte cost of PQC KEMs is in one transferred public key and ciphertext, whereas the cost of PQC signatures is generally in 6 signatures and two public keys (WESTERBAAN, 2021). For evaluating PKIELP, Kyber KEM can be used as the KEX method, giving an upper bound cost when comparing wrapped certificates against TLS with PQC signatures.

Following the first handshake, the TLS server can derive `cert_psk` in a similar way as the TLS Key schedule (RESCORLA, 2018) and request a wrapped certificate. The TLS server (now the ACME client) connects with the ACME server and conducts the ACME-required validations. With the authorization, the TLS server sends a Certificate-Signing Request (CSR) containing a wrapped public key and the `cert_psk`, so the ACME server may unwrap the CSR, confirm it, and issue the wrapped certificate.

Because the certificate issuer also employs classical cryptography, making it vulnerable to quantum attacks, the PKIELP proposal wraps the issuer CA certificate in the same `cert_psk`. This extra wrapping causes a trust anchor move. This movement is seen in Figure 31. When the ACME server receives a request to provide a wrapped certificate, it will construct a new issuer CA with a wrapped certificate wrapped under the same `cert_psk` and then use this issuer CA wrapped to produce the server's wrapped certificate. Following issuance, the TLS server can download the certificate chain (end-entity and issuer CA). For the transition, the TLS server can also obtain PQC signature certificate(s) from the ACME server, giving PKIELP more flexibility.

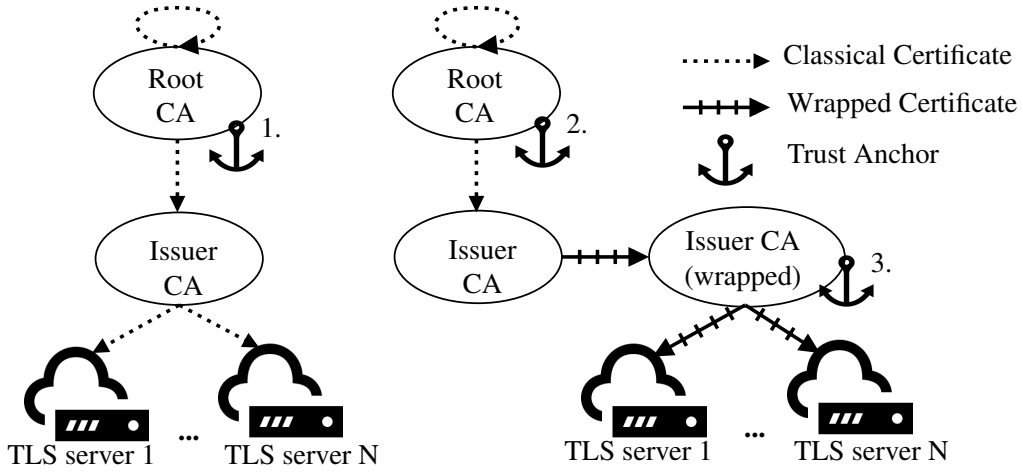

Figure 31 – Lowering PKI's Trust Anchor with Wrapped Certificates.

Regarding the trust anchor, PKIELP assumes parties have theirs set in the Root CA certificate. Then, when issuing a wrapped certificate for the server, the Issuer CA will issue a new Issuer CA with a wrapped certificate, named Issuer CA Wrapped. The first PKI in the pre-quantum period may be seen in the left portion of Figure 31, where the trust anchor is in the Root CA certificate. When a TLS server requests a wrapped certificate, the Issuer CA creates two certificates (under `cert_psk`): one for the server and one for the Issuer CA wrapped. The PKI then resembles the right side of Figure 31. It should be noted that PKIELP assumes this movement occurs during the pre-quantum period. As a result, parties can validate the whole certificate chain, up to the Root CA, prior to the movement. If the chain has been validated, the trust anchor can now be securely transferred to the new Issuer. As a result, in the post-quantum period, the trust anchor is on a wrapped (and hence protected) certificate. A benefit is that this procedure may be automated by utilizing locally trustworthy certificate stores. Despite the

fact that it employs classical cryptography, the trust anchor move to the wrapped certificates increases the PKI's lifespan. PKIELP, when compared to PQC PKIs, can minimize byte costs (see Section 4.4.2).

PKIELP allows the TLS server to offer two types of certificates: PQC signature and wrapped certificate. In the subsequent handshakes, TLS clients now set the CertPSK extension's `identity` field to the corresponding (and previously agreed) `cert_psk`. Now, depending on the received identification, the server searches its database for the associated wrapped certificate. The server signs the handshake and delivers the associated wrapped certificate so that the client may verify the signature, therefore authenticating the server in the traditional manner. It is important to note that because it is totally automated, no user participation or further setting is necessary. However, the clients must also save `cert_psk` for each server and perform the trust anchor movement.

The TLS algorithm negotiation has been modified to accommodate PKIELP extension messages. Therefore, if the client does not possess a `cert_psk`, the server can use a PQC signature certificate for authentication, so no wrapped certificate is used. PQC signatures increase handshake sizes (WESTERBAAN, 2021). In scenarios where clients execute in constrained devices, however, PQC sizes are an essential issue. This issue can be mitigated using wrapped certificates since they rely on classical cryptography. Therefore, it is paramount that devices willing to use PKIELP are updated in a pre-quantum period so that they can establish the required `cert_psk` securely.

4.4.2 PKIELP evaluation

PKIELP is developed in the Go language. It comprises a TLS and an ACME implementation, similar to the implementations provided in Section 4.2 For comparing with PQC signatures, the integrated PQC algorithms are based on liboqs-go bindings (PROJECT, 2022). In order to wrap and unwrap public keys in certificates, this PKIELP implementation offers two symmetric algorithms: `AES256-GCM` and `Ascon-80pq`. AES256 is already supported by standard Go repository and Ascon-80pq was integrated by means of a Go-binding (similar to liboqs-go). The implementation is publicly available for reproducibility purposes[3].

The selected simulation parameters are:

- 1000 TLS handshake time measurements, comparing the usage of certificates built with PQC signatures against our wrapped certificates. The PKIELP handshakes that establishes `cert_psk` were not measured, but their timings should not deviate from the results since their sizes are close.

- The algorithms for authentication are: `Dilithium2` and `Falcon512`, to build PQC signature certificates, and `Ascon80-pq` and `AES256-GCM` as wrappers for ECDSA with NIST's

[3] Available at: https://mega.nz/file/D0lmFRhL#8GpRUIKHFYgK7ROXGJDUTk0HLYoIylbE5lKtI4B2rXU

P256 elliptic curve certificates. The choice of the algorithms are based on the NIST standards. Moreover, `Kyber512` is fixed as PQC KEM in TLS KEX for all quantum-safe comparisons.

- Google VMs using the following the settings provided in Section 4.2.2. Before the experiments, the measured average RTT (ping) was 157 ms. Based on a 5G performance study (XU et al., 2020), 3.1% of packet loss probability was simulated in the link between the VMs, for the executions marked with an '*'. The simulation allows to understand better the performance in a constrained device scenario.

Figure 32 depicts the measured sizes of TLS handshakes of various setups. The baseline solely uses traditional elliptic-curve cryptography. For authentication, PKIELP instantiations use wrapped certificates, with the "wrapper" algorithm being either `AES256` or `Ascon-80pq`. We also measured the sizes of `Dilithium2` and `Falcon512` TLS handshakes under the identical experimental settings.

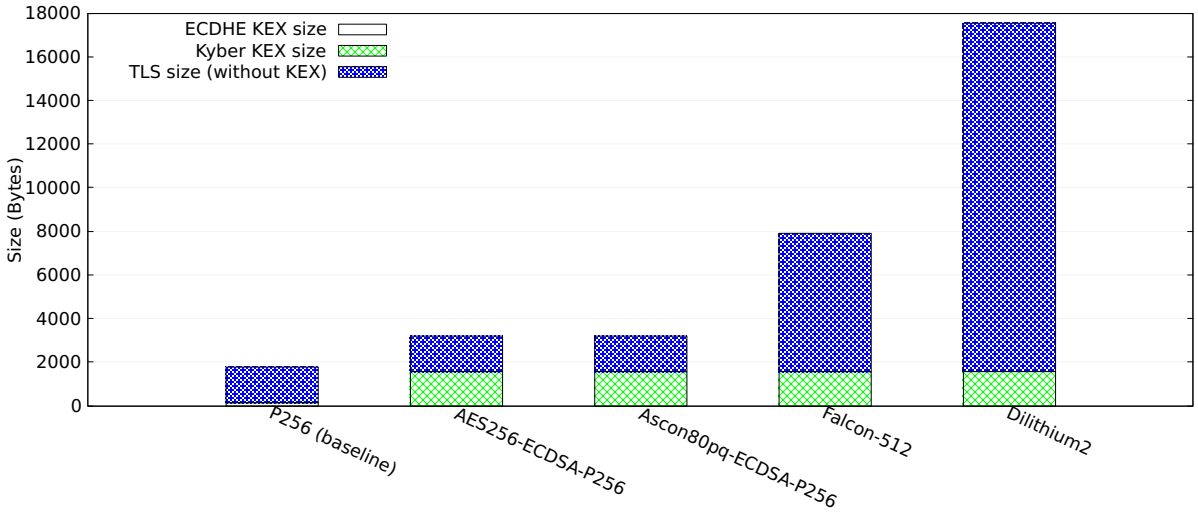

Figure 32 – PKIELP sizes comparison

PKIELP requires protection against *record-now-decrypt-later* attackers to exchange cert_psk. Therefore, we instantiate PKIELP with `Kyber512` for TLS Key Exchange (KEX), which is a post-quantum algorithm and the main responsible for the size increase. The sizes grow from 1791 bytes (baseline) to 3220 and 3223 bytes for AES and Ascon instantiation, respectively. Using a PQC KEX provides an upper-bound size evaluation. Note, however, that in some scenarios, one could assume an Out-Of-Band Pre-Shared-Key (OOB PSK), removing the need for Kyber (e.g., RFC 8773 (HOUSLEY, 2020)). Nevertheless, PKIELP gives quantum-safe guarantees with smaller sizes when compared to Dilithium and Falcon instances. For example, Dilithium2 has 5.44x bigger sizes than PKIELP. The size reduction shows the main benefit of using PKIELP for quantum-safe guarantees.

Figure 33 presents a box plot of the handshake times. It is clear from the figure that the timings are close; the overlaps between the boxes show no significant penalty in using wrapped

certificates for authentication. The Dilithium instance was removed from Figure 33 because its average handshake time is above 300 ms, caused by an additional round trip. For readability, Figure 33 keeps only the instances that did not trigger an additional RTT due to size. Falcon, AES and Ascon instances were tested under 3.1% packet loss probability to comprehend better the size issue present in 5G links of constrained devices (XU et al., 2020). PKIELP provided faster handshakes (compared to Dilithium) and close timings to the Falcon instance.

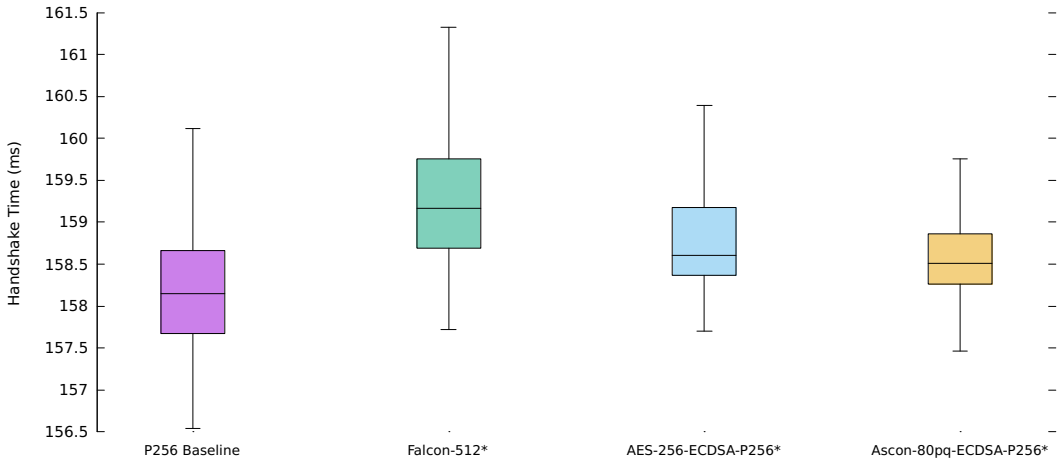

Figure 33 – Handshake times of PKIELP instances compared to baseline and Falcon.

The above results suggest that the approach is viable for establishing TLS connections with smaller sizes and comparable performance. In practice, PKIELP offers a contingency plan which is a feature not supported by current hybrids. Therefore, for a safer PQC transition, and in some particular scenarios, PKIELP provides a fail-safe alternative to establish quantum-safe TLS connections. The next section discusses how to further investigate PKIELP and the identified threats to its security.

4.4.3 PKIELP Applicability and Security

This section discusses the obtained PKIELP performance and how to apply PKIELP in a particular scenario. It also describes possible attacks from quantum adversaries and then analyses dishonest participants and other identified threats.

In the context of Vehicular Networks (VNs), SCMS (BARRETO et al., 2018) and IEEE WAVE standard (IEEE, 2020) have solutions based on Public-Key Infrastructure (PKI), Transport Layer Security (TLS) protocol, and digital certificates for authentication. When adopting PQC, however, communication will be severely affected by PQC's increased sizes, particularly for digital signing. In TLS, studies suggest potential delays caused by PQC, caused mainly by authentication components (e.g., PKI certificate chains) (SIKERIDIS; KAMPANAKIS; DEVETSIKIOTIS, 2020). Such an example violates the QoS requirements of VNs (KADHIM; SENO, 2019), as PQC can severely affect the payload sizes. Therefore, VNs will benefit from a PQC transition approach that costs as fewer bytes as possible, such as PKIELP. PKIELP sizes

are smaller than if using PQC signatures. As a result, PKIELP is suitable for constrained devices and delay-sensitive networks such as VNs.

Figure 34 shows how one can apply PKIELP in the Vehicle-to-Infrastructure (V2I) VN scenario. Certificate authorities can issue wrapped certificates for TLS servers automatically, securing automotive service providers (such as for route recommendations). TLS clients are cars' On-Board-Units (OBUs) that can connect to such services. Their connection is established following Figure 30, thus allowing a smaller TLS handshake due to the wrapped certificates usage.

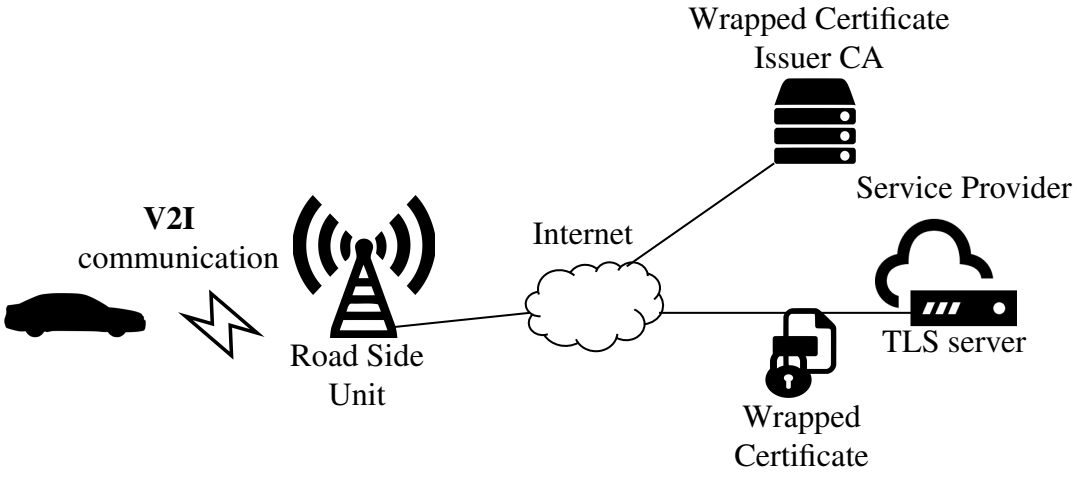

Figure 34 – Proposed Architecture.

It should be noted that the PKIELP certificate distribution procedure, which uses ACME, is incompatible with the V2V scenario. Since ACME currently provides certificates for a different domain (web servers), it does not consider certificates for identifying OBUs in cars. Such an applied evaluation is considered an open challenge for future work.

Aside from the size issue, the literature should address other aspects of PQC transition. The transition will need multiple phases, one of which will include software updates. Although there are secure proposals for remote firmware updates, they generally rely on traditional cryptography. Furthermore, including PQC algorithms will increase the code size of the trusted code base, not to mention the concerns about implementation attacks. When it comes to VNs, the lengthy lifespan of vehicles may result in an expensive transition in embedded technology. The obsolescence of ECC co-processors and RSA-optimized HSMs caused by the transition would result in financial losses as well as environmental issues (e.g., electronic waste). Furthermore, hardware optimizations for PQC may be available only later, making it difficult to synchronize with the product's time-to-market (e.g., a new model of a car). In conclusion, PKIELP can aid in this context because it extends the lifetime period of existing PKI and already deployed classical cryptography.

Lastly, this section discusses the following threats for PKIELP:

1. Record-now-decrypt-later `NewCertPSK` extension message: this quantum attack could be used to recover `cert_psk` (see Figure 30). Having `cert_psk`, the attacker could de-

crypt the wrapped certificate and then recover the private key given a classical (RSA or ECDSA) public key by using a quantum computer (in the future), breaking our authentication scheme. In order to mitigate this issue, one option is to use a PQC KEX algorithm in TLS, which is less costly when compared to PQC authentication in TLS.

2. Compromise of the `cert_psk` symmetric key by an invasion in the TLS client: similarly, the attacker gains access to the unwrapped server's public key. Aided by a quantum attacker, it allows the impersonation of the server. In such a case, a recommended mitigation for PKIELP is to use one wrapped certificate per client. Although this key compromise is obtained from an invasion of the client's system, such an attack allows the impersonation of the server to only one client. Therefore, this attack on the PKIELP level has no advantage because the attacker already has access to the client's system. Therefore, there is no need to attack PKIELP if the client's system is already compromised otherwise.

3. Compromise of the `cert_psk` symmetric key by an invasion in the TLS server: allows a quantum attacker to recover the private key from the unwrapped server's certificate. Again, this key compromise can be equivalent to a system invasion, giving access to the private key. However, a system invasion can access the private key without a quantum attack. Thus no advantage is obtained.

4. TLS Client's PSK label replay attack: the attacker Attacker could replay a Cert. PSK identity, then the server will reply to the wrapped certificate. Since the public key is encrypted, the quantum attacker can not obtain the private key, so no advantage is given.

This work has evaluated a PKIELP implementation in Go. It provides experiments to compare the performance of both symmetric algorithms, but the post-quantum security of these instantiations is different. Moreover, the network times have concealed those algorithms' performance differences in the experiments. While experimenting AES with a 256-bit key provides a theoretically 128-bit security claim against Grover's quantum algorithm, Ascon's selected parameter set has a smaller (160-bit) key. Ascon is weaker in this case, but Ascon's quantum security still needs to be extensively explored in the literature.

There are also additional quantum threats associated with ACME communication. TLS is frequently used by ACME to securely connect to ACME servers. As a result, PKIELP assumes a pre-quantum setting, ACME clients transit to a wrapped certificate in order to use wrapped certificates in a post-quantum scenario. ACME servers can employ either a PQC or a wrapped certificate. Because issuer CAs typically manage ACME servers, they must be equipped with a post-quantum solution. However, if PKIELP is used, the trust anchor movement could be performed in the ACME roles, automating their transition. These aspects of the ACME transition were not measured in the experiments. Because the PKIELP process begins in a pre-quantum period, there is no quantum threat in PKIELP authentication because such attackers cannot impersonate retroactively.

In terms of dishonest parties, a single TLS peer (or ACME peer) can be dishonest and reveal `cert_psk` to a quantum attacker. This is most likely to occur in three scenarios: TLS Client, TLS Server, and ACME Server. If the TLS client is dishonest, the attacker can only pretend to be a server for that client. As a result, there is no benefit to acting dishonestly. Suppose now that the TLS server is dishonest and discloses `cert_psk` to the quantum attacker. This behavior is equivalent to revealing its private key, so the server has no advantage. Finally, if the ACME Server is untrustworthy, the quantum attacker can obtain the desired `cert_psk` keys, indicating that the CA's trust assumption has been violated. Furthermore, if Issuer CAs act dishonestly, it harms their reputation, resulting in financial loss and exclusion from browser trust stores. Standard mitigating techniques, such as Certificate Transparency Logs, can be used in this situation. In conclusion, all threats identified in PKIELP can be mitigated. Therefore PKIELP is feasible and encourages further evaluation analysis.

5 DISCUSSION

This chapter discusses the hybrid modes proposed in this thesis. Section 5.1 discusses the design considerations when using hybrids (in this work's experiments). Section 5.2 discusses the observed performance of hybrids in the results obtained in this work.

5.1 THE HYBRID DESIGN

The first step in understanding the adoption of the hybrid mode in practice is its design. Literature helped in this regard since using concatenation approaches simplifies the implementation. The implementations used concatenation in KEX and authentication mechanisms (e.g., certificates). However, although simpler adoption, it can be challenging. For example, for a complete migration to hybrids in the ACME protocol, the implementation requires hybrids in the account signatures, in the certificates (end-entity, Issuer CA, intermediates, and Root CAs), and also in the underlying TLS channel, i.e., in the KEX and authentication (signatures and certificates).

The awareness of the hybrid design is an important aspect. Understanding the hybrid design characteristics is a direct result of the SMS presented in Section 3.1. From the hybrid design classification (Figure 11), hybrid implementations require conveying and combining approaches, not to mention additional decisions (such as the algorithm and parameter choices). In this work, each hybrid choice paired classical and PQC parameters at the same security level claims for conciseness. Note that the convention of two algorithms suffices for the hybrid properties, but hybrids with more PQC algorithms could be explored (increasing byte costs, however).

Transparent cryptography is important for network protocols. However, a variety of users might want to inspect the presence of hybrids in their connections. Developers and system administrators would test their websites' security, and regular users could perform similar checks. This perspective is not well addressed by the tools available, but it is nevertheless important for the awareness of quantum threats. Since CLIs can threaten usability for regular users, our proposed TLS 1.3 Handshake Analyzer supports both CLI and web interfaces. Improving usability for these inspections benefits a broader community.

5.2 HYBRID PERFORMANCE RESULTS

This work achieves a higher level of completeness when experimenting with hybrids in Internet Security (HTTPS), from a configuration with ACME to testing TLS endpoint connections. In general, the configuration step should not be neglected since it is the primary requirement for authentication in HTTPS connections, i.e., the digital certificate. In this context, the importance of the ACME protocol in Internet security is evident. ACME simplifies certificate management for Internet servers worldwide. This study examined ACME's performance when

secured with Post-Quantum Cryptography (PQC) techniques. The comparisons with classical cryptography revealed differing impacts for ACME clients and servers. Furthermore, from a PQC-transition standpoint, this study has identified minimal disadvantages between PQC-only deployment and hybrid PQC. It is worth noting that this thesis did not consider PQC algorithms above security level 5 (NIST's 256 bits). For example, for 512-bit security, one would have to increase the algorithm parameters (e.g., lattice dimensions), but OQS implementations are not offering such parameters.

In addition, a great number of devices and users connect through the TLS protocol. Therefore, this work evaluated TLS connections with hybrid signatures, hybrid KEMs, and wrapped certificates, all compared to baseline approaches (e.g., classical cryptography). Furthermore, both simulated and realistic networks were tested, including a wide range of algorithm parameters. The extensive experimentation performed in this work allows a better comprehension of the impacts of PQC. Once again, the overall recommendation remains on adopting the hybrid strategy since handshake times and sizes are close to the PQC-only adoption. Since hybrids add a "security layer", their usage can be the first step toward the PQC migration.

TLS connections are used for more than just web browsing. The increased PQC sizes can still be challenging for integration in some applications. In this line, this work's proposal PKIELP can improve the payload by offering flexibility. TLS "thin clients" could enjoy fewer payload bytes by establishing TLS in the PKIELP approach (please see Figure 32). Given that different domains have TLS or PKI solutions being standardized (such as 5G standards (CLANCY; MCGWIER; CHEN, 2019)), PKIELP is also an interesting choice for PQC migration. Moreover, PKIELP is offering contingency by protecting the classical certificates with encryption. Current hybrid solutions in the literature do not offer this functionality, but they should consider doing so for security reasons.

Although the hybrids are being recommended, this thesis did not rank .or recommend a particular instance of hybrid PQC over another. Instead, it shows the results for the given applications and protocols without highlighting the best hybrid configuration. First, some hybrids performed similarly (such as the lattice-based KEMs from NIST Round 3). Secondly, to issue a recommendation, several requirements come into place, where some of them are related to the intended application so that it can differ from the experiments in this thesis. This work demonstrates extensive experiments and situations where hybrids (in general) can be recommended, but they can not be generalized to any application. One can refer to the "PQC for Engineers" IETF draft for additional adoption guidelines (BANERJEE et al., 2023).

All in all, this work has shown that hybrid PQC is worthwhile for different reasons. However, not all scenarios have been captured in this work. For example, IPSec and other protocols part of the TCP/IP model also employ PKC and thus are susceptible to quantum attacks. Experimenting in other TCP/IP layers is important and constitutes one line for further research (discussed in Section 6.2).

6 CONCLUSIONS

Public-key cryptosystems are often used for authentication and Key Exchange (KEX) in a wide range of applications. Such cryptosystems are vulnerable to Shor's quantum algorithm, allowing impersonation attacks and breaking confidentiality. The immediate solution to the quantum threat is a replacement of vulnerable algorithms by PQC. Sometimes called "PQC Drop-in replacement" or "PQC-only deployment", the vulnerable KEX and authentication mechanisms are replaced solely by PQC alternatives. Applications equipped with PQC can resist quantum threats, but there are still threats imposed by classical computation. Therefore, an alternate solution is called hybrid mode. Hybrids combine traditional and PQC, so the system is secure as long as one of the combined parts is secure. This thesis proposed, evaluated and compared different hybrid modes to anticipate adoption problems and solutions.

First, Section 4.1 dealt with the awareness problem. Users might want to assess whether they adopt PQC with or without hybrids. The TLS 1.3 Handshake Analyzer aids users in assessing TLS connection performance and security algorithms. Section 4.2 analyzed the first step in configuring TLS connections for PQC migration: issuing certificates for web servers. The certificates are issued with PQC and hybrids by an ACME implementation, modified to accommodate PQC and also with optimized features for faster issuance. Section 4.3 dealt with the lack of hybrid KEMTLS implementations as a quantum-safe strategy for TLS connections. Finally, Section 4.4 proposed a novel hybrid concept for authentication in aid of scenarios where sizes matter most, and it is the first hybrid allowing quantum-safe contingency.

In conclusion, the hybrid proposals have addressed the adoption problem from different perspectives. All of the proposals have more benefits than drawbacks compared to the PQC-only migration strategy. The performance penalties were minor in most scenarios: from certificate issuance to hybrid KEMTLS connections. For instance, the results favour hybrids when comparing KEMTLS to hybrid KEMTLS since the penalties are insignificant. When instantiating the proposed PKIELP's wrapped certificates, performance is improved over PQC signature certificates, mostly due to smaller sizes. Therefore, this thesis' results encourage the adoption of hybrid modes as a PQC migration strategy.

When comparing the options for accommodating PQC, the results showed that changing the protocol for PQC scenarios can improve performance. For example, the ACME PQ-Transition Challenge reduced certificate issuance time. Similarly, PKIELP can also reduce TLS connection times by modifying how TLS handles authentication certificates. Such improvements suggest that PQC adoption challenges can be better handled if the protocol is prepared for the increased PQC sizes. However, such protocol modifications should be offered instead of enforced to keep compatibility with older (yet-to-be-updated) implementations.

6.1 FUTURE WORK

The scope of this work left out some network protocols part of the TCP/IP model. They are also important; some are not yet extensively experimented with PQC or hybrids. In particular, hybrids have the 2-ingredient convention (e.g., two signatures in parallel), but hybrids with more than two algorithms have not been extensively evaluated. How to design such a hybrid would be an interesting research direction, aiming at security from different assumptions (e.g., combining lattices and code-based cryptography).

One line of future work for the TLS 1.3 Handshake Analyzer tool (Section 4.1) would be to improve overall usability. Given that pyshark provides a live-capture mode, there is an ongoing integration effort of this feature into this tool. The live mode would allow users to check the security and performance of their connections in real-time. Still, there will be an additional requirement regarding the TLS keylog file configuration for this new feature. The tool's repository tracks the live mode and further improvements for future work.

The ACME protocol currently lacks support for other PQC adoption approaches, for example, KEMTLS. Therefore, future work includes studying approaches for issuing KEMTLS certificates (i.e., certificates with a KEM public key). Given the need for a CSR-like process for KEMs, issuing and managing certificates for KEMTLS in an automated way is challenging. Approaches like verifiable key generation (GüNEYSU et al., 2022), X.509 extensions, and the CMP draft can be an initial guide. In this regard, an Internet Draft (I-D) is being proposed for KEMTLS certificate issuance[1], conducted as a follow-up work of this thesis. Finally, evaluating how a quantum-safe ACME behaves in different computing environments (such as IoT-to-cloud) is also an interesting research direction.

Although this work proposed an extensive evaluation for hybrid KEMTLS, there are other scenarios for evaluating it. Examples include Internet-of-Things (IoT) and 5G networks, which might require energy consumption as an essential evaluation metric. Additionally, incorporating other PQC migration strategies for TLS, such as suppression of intermediate certificates (SIKERIDIS et al., 2022), can lead to better performance.

The PKIELP approach is promising for specific applications where PQC adoption is challenging. Therefore, PKIELP can be further evaluated using devices used in VNs and 5G networks. This thesis provides the basis for understanding PKIELP performance (as Figure 33), but additional experiments were left out of the scope. Besides, PKIELP could be improved to a better fit in Vehicle-to-Vehicle (V2V) communications since nowadays they rely on pseudonym certificates and, currently, there is no such support in ACME. Improving PQC migration alternatives involving hybrids are important for a better and more inclusive PQC migration.

[1] https://github.com/AAGiron/acme-pqc-negotiation

6.2 TAKEAWAYS

This thesis argues favouring a hybrid strategy for the PQC migration for several reasons, such as confidence in traditional cryptography, soft transition, compliance with regulations, compatibility, and minor penalties. However, one finding of this work is that the transition to PQC goes beyond algorithm replacement (hybrid or PQC-only). This section explains why and gives concrete examples below, some are explained in a position paper (GIRON, 2023).

Consider a common Internet-banking application. Users communicate with it using classical cryptography (thus vulnerable to quantum computers). At some point, the application migrates to a PQC infrastructure. From now on, user data in traffic is protected against quantum attackers. However, user data could have been captured before the migration to be decrypted with a quantum computer. If the user has communicated confidential data, such as the bank account number and passwords, this data is at risk of future disclosure. Therefore, even after the PQC migration, long-term confidential data enables further interactions between the attacker and the application, for example, by performing financial transactions on the user's behalf.

The preceding example demonstrates how long-term sensitive information might be exploited after an application's PQC migration. As bank secrecy is a requirement for the internet-banking application, administrators must manage or change user data in light of the knowledge quantum adversaries can obtain. Therefore, PQC algorithm replacement may not be sufficient for a complete migration to the post-quantum age if confidential (or long-term) user data allows future interactions with the software. As a result, a "quantum risk assessment approach" should contain user data policies and security procedures for better protection.

Table 11 depicts the risks of several application-layer protocols and software in the face of a *record-now-decrypt-later* attack. Each application requires a secure channel, which is frequently provided through TLS. Applications transmitting personal data have different risks because secure channel providers can be subject to quantum attacks. With these concrete examples, a complete migration should include not only PQC algorithm replacement but also a careful risk analysis.

It is worth noting that application-layer protocols do not require substantial modifications to accommodate PQC. In certain circumstances, changing their TLS-based setups (such as digital certificates) and implementations is sufficient for the transition to PQC. Nevertheless, this may not be sufficient to protect against the risks associated with *record-now-decrypt-later* attacks. As a result, any application protocol may require further investigation and updates for full security. Considering the risks in Table 11, mitigating strategies include limiting authorizations and access token duration time, implementing a policy for long-term sensitive data usage, and cancelling previous activities conducted with classical cryptography. The biggest disadvantage is that developing such measures in the application increases the PQC migration effort.

This thesis results towards the hybrid adoption shows encourage hybrid usage as a first transition step since it maintains confidence in the security of traditional methods. Given this

Table 11 – Application-layer risks under a record-now-decrypt-later threat (not exhaustive).

Application-layer Protocol/Utility	Specification	Secure Channel Provider	Sensitive Information	Risk Description
Basic HTTP Authentication	RFC 7617	TLS	User Credentials	Exchanged long-term credentials can allow access to server's resources after breaking TLS with a quantum computer
OAuth 2.0	RFC 6749	TLS	Refresh token	RFC leaves to implementations to explicitly define expiration time; an example of refresh token expiration time is one year (RESTREPO, 2022). Attackers could obtain valid tokens exchanged with classical TLS.
OIDC/OAuth 2.0	(SAKIMURA et al., 2023)	TLS	ID Token, Refresh tokens	Similar to OAuth 2.0 (already pointed out by (SCHARDONG et al., 2022))
Kerberos V5 (with kinit)	RFC 4120, RFC 4556	N/A	Renewal Ticket	In theory, ticket-granting tickets exchanged with classical cryptography combined with a long-lifetime ticket renewal policy (from 0 to 99,999 days) (LONG et al., 2023) could be exploited by a quantum attacker.
Email Protocols	RFC 8314	TLS	User Credentials	RFC 8314 recommends TLS for IMAP, SMTP and other email protocols. Quantum attackers could exploit long-term user credentials exchanged with TLS.
WebRTC	(W3C, 2023)	DTLS	Authentication password	WebRTC specifies different authentication methods, if long-term passwords are used, a quantum attacker could recover the password after breaking the DTLS session.
Rsync over SSH	(TRIDGELL; MACKERRAS; DAVISON, 2022)	SSH	Server password	Rsync allows sharing files over SSH for security. A quantum attacker could decrypt SSH tunnels and recover exchanged rsync passwords.

context, the PQC adoption challenges addressed in this work, and the application-level risks mentioned in this section, the main takeaways are:

- Given the favorable circumstances for performance comparisons and security trust in hybrid PQC, consider hybrids as the suggested PQC migration approach. Remember that implementing PQC is still difficult in some applications, such as those with larger sizes. To improve performance when adopting hybrids, consider alternate approaches like hybrid KEMTLS (Section 4.3), the PQ-Transition Challenge for ACME (Section 4.2.3) and PKIELP (Section 4.4).

- Consider that a PQC migration includes: a PQC algorithm selection that best suits the application's needs and risk analysis related to confidential data and other long-term information that quantum-capable attackers could explore. In short, except when dealing with long-term private data sent over quantum-vulnerable protocols, application-layer software requires no substantial adjustments to handle PQC.

- Inspect if the PQC migration plan has been successfully deployed in the application. For applications communicating through quantum-safe TLS, Section 4.1 provides a tool to check whether quantum-safe algorithms were negotiated in the connection.

Hybrid modes for the post-quantum transition might look like a short-term solution (or the first step), but it does not necessarily imply that it will be a brief period. On the contrary, Hybrid PQC can be present in network communications for as long as is required to obtain full trust in PQC security. Furthermore, understanding the effects of quantum threats and how to mitigate them in advance aids in the construction of a secure post-quantum world.

BIBLIOGRAPHY

ALKIM, E. et al. **FrodoKEM learning with errors key encapsulation**. [S.l.]: Technical report, 2020.

AVIRAM, N. et al. **Practical (Post-Quantum) Key Combiners from One-Wayness and Applications to TLS**. 2022. Cryptology ePrint Archive, Paper 2022/065. https://eprint.iacr.org/2022/065. Disponível em: https://eprint.iacr.org/2022/065.

AZARDERAKHSH, R. et al. Hardware deployment of hybrid pqc: Sike+ecdh. In: GARCIA-ALFARO, J. et al. (Ed.). **Security and Privacy in Communication Networks**. Cham: Springer International Publishing, 2021. p. 475–491. ISBN 978-3-030-90022-9.

BANERJEE, A. et al. **Post-Quantum Cryptography for Engineers**. 2023. http://tools.ietf.org/html/draft-ar-pquip-pqc-engineers-03. Internet-Draft.

BARKER, E.; CHEN, L.; DAVIS, R. Recommendation for key-derivation methods in key-establishment schemes revision 2. **NIST Special Publication**, v. 800, p. 56C, 2020.

BARNES, R. et al. **Automatic Certificate Management Environment (ACME)**. [S.l.], 2019.

BARRETO, P. S. L. M. et al. **qSCMS: Post-quantum certificate provisioning process for V2X**. 2018. Cryptology ePrint Archive, Paper 2018/1247. https://eprint.iacr.org/2018/1247. Disponível em: https://eprint.iacr.org/2018/1247.

BERNDT, S.; LIśKIEWICZ, M. Algorithm substitution attacks from a steganographic perspective. In: **Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2017. (CCS '17), p. 1649–1660. ISBN 9781450349468. Disponível em: https://doi.org/10.1145/3133956.3133981.

BERNSTEIN, D. J.; LANGE, T. Post-quantum cryptography. **Nature**, Nature Publishing Group, v. 549, n. 7671, p. 188–194, 2017.

BEULLENS, W. **Breaking Rainbow Takes a Weekend on a Laptop**. 2022. Cryptology ePrint Archive, Paper 2022/214. https://eprint.iacr.org/2022/214. Disponível em: https://eprint.iacr.org/2022/214.

BINDEL, N. et al. Hybrid key encapsulation mechanisms and authenticated key exchange. In: DING, J.; STEINWANDT, R. (Ed.). **Post-Quantum Cryptography**. Cham: Springer International Publishing, 2019. p. 206–226. ISBN 978-3-030-25510-7.

BINDEL, N. et al. Transitioning to a quantum-resistant public key infrastructure. In: LANGE, T.; TAKAGI, T. (Ed.). **Post-Quantum Cryptography**. Cham: Springer International Publishing, 2017. p. 384–405. ISBN 978-3-319-59879-6.

BIRGE-LEE, H. et al. Experiences deploying Multi-Vantage-Point domain validation at let's encrypt. In: **30th USENIX Security Symposium (USENIX Security 21)**. (online): USENIX Association, 2021. p. 4311–4327. ISBN 978-1-939133-24-3. Disponível em: https://www.usenix.org/conference/usenixsecurity21/presentation/birge-lee.

BOS, J. et al. Frodo: Take off the ring! practical, quantum-secure key exchange from lwe. In: **Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2016. (CCS '16), p. 1006–1018. ISBN 9781450341394. Disponível em: https://doi.org/10.1145/2976749.2978425.

BOS, J. et al. Frodo: Take off the ring! practical, quantum-secure key exchange from lwe. In: **Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2016. (CCS '16), p. 1006–1018. ISBN 9781450341394. Disponível em: https://doi.org/10.1145/2976749.2978425.

BOS, J. et al. Crystals-kyber: a cca-secure module-lattice-based kem. In: IEEE. **2018 IEEE European Symposium on Security and Privacy (EuroS&P)**. London, UK: IEEE, 2018. p. 353–367.

BOS, J. W. et al. Post-quantum key exchange for the tls protocol from the ring learning with errors problem. In: **2015 IEEE Symposium on Security and Privacy**. San Jose, CA, USA: IEEE, 2015. p. 553–570.

BRAITHWAITE, M. **Experimenting with post-quantum cryptography**. 2016. https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html.

BRENDEL, J.; FISCHLIN, M.; GÜNTHER, F. Breakdown resilience of key exchange protocols: Newhope, tls 1.3, and hybrids. In: SAKO, K.; SCHNEIDER, S.; RYAN, P. Y. A. (Ed.). **Computer Security – ESORICS 2019**. Cham: Springer International Publishing, 2019. p. 521–541. ISBN 978-3-030-29962-0.

BROWN, R. G. **Dieharder: a random number test suite version 3.31.** 2019. Urlhttp://manpages.ubuntu.com/manpages/bionic/man1/dieharder.1.html.

CAMPAGNA, M.; PETCHER, A. **Security of Hybrid Key Encapsulation**. 2020. Cryptology ePrint Archive, Report 2020/1364. https://eprint.iacr.org/2020/1364.

CASTRYCK, W.; DECRU, T. **An efficient key recovery attack on SIDH (preliminary version)**. 2022. Cryptology ePrint Archive, Paper 2022/975. https://eprint.iacr.org/2022/975. Disponível em: https://eprint.iacr.org/2022/975.

CELI, S. et al. Implementing and measuring kemtls. In: LONGA, P.; RÀFOLS, C. (Ed.). **Progress in Cryptology – LATINCRYPT 2021**. Cham: Springer International Publishing, 2021. p. 88–107. ISBN 978-3-030-88238-9.

CHAN, C.-l. et al. Monitoring tls adoption using backbone and edge traffic. In: IEEE. **IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)**. [S.l.], 2018. p. 208–213.

CHO, J. Y. Securing optical networks by modern cryptographic techniques. In: ASKAROV, A.; HANSEN, R. R.; RAFNSSON, W. (Ed.). **Secure IT Systems**. Cham: Springer International Publishing, 2019. p. 120–133. ISBN 978-3-030-35055-0.

CLANCY, T. C.; MCGWIER, R. W.; CHEN, L. Post-quantum cryptography and 5g security: Tutorial. In: **Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks**. New York, NY, USA: Association for Computing Machinery, 2019. (WiSec '19), p. 285. ISBN 9781450367264. Disponível em: https://doi.org/10.1145/3317549.3324882.

CROCKETT, E.; PAQUIN, C.; STEBILA, D. **Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH.** 2019. Cryptology ePrint Archive, Report 2019/858.

DOWLING, B.; HANSEN, T. B.; PATERSON, K. G. Many a mickle makes a muckle: A framework for provably quantum-secure hybrid key exchange. In: SPRINGER. **International Conference on Post-Quantum Cryptography**. Cham: Springer International Publishing, 2020. p. 483–502.

ESI. **KSI Blockchain**. 2023. Available at: https://e-estonia.com/solutions/cyber-security/ksi-blockchain/.

F5 Labs. **2019 TLS Telemetry Report Summary**. 2019. https://www.f5.com/content/dam/f5-labs-v2/article/pdfs/F5Labs-2019-TLS-Telemetry-Report-Summary.pdf.

FACTOR, K. **Post-Quantum Hybrid Cryptography in Bouncy Castle**. 2023. https://doc.primekey.com/bouncycastle/post-quantum-hybrid-cryptography-in-bouncy-castle.

FAZ-HERNáNDEZ, A.; KWIATKOWSKI, K. **Introducing CIRCL: An Advanced Cryptographic Library**. [S.l.], 2019. Available at https://github.com/cloudflare/circl. v1.3.2 Accessed Jan, 2023.

FOUNDATION, E. F. **Certbot - get your site on https.** 2022. https://certbot.eff.org/.

GHEDINI, A.; VASILIEV, V. **TLS Certificate Compression**. [S.l.], 2020.

GHOSH, S.; KATE, A. Post-quantum forward-secure onion routing. In: MALKIN, T. et al. (Ed.). **Applied Cryptography and Network Security**. Cham: Springer International Publishing, 2015. p. 263–286. ISBN 978-3-319-28166-7.

GIACON, F.; HEUER, F.; POETTERING, B. Kem combiners. In: ABDALLA, M.; DAHAB, R. (Ed.). **Public-Key Cryptography – PKC 2018**. Cham: Springer International Publishing, 2018. p. 190–218. ISBN 978-3-319-76578-5.

GIRON, A.; CUSTÓDIO, R. An entropy source based on the bluetooth received signal strength indicator. In: **Anais do XX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**. Porto Alegre, RS, Brasil: SBC, 2020. p. 106–118. Disponível em: https://sol.sbc.org.br/index.php/sbseg/article/view/19231.

GIRON, A.; SCHARDONG, F.; CUSTÓDIO, R. Tls 1.3 handshake analyzer. In: **Anais Estendidos do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**. Porto Alegre, RS, Brasil: SBC, 2022. p. 63–70. Disponível em: https://sol.sbc.org.br/index.php/sbseg_estendido/article/view/21693.

GIRON, A. A. Encouraging the adoption of post-quantum hybrid key exchange in network security. In: GARCIA-ALFARO, J. et al. (Ed.). **Security and Privacy in Communication Networks**. Cham: Springer International Publishing, 2021. p. 363–371. ISBN 978-3-030-90022-9.

GIRON, A. A. **Migrating Applications to Post-Quantum Cryptography: Beyond Algorithm Replacement**. 2023. Cryptology ePrint Archive, Paper 2023/709. https://eprint.iacr.org/2023/709. Disponível em: https://eprint.iacr.org/2023/709.

GIRON, A. A.; CUSTÓDIO, R.; RODRÍGUEZ-HENRÍQUEZ, F. Post-quantum hybrid key exchange: a systematic mapping study. **Journal of Cryptographic Engineering**, v. 13, n. 1, p. 71–88, Apr 2023. ISSN 2190-8516. Disponível em: https://doi.org/10.1007/s13389-022-00288-9.

GIRON, A. A.; MARTINA, J. E.; CUSTÓDIO, R. Bitcoin blockchain steganographic analysis. In: ZHOU, J. et al. (Ed.). **Applied Cryptography and Network Security Workshops**. Cham: Springer International Publishing, 2020. p. 41–57. ISBN 978-3-030-61638-0.

GIRON, A. A.; MARTINA, J. E.; CUSTÓDIO, R. Steganographic analysis of blockchains. **Sensors**, v. 21, n. 12, 2021. ISSN 1424-8220. Disponível em: https://www.mdpi.com/1424-8220/21/12/4078.

GIRON, A. A.; NASCIMENTO, J. P. A. do; CUSTóDIO, R. **Quantum-Safe Protection for the TLS Protocol Using Wrapped Certificates**. 2023. Available at SSRN: https://ssrn.com/abstract=4536384 or. http://dx.doi.org/10.2139/ssrn.4536384. Disponível em: http://dx.doi.org/10.2139/ssrn.4536384.

GIRON, A. A. et al. **Post-Quantum Hybrid KEMTLS Performance in Simulated and Real Network Environments**. 2022. Cryptology ePrint Archive, Paper 2022/1639. https://eprint.iacr.org/2022/1639. Disponível em: https://eprint.iacr.org/2022/1639.

GREEN, D. **Pyshark**. 2022. Available at: https://kiminewt.github.io/pyshark/. Accessed on 2022-07-13.

GROVER, L. K. A fast quantum mechanical algorithm for database search. In: **Proceedings of the twenty-eighth annual ACM symposium on Theory of computing**. [S.l.: s.n.], 1996. p. 212–219.

GüNEYSU, T. et al. Proof-of-possession for kem certificates using verifiable generation. In: . New York, NY, USA: Association for Computing Machinery, 2022. (CCS '22). ISBN 9781450394505. Disponível em: https://doi.org/10.1145/3548606.3560560.

HEESCH, M. van et al. **Towards Quantum-Safe VPNs and Internet.** 2019. Cryptology ePrint Archive, Report 2019/1277.

HEIDER, T. **Towards a Verifiably Secure Quantum-Resistant Key Exchange in IKEv2**. Dissertação (Mestrado) — Ludwig Maximilian University of Munich, 2019.

HEMMINGER, S. **Linux Network Emulator**. 2011. Online. https://www.linux.org/docs/man8/tc-netem.html.

HERZINGER, D.; GAZDAG, S.-L.; LOEBENBERGER, D. Real-world quantum-resistant ipsec. In: **2021 14th International Conference on Security of Information and Networks (SIN)**. [S.l.: s.n.], 2021. v. 1, p. 1–8.

HESAMIAN, S. **Analysis of BCNS and NewHope Key-Exchange Protocols**. Dissertação (Mestrado) — University of Wisconsin-Milwaukee, 2017.

HOUSLEY, R. **TLS 1.3 Extension for Certificate-Based Authentication with an External Pre-Shared Key**. [S.l.], 2020.

HOWE, J.; PREST, T.; APON, D. Sok: How (not) to design and implement post-quantum cryptography. In: PATERSON, K. G. (Ed.). **Topics in Cryptology – CT-RSA 2021**. Cham: Springer International Publishing, 2021. p. 444–477. ISBN 978-3-030-75539-3.

HUGUENIN-DUMITTAN, L.; VAUDENAY, S. Fo-like combiners and hybrid post-quantum cryptography. In: CONTI, M.; STEVENS, M.; KRENN, S. (Ed.). **Cryptology and Network Security**. Cham: Springer International Publishing, 2021. p. 225–244. ISBN 978-3-030-92548-2.

HUSAIN, S. S. et al. Ultra-high reliable 5g v2x communications. **IEEE Communications Standards Magazine**, v. 3, n. 2, p. 46–52, 2019.

IEEE. IEEE standard for wireless access in vehicular environments (wave)–certificate management interfaces for end entities. **IEEE Std 1609.2.1-2020**, p. 1–287, 2020.

JONES, M.; BRADLEY, J.; SAKIMURA, N. **JSON Web Signature (JWS)**. [S.l.], 2015. http://www.rfc-editor.org/rfc/rfc7515.txt. Disponível em: http://www.rfc-editor.org/rfc/rfc7515.txt.

JOSEPH, D. et al. Transitioning organizations to post-quantum cryptography. **Nature**, v. 605, n. 7909, p. 237–243, May 2022. ISSN 1476-4687. Disponível em: https://doi.org/10.1038/s41586-022-04623-2.

KADHIM, A. J.; SENO, S. A. H. Energy-efficient multicast routing protocol based on sdn and fog computing for vehicular networks. **Ad Hoc Networks**, v. 84, p. 68–81, 2019. ISSN 1570-8705. Disponível em: https://www.sciencedirect.com/science/article/pii/S1570870518301616.

KAMPANAKIS, P. **Post-Quantum TLS 1.3 and SSH Performance (preliminary results)**. 2020. Https://blogs.cisco.com/security/tls-ssh-performance-pq-kem-auth.

KAMPANAKIS, P.; LEPOINT, T. **Do we need to change some things? Open questions posed by the upcoming post-quantum migration to existing standards and deployments**. 2023. Cryptology ePrint Archive, Paper 2023/266. https://eprint.iacr.org/2023/266. Disponível em: https://eprint.iacr.org/2023/266.

KAUFMAN, C. et al. **Internet Key Exchange Protocol Version 2 (IKEv2)**. [S.l.], 2014. http://www.rfc-editor.org/rfc/rfc7296.txt. Disponível em: http://www.rfc-editor.org/rfc/rfc7296.txt.

KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, O. P. Using mapping studies as the basis for further research–a participant-observer case study. **Information and Software Technology**, Elsevier, v. 53, n. 6, p. 638–651, 2011.

KOCK, B. de. **A Non-Interactive Key Exchange based on Ring-Learning With Errors**. Dissertação (Mestrado) — Eindhoven University of Technology, 2018.

KRAWCZYK, H. Cryptographic extraction and key derivation: The hkdf scheme. In: SPRINGER. **Annual Cryptology Conference**. [S.l.], 2010. p. 631–648.

KRAWCZYK, H.; WEE, H. The optls protocol and tls 1.3. In: IEEE. **2016 IEEE European Symposium on Security and Privacy (EuroS&P)**. [S.l.], 2016. p. 81–96.

KUROSAWA, K.; DESMEDT, Y. A new paradigm of hybrid encryption scheme. In: FRANKLIN, M. (Ed.). **Advances in Cryptology – CRYPTO 2004**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 426–442. ISBN 978-3-540-28628-8.

Let's Encrypt. **Challenge Types**. 2020. https://letsencrypt.org/docs/challenge-types/.

Let's Encrypt. **ACME Client Implementations**. 2022. https://letsencrypt.org/docs/client-options/.

LONG, L. et al. **Maximum lifetime for user ticket renewal**. 2023. https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/maximum-lifetime-for-user-ticket-renewal.

MARCHIORI, D. et al. Timing analysis of algorithm substitution attacks in a post-quantum tls protocol. In: **Anais do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**. Porto Alegre, RS, Brasil: SBC, 2021. p. 127–140. Disponível em: https://sol.sbc.org.br/index.php/sbseg/article/view/17311.

MARSH, S. P. **Formalising trust as a computational concept**. Tese (Doutorado) — University of Stirling, 1994.

MOODY, D. **Let's Get Ready to Rumble- The NIST PQC Competition**. 2018. https://csrc.nist.gov/CSRC/media/Presentations/Let-s-Get-Ready-to-Rumble-The-NIST-PQC-Competiti/images-media/PQCrypto-April2018_Moody.pdf.

MOSCA, M.; PIANI, M. **Quantum Threat Timeline Report**. 2022. Available at: https://globalriskinstitute.org/publication/2022-quantum-threat-timeline-report/. Accessed on 18.04.2023.

NIST. **Post-quantum cryptography**. 2016. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography. Acessed: 2020-06-26.

OPENSSH. **OpenSSH 9.0 Release notes**. 2022. https://www.openssh.com/txt/release-9.0.

OTT, D.; PEIKERT, C.; and other workshop participants. **Identifying Research Challenges in Post Quantum Cryptography Migration and Cryptographic Agility**. 2019.

OUNSWORTH, M. **PQC at the IETF**. 2023. Https://pkic.org/events/2023/post-quantum-cryptography-conference/pkic-pqcc-pqc-at-ietf-mike-ounsworth-entrust.pdf.

OUNSWORTH, M.; PALA, M. **Composite Keys and Signatures For Use In Internet PKI**. 2019. https://datatracker.ietf.org/doc/html/draft-ounsworth-pq-composite-sigs-01. Internet-Draft.

PAAR, C.; PELZL, J. **Understanding cryptography: a textbook for students and practitioners**. Berlin, Heidelberg: Springer Science & Business Media, 2009.

PAQUIN, C.; STEBILA, D.; TAMVADA, G. Benchmarking post-quantum cryptography in tls. In: DING, J.; TILLICH, J.-P. (Ed.). **Post-Quantum Cryptography**. Cham: Springer International Publishing, 2020. p. 72–91. ISBN 978-3-030-44223-1.

PARACHA, M. T. et al. Iotls: Understanding tls usage in consumer iot devices. In: **21st ACM Internet Measurement Conference**. New York, NY, USA: Association for Computing Machinery, 2021. (IMC '21). ISBN 9781450391290. Disponível em: https://doi.org/10.1145/3487552.3487830.

PARTALA, J. Provably secure covert communication on blockchain. **Cryptography**, Multidisciplinary Digital Publishing Institute, v. 2, n. 3, p. 18, 2018.

PAUL, S.; SCHEIBLE, P. Towards post-quantum security for cyber-physical systems: Integrating pqc into industrial m2m communication. In: CHEN, L. et al. (Ed.). **Computer Security – ESORICS 2020**. Cham: Springer International Publishing, 2020. p. 295–316. ISBN 978-3-030-59013-0.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, Elsevier, v. 64, p. 1–18, 2015.

PLOTLY. **Dash Documentation and User Guide**. 2022. Available at: https://dash.plotly.com/. Accessed on 2022-07-13.

PROJECT, O. Q. S. **liboqs-go: Go bindings for liboqs**. 2022. Available at: https://github.com/open-quantum-safe/liboqs-go. [Online; accessed 25-Jan-2022].

RESCORLA, E. **The Transport Layer Security (TLS) Protocol Version 1.3**. [S.l.], 2018.

RESCORLA, E. et al. **TLS Encrypted Client Hello**. [S.l.], 2022. https://www.ietf.org/archive/id/draft-ietf-tls-esni-14.txt. Disponível em: https://www.ietf.org/archive/id/draft-ietf-tls-esni-14.txt.

RESTREPO, R. **OAuth 2.0 Refresh Token Best Practices**. 2022. https://stateful.com/blog/oauth-refresh-token-best-practices.

RNP. **Diploma Digital**. 2023. Available at: https://www.nasnuvens.rnp.br/diploma-digital. Accessed on 18.03.2023.

RUDOLPH, H. C.; GRUNDMANN, N. **Ciphersuite Info**. 2022. Available at: https://ciphersuite.info/. Accessed on 2022-07-12.

SAARINEN, M. O. Mobile energy requirements of the upcoming nist post-quantum cryptography standards. In: **2020 8th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)**. Oxford, GB, United Kingdom: IEEE, 2020. p. 23–30.

SAKIMURA, N. et al. **OpenID Connect Core 1.0**. [S.l.], 2023. Available at https://openid.net/specs/openid-connect-core-1_0.html. Accessed March, 2023.

SANTESSON, S.; TSCHOFENIG, H. **Transport Layer Security (TLS) Cached Information Extension**. [S.l.], 2016.

SCHANCK, J. M.; WHYTE, W.; ZHANG, Z. Circuit-extension handshakes for tor achieving forward secrecy in a quantum world. **Proceedings on Privacy Enhancing Technologies**, Sciendo, Berlin, v. 2016, n. 4, p. 219 – 236, 2016. Disponível em: https://content.sciendo.com/view/journals/popets/2016/4/article-p219.xml.

SCHARDONG, F. et al. Post-quantum electronic identity: Adapting openid connect and oauth 2.0 to the post-quantum era. In: BERESFORD, A. R.; PATRA, A.; BELLINI, E. (Ed.). **Cryptology and Network Security**. Cham: Springer International Publishing, 2022. p. 371–390. ISBN 978-3-031-20974-1.

SCHWABE, P.; STEBILA, D.; WIGGERS, T. Post-quantum tls without handshake signatures. In: **Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2020. (CCS '20), p. 1461–1480. ISBN 9781450370899. Disponível em: https://doi.org/10.1145/3372297.3423350.

SCHWABE, P.; STEBILA, D.; WIGGERS, T. Post-quantum tls without handshake signatures. **IACR Cryptol. ePrint Arch.**, v. 2020, p. 534, 2020.

SCHWABE, P.; STEBILA, D.; WIGGERS, T. More efficient post-quantum kemtls with pre-distributed public keys. In: BERTINO, E.; SHULMAN, H.; WAIDNER, M. (Ed.). **Computer Security – ESORICS 2021**. Cham: Springer International Publishing, 2021. p. 3–22. ISBN 978-3-030-88418-5.

SHOR, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In: IEEE. **Proceedings 35th annual symposium on foundations of computer science**. Santa Fe, NM, USA: IEEE, 1994. p. 124–134.

SIKERIDIS, D. et al. Intermediate certificate suppression in post-quantum tls: An approximate membership querying approach. In: . New York, NY, USA: Association for Computing Machinery, 2022. (CoNEXT '22), p. 35–42. ISBN 9781450395083. Disponível em: https://doi.org/10.1145/3555050.3569127.

SIKERIDIS, D.; KAMPANAKIS, P.; DEVETSIKIOTIS, M. Assessing the overhead of post-quantum cryptography in tls 1.3 and ssh. In: **Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies**. New York, NY, USA: Association for Computing Machinery, 2020. p. 149–156.

STADLER, S. et al. **Hybrid Signal protocol for post-quantum email encryption**. 2021. Cryptology ePrint Archive, Paper 2021/875. https://eprint.iacr.org/2021/875. Disponível em: https://eprint.iacr.org/2021/875.

STEBILA, D.; FLUHRER, S.; GUERON, S. **Hybrid key exchange in TLS 1.3**. 2023. http://tools.ietf.org/html/draft-ietf-tls-hybrid-design-06. Internet-Draft.

STEBILA, D.; MOSCA, M. Post-quantum key exchange for the internet and the open quantum safe project. In: SPRINGER. **International Conference on Selected Areas in Cryptography**. [S.l.], 2016. p. 14–37.

TRIDGELL, A.; MACKERRAS, P.; DAVISON, W. **rsync - a fast, versatile, remote (and local) file-copying tool**. [S.l.], 2022. Available at https://download.samba.org/pub/rsync/rsync.1. Accessed Apr, 2023.

TUJNER, Z. et al. **QSOR: Quantum-Safe Onion Routing**. 2020.

UNGER, N.; GOLDBERG, I. Improved strongly deniable authenticated key exchanges for secure messaging. **Proceedings on Privacy Enhancing Technologies**, Sciendo, Berlin, v. 2018, n. 1, p. 21 – 66, 2018. Disponível em: https://content.sciendo.com/view/journals/popets/2018/1/article-p21.xml.

VELÁZQUEZ, J. A. S. **Practical Implementations of Quantum-Resistant Cryptography**. [S.l.], 2017.

W3C, W. **WebRTC: Real-Time Communication in Browsers**. [S.l.], 2023. Available at https://www.w3.org/TR/webrtc/. Accessed March, 2023.

WESTERBAAN, B. **Sizing Up Post-Quantum Signatures**. 2021. Https://blog.cloudflare.com/sizing-up-post-quantum-signatures/.

WORMLY, I. **Free SSL Web Server Tester**. 2022. Available at: https://www.wormly.com/test_ssl. Accessed on 2022-08-27.

XU, D. et al. Understanding operational 5g: A first measurement study on its coverage, performance and energy consumption. In: **Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication**. [S.l.: s.n.], 2020. p. 479–494.

XU, J.; GAO, Y.; LIM, H. **Practical Quantum-Safe Stateful Hybrid Key Exchange Protocol**. 2020. Cryptology ePrint Archive, Report 2020/763.

YLONEN, T.; LONVICK, C. **The Secure Shell (SSH) Protocol Architecture**. [S.l.], 2006. http://www.rfc-editor.org/rfc/rfc4251.txt. Disponível em: http://www.rfc-editor.org/rfc/rfc4251.txt.