



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

Thiago Leucz Astrizi

**Post-Quantum Cryptographic Adaptations for Legacy Systems: Methodologies  
for Blockchain Redactability and TLS Protocol Evolution**

Florianópolis  
2025

Thiago Leucz Astrizi

**Post-Quantum Cryptographic Adaptations for Legacy Systems: Methodologies  
for Blockchain Redactability and TLS Protocol Evolution**

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

Florianópolis  
2025

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.  
Dados inseridos pelo próprio autor.

Astrizi, Thiago Leucz  
Post-Quantum Cryptographic Adaptations for Legacy  
Systems: Methodologies for Blockchain Redactability and  
TLS Protocol Evolution / Thiago Leucz Astrizi ;  
orientador, Ricardo Felipe Custódio, 2025.  
137 p.

Tese (doutorado) - Universidade Federal de Santa  
Catarina, Centro Tecnológico, Programa de Pós-Graduação em  
Ciência da Computação, Florianópolis, 2025.

Inclui referências.

1. Ciência da Computação. 2. Criptografia Pós-Quântica. 3.  
Funções Hash Camaleão. 4. Transport Layer Security. 5.  
Criptografia Baseada em Identidade. I. Custódio, Ricardo  
Felipe. II. Universidade Federal de Santa Catarina.  
Programa de Pós-Graduação em Ciência da Computação. III.  
Título.

Thiago Leucz Astrizi

**Post-Quantum Cryptographic Adaptations for Legacy Systems: Methodologies  
for Blockchain Redactability and TLS Protocol Evolution**

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

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

Prof. Diego de Freitas Aranha, Dr.  
Universidade de Aarhus

Prof. Roberto Samarone dos Santos Araújo, Dr.  
Universidade Federal do Pará

Prof. Alexandre Augusto Giron, Dr.  
Universidade Tecnológica Federal do Paraná

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi  
julgada adequada para obtenção do título de Doutor em Ciência da Computação.



Documento assinado digitalmente  
**Marcio Bastos Castro**  
Data: 08/09/2025 11:45:58-0300  
CPF: \*\*\*.707.720-\*\*  
Verifique as assinaturas em <https://v.ufsc.br>

---

Coordenação do Programa de  
Pós-Graduação



Documento assinado digitalmente  
**Ricardo Felipe Custodio**  
Data: 08/09/2025 07:41:56-0300  
CPF: \*\*\*.981.609-\*\*  
Verifique as assinaturas em <https://v.ufsc.br>

---

Prof. Ricardo Custódio, Dr.  
Orientador



## **AGRADECIMENTOS**

O presente trabalho foi realizado com o apoio da Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina (FAPESC), por meio do edital de chamada pública CP 48/2021.

Agradeço profundamente a todas as pessoas que, de diferentes formas, contribuíram para que eu pudesse me dedicar aos estudos durante o doutorado. À Ana Leuch Lozovei, por sua constante ajuda e encorajamento ao longo de toda a pós-graduação. A Bruno Ricci Piacenti dos Santos e Gabriela Machado, pela amizade e apoio na vida pessoal, especialmente nos momentos em que precisei me ausentar, cuidando com carinho da minha gata Frida, cuja presença, por sua vez, foi essencial para manter minha saúde mental e serenidade.

Expresso também minha gratidão aos colegas e colaboradores da Universidade Federal de Santa Catarina, em especial aos membros do Laboratório de Segurança em Computação (LabSEC), que contribuíram com ideias, conselhos, estímulos e auxílios práticos em diversas fases da pesquisa. Ao meu orientador, Prof. Ricardo Custódio, por tornar este trabalho possível, sempre oferecendo orientações valiosas voltadas tanto ao crescimento acadêmico quanto ao desenvolvimento pessoal. À Lucila Alosilla e à Simone Cruz, pelo imprescindível suporte administrativo na secretaria do laboratório. Aos colegas Alexandre Augusto Giron, Frederico Schardong e Wellington Fernandes Silvano, por suas colaborações e companheirismo ao longo dessa jornada.

Por fim, agradeço ao Technology Innovation Institute (TII) e a seus pesquisadores que, por meio da parceria de pesquisa envolvendo a segunda parte desta tese sobre TLS, ofereceram críticas construtivas, sugestões e apoio técnico. Em especial, agradeço a Lucas Perin e Victor Mateu pelas contribuições diretas e pela colaboração generosa.

*"The chameleon teaches us that to change is not to hide, but to survive with elegance. In cryptography,  
as in life, adaptability is the subtlest of strengths."  
- Anonymous*

## RESUMO

Esta tese de doutorado se propõe a atacar o problema da dificuldade de migração de tecnologias clássicas para pós-quânticas, e para isso se desenvolve em duas linhas de pesquisa complementares. A primeira contribuição apresenta um framework teórico e implementação prática para construção de blockchains redatáveis pós-quânticos utilizando mecanismos especializados de encadeamento baseados em funções hash camaleão com propriedades de pré-imagem. Esta abordagem permite mutabilidade controlada enquanto mantém a integridade criptográfica, solucionando limitações-chave em frameworks blockchain existentes. A segunda contribuição aborda os desafios da transição do protocolo Transport Layer Security (TLS) 1.3 para segurança pós-quântica sem interromper a infraestrutura existente. Ao integrar estrategicamente criptografia baseada em identidade (IBE) com criptografia clássica, propomos um mecanismo híbrido que facilita trocas de chaves pós-quânticas preservando compatibilidade com as cadeias de certificados e mecanismos de autenticação atuais. Avaliações experimentais rigorosas conduzidas em diversos ambientes de rede demonstram a viabilidade e características de desempenho de ambas as propostas: nossa blockchain redatável mostrou-se mais rápida que outras alternativas e a proposta para TLS obteve desempenho competitivo em relação a outros algoritmos em testes de alta largura de banda e baixa latência. As metodologias apresentadas foram implementadas como soluções de código aberto para incentivar pesquisas futuras e adoção prática no campo da criptografia pós-quântica.

**Palavras-chave:** Criptografia Pós-Quântica, Funções Hash Camaleão, Transport Layer Security, Criptografia Baseada em Identidade, Redatabilidade em Blockchain, Migração de Protocolos Criptográficos, Criptossistemas Híbridos,

## RESUMO EXPANDIDO

### INTRODUÇÃO

A criptografia de chave pública é um dos pilares fundamentais da comunicação segura na Internet e em diversas redes computacionais. A maioria dos sistemas modernos de chave pública baseia-se na dificuldade computacional de fatorar números semiprimos ou resolver o problema do logaritmo discreto em grupos de curvas elípticas. Ao longo das últimas décadas, esses dois problemas matemáticos forneceram não apenas a base para a segurança, mas também sustentação para algoritmos eficientes e implementações robustas de primitivas criptográficas.

A complexidade computacional de fatorar números e calcular logaritmos discretos permaneceu incontestável durante décadas frente às análises e algoritmos clássicos. Entretanto, na década de 1990, Peter Shor apresentou um algoritmo quântico revolucionário capaz de fatorar números semiprimos em tempo polinomial (SHOR, 1994) usando técnica que posteriormente foi também adaptada para resolver o problema do logaritmo discreto em curvas elípticas (SHOR, 1999). Este algoritmo fundamenta-se em uma máquina de Turing probabilística ampliada com uma fita quântica, viabilizando operações baseadas em circuitos quânticos. O trabalho de Shor também estabeleceu a classe de complexidade BQP (*Bounded-Error Quantum Polynomial*), que compreende problemas de decisão solucionáveis por um computador quântico em tempo polinomial, com probabilidade de erro máxima de  $1/3$  para quaisquer instâncias.

Nas décadas subsequentes, avanços em nanotecnologia e engenharia possibilitaram a construção de computadores quânticos em escala reduzida, demonstrando a viabilidade dos algoritmos quânticos. Tais desenvolvimentos indicam que a computação quântica pode representar uma ameaça significativa à criptografia no futuro. O potencial surgimento de Computadores Quânticos Relevantes para a Criptografia (CRQCs), capazes de implementar o algoritmo de Shor ou outros algoritmos quânticos, poderia comprometer a segurança dos sistemas criptográficos contemporâneos (uma visão geral das perspectivas de diversos pesquisadores sobre o cronograma esperado para o desenvolvimento de tais computadores pode ser encontrada em (MOSCA; PIANI, 2022)).

Para determinadas tecnologias, antecipar ameaças futuras e implementar medidas de proteção preventivas torna-se essencial. Caso um CRQC se materialize nos próximos 20 anos, um atacante poderia interceptar e armazenar comunicações criptografadas atuais para descriptografá-las posteriormente, em um ataque denominado “coletar agora, descriptografar depois” (*harvest now, decrypt later*). Esta ameaça é particularmente relevante para dados que necessitam permanecer seguros por períodos prolongados, como informações armazenadas em blockchains ou arquivos de longo prazo. Para se proteger dela, a migração para algoritmos pós-quânticos—algoritmos

criptográficos que, até onde se sabe, permanecem seguros mesmo diante de computadores quânticos—precisa ser feita com uma antecedência bem maior do que o surgimento do primeiro CRQC.

Contudo, migrar para algoritmos pós-quânticos constitui uma tarefa complexa. A criptografia fundamentada em logaritmos discretos e suposições de fatoração é geralmente mais eficiente e produz chaves e textos cifrados menores. A transição para algoritmos pós-quânticos frequentemente resulta em tempos de processamento mais extensos, maior volume de transmissão de dados e requisitos de armazenamento mais elevados, apresentando desafios significativos, especialmente para dispositivos com recursos limitados. Adicionalmente, em protocolos amplamente utilizados, a adoção de algoritmos pós-quânticos demanda mudanças substanciais na infraestrutura, incluindo modificações em cadeias de certificação e autoridades certificadoras.

## MOTIVAÇÃO

Este trabalho busca responder à questão fundamental de como simplificar a adoção e a transição para algoritmos pós-quânticos em aplicações criptográficas, especialmente para autenticação e integridade. Nosso objetivo consiste em oferecer novos métodos para mitigar os desafios da migração pós-quântica, explorando primitivas criptográficas menos convencionais, como funções hash camaleão (funções hash que possuem um segredo associado que permite que apenas quem o conhece seja capaz de gerar colisões nela), e aproveitando a criptografia baseada em identidade para eliminar a necessidade de distribuição ou certificação de chaves pós-quânticas.

Entre os cenários que motivam esta pesquisa está o interesse em adaptar blockchains redatáveis ao contexto pós-quântico, preservando sua flexibilidade e garantindo segurança a longo prazo. Blockchains redatáveis dependem criticamente de funções hash camaleão para permitir modificações controladas de dados históricos sem comprometer a integridade. Isso gera um desafio significativo no contexto pós-quântico, pois atualmente há uma escassez de construções de hash camaleão que sejam simultaneamente resistentes a ataques quânticos e capazes de atender aos rigorosos requisitos de segurança desses sistemas. As soluções existentes ou são vulneráveis a adversários quânticos, ou não fornecem a resistência a colisões e as propriedades de armadilha necessárias. Essa lacuna motivou a busca por novas construções criptográficas voltadas a viabilizar a redatabilidade em sistemas de blockchain sob pressupostos pós-quânticos.

Esta pesquisa também foi motivada por evidências que indicam que a maior sobrecarga na migração de protocolos como SSH e TLS para versões pós-quânticas concentra-se em seus componentes de autenticação, como trocas de chaves e assinaturas (SIKERIDIS *et al.*, 2020). Ao explorar primitivas criptográficas alternativas, como funções hash camaleão, identificamos construções práticas com propriedades

adicionais que podem ser utilizadas para desenvolver novas técnicas e resultados criptográficos (ASTRIZI, T. et al., 2020). Consequentemente, investigamos se essas propriedades poderiam ser aplicáveis em protocolos e cenários mais complexos.

Artigos como (BANERJEE; CHANDRAKASAN, 2020) evidenciaram a necessidade de reavaliar quais primitivas são viáveis para utilização em protocolos consolidados. Por exemplo, a criptografia baseada em identidade (IBE) foi tradicionalmente considerada menos atrativa que os algoritmos de chave pública, pois embora algumas propostas, como em (PENG et al., 2009) mostrassem que ela poderia permitir uma economia no envio de dados pela rede, a demanda computacional para executar os algoritmos era muito maior. Entretanto, no contexto dos algoritmos pós-quânticos, a sobrecarga computacional do IBE é comparativamente moderada, enquanto sua redução na transmissão de dados é significativamente superior, conforme demonstrado por tal artigo. Embora esse estudo tenha focalizado algoritmos iniciais e tenha sido limitado a cenários de Internet das Coisas (IoT), seus resultados foram promissores e inspiraram uma investigação mais aprofundada de soluções baseadas em IBE neste trabalho.

Com o possível advento de Computadores Quânticos Relevantes para a Criptografia, observa-se um crescente interesse em migrar protocolos existentes para versões pós-quânticas. Contudo, essa transição enfrenta desafios significativos, incluindo a ineficiência dos algoritmos pós-quânticos atuais e as complexidades da migração em larga escala. Esta tese visa abordar essas questões propondo soluções alternativas para simplificar e aprimorar o processo de migração.

## OBJETIVOS

Para refinar nossas propostas e fundamentar nossa análise em exemplos concretos, selecionamos dois protocolos, o Protocolo Backbone do Bitcoin (definido em (GARAY et al., 2015), e utilizado como uma estrutura formal para analisar operações gerais de blockchain, não se limitando a aplicações em criptomoedas) e o TLS 1.3, com o objetivo de avaliar os desafios impostos pelos algoritmos pós-quânticos e sua integração em protocolos de comunicação amplamente utilizados. Embora esses protocolos sejam usados como estudos de caso, nosso objetivo é desenvolver soluções que possam ser generalizadas para outros protocolos e domínios, assegurando uma aplicabilidade mais ampla.

Portanto, os objetivos desta tese são:

- Identificar desafios na adoção de algoritmos pós-quânticos para os protocolos escolhidos por meio de uma revisão sistemática da literatura;
- Mitigar os problemas identificados com algoritmos pós-quânticos, utilizando os mecanismos mais simples e eficazes sempre que possível;

- Fornecer contribuições teóricas com reduções de segurança rigorosas para nossas propostas, garantindo que quaisquer desvios em relação aos protocolos originais não introduzam novas vulnerabilidades;
- Discutir a aplicabilidade de nossas propostas e soluções a outros protocolos ou problemas relevantes.

Esses objetivos estão alinhados com a meta geral da tese de oferecer soluções mais abrangentes, porém com foco em protocolos específicos para uma análise mais detalhada e concreta.

## METODOLOGIA

A metodologia deste trabalho está dividida em três partes. Como contribuição inicial, o Capítulo 3 apresenta um mapeamento sistemático para estudar as definições, propriedades, suposições de segurança e casos de uso das funções hash camaleão na literatura. Essa metodologia visa pesquisar, classificar e fornecer novas perspectivas sobre o conhecimento existente nesse tema. Esse estudo serviu como ponto de partida para nossa pesquisa, expandindo o trabalho anterior do autor durante seu mestrado. Especificamente, focamos em construções de funções hash camaleão com propriedades não convencionais, como a capacidade de calcular primeiras pré-imagens quando um segredo gerado na inicialização da primitiva é conhecido, em vez de apenas colisões regulares, e em particular nas construções pós-quânticas de funções hash camaleão.

Na segunda parte desta tese, apresentada no Capítulo 4, analisamos o protocolo Bitcoin. Em vez de realizar um mapeamento sistemático sobre esse tema, baseamo-nos em levantamentos existentes, como (ZHANG, D. *et al.*, 2021), que destacaram desafios relacionados à imutabilidade das blockchains. Embora essa propriedade seja fundamental para a tecnologia blockchain, ela nem sempre é adequada para todas as aplicações e pode expor o sistema a ataques em que usuários mal-intencionados carregam conteúdo ilegal na rede blockchain.

Com base nas informações do levantamento e na análise do protocolo Bitcoin, propomos uma nova construção para blockchains utilizando funções hash camaleão para um encadeamento alternativo. Essa abordagem permite a criação de blockchains redatáveis, que são mais simples e mais rápidas do que os designs existentes. Para validar nossa proposta, implementamos a construção e conduzimos experimentos comparando o desempenho das operações de hash, verificação e redação com propostas semelhantes baseadas em funções hash camaleão presentes na literatura. Os resultados, apresentados em tabelas, incluem a média e a margem de erro para cada caso, assumindo uma distribuição Gaussiana.

Na terceira e última parte desta tese, analisamos o protocolo TLS 1.3. Com base em estudos como (SIKERIDIS *et al.*, 2020) e (PAQUIN *et al.*, 2020), focamos nos desafios da transição do TLS 1.3 para uma versão pós-quântica. Esses desafios incluem o impacto significativo no desempenho devido ao aumento do tamanho das chaves e das mensagens TLS exigidas pelos algoritmos pós-quânticos, além da complexidade de migrar um ecossistema que envolve clientes, servidores, autoridades certificadoras, sistemas de validação de certificados, verificações de revogação e confiança nas autoridades certificadoras raiz.

Para facilitar essa transição, desenvolvemos uma abordagem inovadora que evita a introdução de chaves pós-quânticas na infraestrutura existente. Em vez disso, aproveitamos a criptografia baseada em identidade para integrar segurança pós-quântica ao protocolo sem modificar certificados atuais, protocolos de certificação ou mecanismos de revogação. Para avaliar a viabilidade e o desempenho da nossa solução, medimos os tempos de protocolo tanto em um ambiente simulado quanto em testes reais entre máquinas localizadas em diferentes continentes. Também comparamos esses resultados com outras soluções híbridas existentes.

## RESULTADOS E DISCUSSÃO

As principais contribuições desta tese são:

- Como parte de nossa estratégia para lidar com os desafios pós-quânticos, realizamos um mapeamento sistemático para explorar diferentes tipos de funções hash camaleão, suas suposições de segurança e suas aplicações em diversos problemas. Um Estudo de Mapeamento Sistemático é uma abordagem estruturada para levantar e categorizar a literatura existente com o objetivo de gerar novos entendimentos sobre um determinado tema, em conformidade com uma metodologia de pesquisa rigorosa;
- Propusemos uma nova construção para blockchains redatáveis utilizando funções hash camaleão. Essa construção permite que partes confiáveis redatem blocos em cenários específicos, como a remoção de conteúdo ilegal da blockchain. Além disso, identificamos instâncias em que o tempo de hash foi mais rápido do que construções compatíveis previamente documentadas;
- Avaliamos o desempenho de diversas implementações de blockchains redatáveis baseadas em funções hash camaleão, fornecendo resultados experimentais detalhados.
- Introduzimos um novo protocolo híbrido baseado no KEMTLS, integrando criptografia de chave pública tradicional com criptografia baseada em identidade

pós-quântica. Essa abordagem melhora a segurança, garantindo resiliência contra adversários clássicos e quânticos;

- Testamos nosso protocolo TLS pós-quântico juntamente com outros algoritmos híbridos que estão sendo padronizados pelo NIST. Esses testes foram conduzidos em cenários realistas, utilizando computadores alugados de diferentes empresas em diversos continentes para comparar o desempenho entre máquinas geograficamente distantes;
- Realizamos uma análise de segurança abrangente de nossas propostas. Para o protocolo TLS, desenvolvemos um novo algoritmo híbrido que conecta duas primitivas criptográficas que, embora semelhantes, exigiam formalizações distintas. Essa construção híbrida necessitou da definição de requisitos de segurança realistas, capturando as propriedades únicas de ambas as primitivas.

Por meio das contribuições, obtivemos resultados bastante promissores. Em relação à nossa proposta de blockchain redatável, enquanto as construções prévias apresentavam uma penalidade de cerca de 38% de tempo de hash em relação à uma blockchain tradicional não-redatável, fomos capazes de construir uma versão pós-quântica que apresentou uma penalidade menor, de aproximadamente 20%. Se usarmos nossa proposta de protocolo com construções clássicas, além das pós-quânticas, um resultado ainda melhor pode ser alcançado, trazendo uma penalidade de apenas 13%.

Se formos comparar não só o tempo de cálculo de hash, mas também o tempo necessário para realizar a redação, a nossa construção pós-quântica também obteve um resultado melhor, sendo cerca de 20% mais rápida em relação à propostas existentes comparáveis.

A única medida em que nossa blockchain redatável apresentou resultados menos vantajosos foi na quantidade de dados adicionais que precisam ser acrescentados ao hash para que a proposta possa ser usada, requerendo cerca de 800 bytes a mais por cada bloco.

Em relação à nossa proposta de protocolo TLS pós-quântico, quando comparado à outros algoritmos híbridos pós quânticos, como resultado notável conseguimos reduzir a quantidade de informação enviada durante o ServerHello, principalmente pelo fato de conseguirmos evitar enviar chaves criptográficas pós-quânticas em nossos certificados. Infelizmente tal economia é perdida com um aumento no tamanho da mensagem subsequente que o cliente precisa enviar para o servidor. Tal aumento, contudo, pode vir a ser evitado caso sejam propostos no futuro mecanismos de encapsulamento pós-quânticos com um tamanho menor para o texto cifrado.

De qualquer forma, embora não esperássemos que nossa construção fosse mais rápida (sua principal vantagem é minimizar a disrupção da infra-estrutura existente durante a migração, não maior velocidade), ainda assim encontramos resultados animadores em nossas simulações e em nossos testes feitos em redes reais. O mecanismo de encapsulamento testado mostrou-se mais rápido em tempo de execução comparado a vários finalistas do NIST. Tal resultado é ainda preliminar e deve ser visto com cautela—estamos medindo códigos com diferentes graus de maturidade. Contudo, esta vantagem em tempo de computação acabou deixando nossa construção mais rápida em cenários de alta largura de banda e baixa latência, onde o tamanho maior de nossas mensagens é menos penalizado.

## ABSTRACT

This doctoral thesis aims to address the challenge of migrating from classical to post-quantum technologies through two complementary research directions. The first contribution introduces a theoretical framework and practical implementation for constructing post-quantum redactable blockchains utilizing specialized chaining mechanisms based on chameleon hash functions with preimage properties. This approach enables controlled mutability while maintaining cryptographic integrity, addressing key limitations in existing blockchain frameworks. The second contribution addresses the challenges of transitioning the Transport Layer Security (TLS) 1.3 protocol to post-quantum security without disrupting existing infrastructure. By strategically integrating identity-based encryption (IBE) with classical cryptography, we propose a hybrid mechanism that facilitates post-quantum key exchanges while preserving compatibility with current certificate chains and authentication mechanisms. Rigorous experimental evaluations conducted across various network environments demonstrate the viability and performance characteristics of both proposals: our redactable blockchain proved to be faster than other alternatives, and the proposed solution for TLS achieved competitive performance compared to other algorithms in high-bandwidth, low-latency tests.. The methodologies presented herein have been implemented as open-source solutions to encourage further research and practical adoption in the field of post-quantum cryptography.

**Keywords:** Post-Quantum Cryptography, Chameleon Hash Functions, Transport Layer Security, Identity-Based Encryption, Blockchain Redactability, Cryptographic Protocol Migration, Hybrid Cryptosystems, .

## LIST OF FIGURES

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Figure 1 – Diagram illustrating the usage of TLS 1.3. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                             | 46  |
| Figure 2 – Diagram with number of works selected by our systematic mapping<br>in each step. . . . .                                                                                                                                                                                                                                                                                                                                                                                       | 55  |
| Figure 3 – Selected publications by year. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                         | 56  |
| Figure 4 – Different chameleon hash scheme definitions by publication. . . . .                                                                                                                                                                                                                                                                                                                                                                                                            | 56  |
| Figure 5 – Different security assumptions by publication. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                         | 58  |
| Figure 6 – Different applications for chameleon hash schemes by publication. . . . .                                                                                                                                                                                                                                                                                                                                                                                                      | 60  |
| Figure 7 – Graphs representing toy examples of chameleon hash functions. Each<br>message $m_i$ , represented by a circle, is connected to a digest $d_i$ by an<br>edge labeled $r_i$ if, for some fixed key $ek$ , we have $CH.Hash(ek, m_i, r_i) =$<br>$d_i$ . In the first image, the chameleon hash does not have the unifor-<br>mity property. In the second image, it satisfies the weak uniformity<br>property. The third image illustrates the strong uniformity property. . . . . | 61  |
| Figure 8 – Before redacting block $B_1$ . . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                         | 82  |
| Figure 9 – After redacting block $B_1$ . . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                          | 82  |
| Figure 10 – Side-by-side comparison between KEMTLS and our proposal . . . . .                                                                                                                                                                                                                                                                                                                                                                                                             | 102 |
| Figure 11 – Detailed view of our proposal mixing KEMTLS and IBE . . . . .                                                                                                                                                                                                                                                                                                                                                                                                                 | 103 |
| Figure 12 – Setup for our local tests in a simulated environment . . . . .                                                                                                                                                                                                                                                                                                                                                                                                                | 106 |
| Figure 13 – Setup for remote experiments over a real network . . . . .                                                                                                                                                                                                                                                                                                                                                                                                                    | 107 |
| Figure 14 – Comparison of our proposal with TLS 1.3 at 0ms latency . . . . .                                                                                                                                                                                                                                                                                                                                                                                                              | 108 |
| Figure 15 – Comparison of our proposal with TLS 1.3 at 1ms latency . . . . .                                                                                                                                                                                                                                                                                                                                                                                                              | 109 |
| Figure 16 – Comparison of our proposal with TLS 1.3 at 5ms latency . . . . .                                                                                                                                                                                                                                                                                                                                                                                                              | 110 |
| Figure 17 – Comparison of our proposal with TLS 1.3 at 50ms latency . . . . .                                                                                                                                                                                                                                                                                                                                                                                                             | 111 |
| Figure 18 – Comparison of our proposal with TLS 1.3 at 150ms latency . . . . .                                                                                                                                                                                                                                                                                                                                                                                                            | 112 |
| Figure 19 – Comparison of our proposal with TLS 1.3 over a real network . . . . .                                                                                                                                                                                                                                                                                                                                                                                                         | 113 |
| Figure 20 – Comparison of our proposal with other hybrid algorithms (low latency<br>scenarios) . . . . .                                                                                                                                                                                                                                                                                                                                                                                  | 114 |
| Figure 21 – Comparison of our proposal with other hybrid algorithms (high latency<br>and real network scenarios) . . . . .                                                                                                                                                                                                                                                                                                                                                                | 115 |
| Figure 22 – Comparison of our proposal with other hybrid algorithms using pre-<br>distributed keys (low latency scenarios) . . . . .                                                                                                                                                                                                                                                                                                                                                      | 116 |
| Figure 23 – Comparison of our proposal with other hybrid algorithms using pre-<br>distributed keys (high latency and real network scenarios) . . . . .                                                                                                                                                                                                                                                                                                                                    | 117 |

## LIST OF TABLES

|                                                                                                                                                        |     |
|--------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Table 1 – Testing existing constructions proposed in the literature . . . . .                                                                          | 86  |
| Table 2 – Testing our proposed construction with different preimage chameleon<br>hash functions . . . . .                                              | 87  |
| Table 3 – Handshake message sizes (in bytes). . . . .                                                                                                  | 106 |
| Table 4 – Handshake timings in milliseconds. . . . .                                                                                                   | 107 |
| Table 5 – Comparing our Hybrid KEMTLS with traditional Hybrid KEMTLS . . .                                                                             | 109 |
| Table 6 – Handshake time for various hybrid KEMTLS configurations . . . . .                                                                            | 111 |
| Table 7 – Handshake size (in bytes) comparing our Hybrid KEMTLS construction<br>with traditional Hybrid KEMTLS protocols using pre-distributed keys. . | 112 |
| Table 8 – Handshake time in milliseconds when server keys were pre-distributed.                                                                        | 113 |

## LIST OF ABBREVIATIONS

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>BQP</b>            | Bounded-Error Quantum Polynomial                    |
| <b>CA</b>             | Certification Authority                             |
| <b>CDH</b>            | Computational Diffie-Hellman                        |
| <b>CH</b>             | Chameleon Hash                                      |
| <b>CRQC</b>           | Cryptographically Relevant Quantum Computer         |
| <b>CSR</b>            | Certificate Signing Request                         |
| <b>DDH</b>            | Decisional Diffie-Hellman                           |
| <b>DHKEM</b>          | Diffie-Hellman KEM                                  |
| <b>DLP-IBE</b>        | Ducas-Lyubashevsky-Prest IBE                        |
| <b>ECDSA</b>          | Elliptic Curve Digital Signature Algorithm          |
| <b>HKEM</b>           | Hybrid KEM                                          |
| <b>IBE</b>            | Identity-Based Encryption                           |
| <b>IDKEM</b>          | Identity-based KEM                                  |
| <b>IND-CCA</b>        | Indistinguishability under Chosen Ciphertext Attack |
| <b>IoT</b>            | Internet of Things                                  |
| <b>KEM</b>            | Key Encapsulation Mechanism                         |
| <b>KEMTLS</b>         | KEM-based Transport Layer Security                  |
| <b>KEMTLS-IBE</b>     | KEMTLS with Identity-Based Encryption               |
| <b>KEMTLS-IBE-MPK</b> | KEMTLS with IBE and Master Public Key transmission  |
| <b>KEMTLS-IBE-PDK</b> | KEMTLS with IBE and Pre-Distributed Keys            |
| <b>MAC</b>            | Message Authentication Code                         |
| <b>NIST</b>           | National Institute of Standards and Technology      |
| <b>OCSP</b>           | Online Certificate Status Protocol                  |
| <b>PKG</b>            | Private Key Generator                               |
| <b>PKI</b>            | Public Key Infrastructure                           |
| <b>R-SIS</b>          | Ring-Short Integer Solution                         |
| <b>SCH</b>            | Secret-coin Chameleon Hash                          |
| <b>SIG</b>            | Signature Scheme                                    |
| <b>SMS</b>            | Systematic Mapping Study                            |
| <b>TLS</b>            | Transport Layer Security                            |
| <b>UF-CMA</b>         | Unforgeability under Chosen Message Attacks         |

## LIST OF SYMBOLS

|                              |                                                                             |
|------------------------------|-----------------------------------------------------------------------------|
| $\lambda$                    | Security parameter                                                          |
| $\leftarrow$                 | Assignment operator                                                         |
| $\$ \leftarrow$              | Random sampling operator                                                    |
| $\mathbb{Z}_q$               | Integers modulo $q$                                                         |
| $\mathbb{G}$                 | Multiplicative group (typically derived from an elliptic curve)             |
| $g$                          | Generator of a multiplicative group                                         |
| $q$                          | Order of a group                                                            |
| $\mathcal{M}$                | Message space                                                               |
| $\mathcal{R}$                | Randomness space                                                            |
| $\mathcal{D}$                | Digest space                                                                |
| $\mathcal{K}$                | Key space                                                                   |
| $\mathcal{C}$                | Ciphertext space                                                            |
| $\mathcal{T}$                | Tag space                                                                   |
| $\mathcal{S}$                | Signature space                                                             |
| $\ \cdot\ $                  | Euclidean norm                                                              |
| $Adv$                        | Adversary's advantage                                                       |
| $Pr[\cdot]$                  | Probability of an event                                                     |
| $Adv_{CH}^{ColRes}$          | Advantage in collision resistance game for chameleon hash                   |
| $Adv_{KEM}^{IND-CCA}$        | Advantage in IND-CCA game for KEM                                           |
| $Adv_{IDKEM}^{IND-CCA}$      | Advantage in IND-CCA game for identity-based KEM                            |
| $Adv_{HKEM^{XIM}}^{IND-CCA}$ | Advantage in IND-CCA game for hybrid KEM with XOR-then-MAC combiner         |
| $Adv_{MAC}^{OT-sEUF}$        | Advantage in one-time strongly existentially unforgeable game for MAC       |
| $Adv_{SIG}^{UF-CMA}$         | Advantage in unforgeability under chosen message attack game for signatures |
| $\oplus$                     | Bitwise XOR operation                                                       |
| $\ $                         | Concatenation operation                                                     |
| $R_q$                        | Polynomial ring $\mathbb{Z}_q[x]/(f(x))$                                    |
| $\hat{a}$                    | Vector in polynomial ring $R_q^m$                                           |
| $\cdot$                      | Dot product                                                                 |
| $f_{\hat{a}}(\hat{x})$       | Preimage sampleable function                                                |
| $H(\cdot)$                   | Cryptographic hash function                                                 |
| $CH.Hash$                    | Chameleon hash function                                                     |
| $CH.Collision$               | Collision-finding algorithm for chameleon hash                              |
| $CH.Preimage$                | Preimage-finding algorithm for chameleon hash                               |
| $KEM.KeyGen$                 | Key generation algorithm for KEM                                            |

|                      |                                                 |
|----------------------|-------------------------------------------------|
| <i>KEM.Encaps</i>    | Encapsulation algorithm for KEM                 |
| <i>KEM.Decaps</i>    | Decapsulation algorithm for KEM                 |
| <i>IDKEM.Extract</i> | Key extraction algorithm for identity-based KEM |
| <i>MAC.Sign</i>      | Signing algorithm for MAC                       |
| <i>MAC.Verify</i>    | Verification algorithm for MAC                  |
| <i>SIG.Sign</i>      | Signing algorithm for signature scheme          |
| <i>SIG.Verify</i>    | Verification algorithm for signature scheme     |

## CONTENTS

|              |                                                                                     |           |
|--------------|-------------------------------------------------------------------------------------|-----------|
| <b>1</b>     | <b>INTRODUCTION</b>                                                                 | <b>24</b> |
| 1.1          | MOTIVATION                                                                          | 25        |
| 1.2          | THESIS GOALS                                                                        | 26        |
| 1.3          | OVERVIEW OF THIS THESIS CONTRIBUTIONS                                               | 26        |
| 1.4          | METHODOLOGY                                                                         | 27        |
| <b>1.4.1</b> | <b>Exploratory Work on Chameleon Hashes for Signatures with Enhanced Guarantees</b> | <b>28</b> |
| 1.5          | METHODOLOGICAL LIMITATIONS                                                          | 29        |
| <b>1.5.1</b> | <b>Experimental Constraints</b>                                                     | <b>29</b> |
| <b>1.5.2</b> | <b>Security Model Simplifications</b>                                               | <b>29</b> |
| <b>1.5.3</b> | <b>Implementation and Deployment Considerations</b>                                 | <b>30</b> |
| <b>1.5.4</b> | <b>Scope Limitations</b>                                                            | <b>30</b> |
| 1.6          | THESIS STRUCTURE                                                                    | 31        |
| <b>2</b>     | <b>PRELIMINARIES</b>                                                                | <b>32</b> |
| 2.1          | NOTATION                                                                            | 32        |
| 2.2          | CRYPTOGRAPHIC SCHEMES                                                               | 33        |
| <b>2.2.1</b> | <b>Hash Functions</b>                                                               | <b>33</b> |
| <b>2.2.2</b> | <b>Traditional Chameleon Hash Scheme</b>                                            | <b>34</b> |
| <b>2.2.3</b> | <b>Secret-Coin Chameleon-Hash</b>                                                   | <b>36</b> |
| <b>2.2.4</b> | <b>Key Encapsulation Mechanisms (KEMs)</b>                                          | <b>37</b> |
| 2.2.4.1      | KEM Attack Games                                                                    | 38        |
| 2.2.4.2      | Diffie-Hellman KEM                                                                  | 38        |
| 2.2.4.3      | Combiners and Hybrid Constructions                                                  | 39        |
| <b>2.2.5</b> | <b>Identity Based KEMs</b>                                                          | <b>40</b> |
| <b>2.2.6</b> | <b>Message Authentication Codes (MACs)</b>                                          | <b>42</b> |
| <b>2.2.7</b> | <b>Signature Schemes</b>                                                            | <b>43</b> |
| 2.2.7.1      | ECDSA Signature Scheme                                                              | 44        |
| 2.3          | PROTOCOLS                                                                           | 45        |
| <b>2.3.1</b> | <b>Bitcoin Backbone Protocol</b>                                                    | <b>45</b> |
| <b>2.3.2</b> | <b>The Transport Layer Security 1.3 Protocol</b>                                    | <b>45</b> |
| <b>2.3.3</b> | <b>The KEMTLS Protocol</b>                                                          | <b>48</b> |
| 2.4          | CONCLUSION                                                                          | 49        |
| <b>3</b>     | <b>LITERATURE REVIEW</b>                                                            | <b>50</b> |
| 3.1          | INTRODUCTION                                                                        | 50        |
| 3.2          | RESEARCH QUESTIONS                                                                  | 52        |
| 3.3          | SEARCH AND SELECTION STRATEGY                                                       | 52        |
| 3.4          | RESULTS                                                                             | 55        |

|       |                                                                          |     |
|-------|--------------------------------------------------------------------------|-----|
| 3.4.1 | <b>Scheme Definitions</b>                                                | 56  |
| 3.4.2 | <b>Security Assumptions</b>                                              | 57  |
| 3.4.3 | <b>Applications and Relevant Properties</b>                              | 59  |
| 3.5   | SUPPLEMENTARY INFORMATION                                                | 67  |
| 3.6   | THREATS TO VALIDITY OF THE SYSTEMATIC MAPPING                            | 68  |
| 3.7   | DISCUSSION OF RESULTS                                                    | 68  |
| 3.8   | CONCLUSION                                                               | 70  |
| 4     | <b>A NEW PROTOCOL FOR REDACTABLE BLOCKCHAINS</b>                         | 72  |
| 4.1   | INTRODUCTION                                                             | 72  |
| 4.2   | INTRODUCTION TO REDACTABLE BLOCKCHAIN TECHNOLOGY                         | 73  |
| 4.3   | ATENIESE'S REDACTABLE BLOCKCHAIN                                         | 74  |
| 4.4   | PREIMAGE CHAMELEON HASH                                                  | 75  |
| 4.4.1 | <b>Classical Construction Based on Factoring and Modular Square Root</b> | 76  |
| 4.4.2 | <b>Post-Quantum Construction Based on Lattices</b>                       | 78  |
| 4.5   | REDACTED BLOCKCHAIN BASED ON PREIMAGE CHAMELEON HASH FUNCTIONS           | 81  |
| 4.6   | SECURITY DISCUSSION                                                      | 83  |
| 4.7   | EXPERIMENTAL RESULTS                                                     | 85  |
| 4.8   | CONCLUSION                                                               | 87  |
| 5     | <b>EASING POST-QUANTUM MIGRATION ON TLS PROTOCOL</b>                     | 89  |
| 5.1   | INTRODUCTION                                                             | 89  |
| 5.2   | TRANSPORT LAYER SECURITY FUNDAMENTALS                                    | 90  |
| 5.3   | QUANTUM THREAT AND POST-QUANTUM INITIATIVES                              | 91  |
| 5.4   | HYBRID APPROACHES AND TRANSITION CHALLENGES                              | 91  |
| 5.5   | OTHER NOTABLE RESEARCH EFFORTS AIMING TO EASE MIGRATION                  | 93  |
| 5.6   | OUR PROPOSED SOLUTION                                                    | 93  |
| 5.7   | ADDRESSING IBE CHALLENGES IN TLS                                         | 94  |
| 5.7.1 | <b>Revocation Problem</b>                                                | 94  |
| 5.7.2 | <b>Key Escrow Problem</b>                                                | 94  |
| 5.8   | OUR HYBRID KEMTLS PROPOSAL                                               | 95  |
| 5.8.1 | <b>The Proposed Hybrid KEM Scheme (HKEM)</b>                             | 95  |
| 5.8.2 | <b>Security Modelling for the Hybrid Construction</b>                    | 97  |
| 5.8.3 | <b>The Full Proposal and General Security Discussion</b>                 | 99  |
| 5.8.4 | <b>Migrating to Post-Quantum TLS</b>                                     | 102 |
| 5.9   | EXPERIMENTS                                                              | 104 |
| 5.10  | CONCLUSION                                                               | 118 |
| 6     | <b>DISCUSSION ABOUT THE CONTRIBUTIONS AND CONCLUSION</b>                 | 120 |

|       |                                                                        |            |
|-------|------------------------------------------------------------------------|------------|
| 6.1   | INTRODUCTION . . . . .                                                 | 120        |
| 6.2   | CONTRIBUTIONS TO BLOCKCHAIN PROTOCOLS . . . . .                        | 121        |
| 6.2.1 | <b>Novel Preimage-Based Redaction Mechanism . . . . .</b>              | <b>121</b> |
| 6.2.2 | <b>Transparency Through Structural Differentiation . . . . .</b>       | <b>122</b> |
| 6.2.3 | <b>Quantum-Resistant Security Architecture . . . . .</b>               | <b>122</b> |
| 6.2.4 | <b>Practical Impact and Extensions . . . . .</b>                       | <b>123</b> |
| 6.3   | CONTRIBUTIONS TO THE TLS 1.3 PROTOCOL . . . . .                        | 123        |
| 6.3.1 | <b>Novel Hybrid Architecture . . . . .</b>                             | <b>124</b> |
| 6.3.2 | <b>Implementation and Validation . . . . .</b>                         | <b>124</b> |
| 6.3.3 | <b>Cryptographic Foundations . . . . .</b>                             | <b>125</b> |
| 6.3.4 | <b>Migration Pathway Innovations . . . . .</b>                         | <b>125</b> |
| 6.3.5 | <b>Performance and Compatibility . . . . .</b>                         | <b>126</b> |
| 6.4   | BROADER IMPLICATIONS FOR CRYPTOGRAPHY AND SECURE<br>SYSTEMS . . . . .  | 126        |
| 6.4.1 | <b>Paradigm Shift in Cryptographic Migration Strategies . . . . .</b>  | <b>126</b> |
| 6.4.2 | <b>Reconciling Immutability with Practical Governance . . . . .</b>    | <b>127</b> |
| 6.4.3 | <b>Cryptographic Primitive Repurposing . . . . .</b>                   | <b>127</b> |
| 6.4.4 | <b>Bridging Theory and Practice in Cryptographic Engineering . . .</b> | <b>128</b> |
| 6.4.5 | <b>Implications for Quantum-Era Security Architecture . . . . .</b>    | <b>128</b> |
| 6.5   | FUTURE WORK . . . . .                                                  | 129        |
| 6.5.1 | <b>Preimage Chameleon Hash Applications . . . . .</b>                  | <b>129</b> |
| 6.5.2 | <b>IBE Optimization and Integration . . . . .</b>                      | <b>129</b> |
| 6.5.3 | <b>Post-Quantum Protocol Enhancements . . . . .</b>                    | <b>130</b> |
| 6.5.4 | <b>Implementation and Deployment . . . . .</b>                         | <b>130</b> |
| 6.5.5 | <b>Security Hardening and Verification . . . . .</b>                   | <b>131</b> |
| 6.5.6 | <b>Standardization and Practical Adoption . . . . .</b>                | <b>131</b> |
|       | <b>References . . . . .</b>                                            | <b>132</b> |
|       | <b>APPENDIX A – PROOF OF THEOREM 7 . . . . .</b>                       | <b>145</b> |
|       | <b>APPENDIX B – PROOF OF THEOREM 8 . . . . .</b>                       | <b>153</b> |

## 1 INTRODUCTION

Public key cryptography forms the foundation of secure data exchange over the Internet and other computer networks. Most modern public key systems rely on the computational difficulty of factoring semi-prime numbers or solving the discrete logarithm problem in elliptic curve groups. Over the past few decades, these two mathematical problems have provided the basis for security while enabling efficient algorithms and implementations for cryptographic primitives.

The computational hardness of factoring numbers and computing discrete logarithms remained unchallenged by classical analysis and algorithms for decades. However, in the 1990s, Peter Shor introduced a groundbreaking quantum algorithm capable of factoring semiprime numbers and solving discrete logarithms on elliptic curves in polynomial time (SHOR, 1999). This algorithm relies on a probabilistic Turing machine augmented with a quantum tape, enabling operations based on quantum circuits. Shor's work also defined the complexity class BQP (Bounded-Error Quantum Polynomial), encompassing decision problems solvable by a quantum computer in polynomial time with an error probability of at most  $1/3$  for all instances.

In the following decades, advancements in nanotechnology and engineering enabled the construction of small-scale quantum computers, demonstrating the feasibility of quantum algorithms. These developments suggest that quantum computing could pose a significant threat to cryptography in the future. The potential emergence of Cryptographically Relevant Quantum Computers (CRQCs), capable of implementing Shor's algorithm or other quantum algorithms, could compromise the security of current cryptographic systems (a survey presenting perspectives from several researchers on when they expect such computers to be built can be found in (MOSCA; PIANI, 2022)).

For certain technologies, anticipating future threats and implementing protective measures well in advance is essential. If a CRQC becomes a reality within the next 20 years, an attacker could intercept and store current encrypted communications and decrypt them years later in an attack known as "harvest now, decrypt later." This threat is particularly relevant for data intended to remain secure over long periods, such as information stored on blockchains or in long-term archives.

However, migrating to post-quantum algorithms is a complex task. Cryptography based on discrete logarithms and factoring assumptions is typically faster and generates smaller keys and ciphertexts. Transitioning to post-quantum algorithms often results in longer processing times, increased data transmission, and greater storage requirements, posing significant challenges, particularly for resource-constrained devices. Additionally, in many widely used protocols, adopting post-quantum algorithms necessitates substantial infrastructure changes, including modifications to certificate chains and certificate authorities.

## 1.1 MOTIVATION

This work seeks to address the key question of how to simplify the adoption and transition to post-quantum algorithms in cryptographic applications, particularly for authentication and integrity. Our goal is to offer new methods to mitigate the challenges of post-quantum migration by exploring less common cryptographic primitives, such as chameleon hash functions, and leveraging identity-based encryption to eliminate the need for distributing or certifying post-quantum keys.

Among the motivating scenarios for this research is the desire to adapt redactable blockchains to the post-quantum setting, while preserving their flexibility and ensuring long-term security. Redactable blockchains depend critically on chameleon hash functions to allow controlled modifications of historical data without compromising integrity. This creates a significant challenge in the post-quantum context, as there is currently a lack of chameleon hash constructions that are both quantum-resistant and capable of meeting the stringent security requirements of such systems. Existing solutions are either vulnerable to quantum attacks or fail to provide the necessary collision resistance and trapdoor properties. This gap motivated the search for new cryptographic constructions tailored to enable redactability in blockchain systems under post-quantum assumptions.

This work was also motivated by findings that the greatest overhead in migrating protocols like SSH and TLS to post-quantum versions lies in their authentication components, such as key exchanges and signatures (SIKERIDIS et al., 2020). While exploring alternative cryptographic primitives, such as chameleon hash functions, we identified practical constructions with additional properties that could be leveraged to develop new cryptographic techniques and results (ASTRIZI, T. et al., 2020). Therefore, we wanted to explore if these properties could help in more complex protocols and scenarios.

New benchmarks for post-quantum cryptographic algorithms have highlighted the need to reevaluate which primitives are viable for use in known protocols. For instance, identity-based cryptography (IBE) was traditionally considered less attractive than public-key algorithms due to its higher computational demands and only marginal savings in network bandwidth. However, in the context of post-quantum algorithms, IBE's computational overhead is comparatively modest, while its reduction in data transmission is significantly greater, as demonstrated in (BANERJEE; CHANDRAKASAN, 2020). Although that study focused on early algorithms and was limited to Internet-of-Things (IoT) scenarios, its findings were promising and inspired further exploration of IBE solutions in this work.

With the potential emergence of Cryptographically Relevant Quantum Computers, there is growing interest in migrating existing protocols to post-quantum versions. However, this transition faces significant challenges, including the inefficiency of current

post-quantum algorithms and the complexities of large-scale migration. This thesis aims to address these issues by proposing alternative solutions to simplify and enhance the migration process.

## 1.2 THESIS GOALS

To refine our proposals and ground our analysis in concrete examples, we selected two protocols, the Bitcoin Backbone Protocol (defined in (GARAY et al., 2015), and used as a formal framework to analyze general blockchain operations, not limited to cryptocurrency applications) and TLS 1.3, to evaluate the challenges posed by post-quantum algorithms and their integration into widely used communication protocols. While these protocols serve as case studies, our objective is to develop solutions that can be generalized to other protocols and domains, ensuring broader applicability.

Therefore, the goals of this thesis are:

- Identify challenges in adopting post-quantum algorithms for the chosen protocols through a literature review;
- Mitigate the identified problems with post-quantum algorithms, using the simplest effective mechanisms whenever possible;
- Provide theoretical contributions with rigorous security reductions for our proposals, ensuring that any deviations from the original protocols do not introduce new security weaknesses;
- Discuss the applicability of our proposals and solutions to other protocols or problems where relevant.

These goals align with the overall objective of offering more general solutions while focusing on specific protocols for detailed and concrete analysis.

## 1.3 OVERVIEW OF THIS THESIS CONTRIBUTIONS

The main contributions of this thesis are:

- As part of our strategy to address post-quantum challenges, we conducted a systematic mapping to explore different types of chameleon hash functions, their security assumptions, and their applications in various problems. A Systematic Mapping is a structured approach to surveying and categorizing the existing literature with the aim of generating new insights into a specific topic, in accordance with a rigorous research methodology;

- We proposed a new construction for redactable blockchains using chameleon hash functions. This construction allows trusted parties to redact blocks in specific scenarios, such as the removal of illegal content from the blockchain. Additionally, we identified instantiations where the hash time was faster than previously reported compatible constructions;
- We evaluated the performance of various redactable blockchain implementations based on chameleon hash functions, providing detailed experimental results;
- We introduced a novel hybrid protocol based on KEMTLS, integrating traditional public-key cryptography with post-quantum identity-based encryption. This approach improves security by ensuring resilience against both classical and quantum adversaries;
- We tested our proposed post-quantum TLS protocol alongside other hybrid algorithms currently being standardized by NIST. These tests were conducted in realistic scenarios using rented computers from different companies across continents to compare performance between geographically distant machines;
- We performed a comprehensive security analysis of our proposals. For the TLS protocol, we developed a novel hybrid algorithm connecting two cryptographic primitives that, while similar, required distinct formalization. This hybrid construction necessitated the definition of realistic security requirements, capturing the unique properties of both primitives.

## 1.4 METHODOLOGY

The methodology of this work is divided into three parts. As an initial contribution, Chapter 3 presents a systematic mapping to study the definitions, properties, security assumptions, and use cases of chameleon hash functions in the literature. This methodology aims to research, classify, and provide new insights into the existing body of work on this topic. It served as the starting point of our research, building on the author's earlier work during their master's degree. Specifically, we focused on chameleon hash constructions with non-standard properties, such as the ability to compute first preimages when the trapdoor is known, rather than just regular collisions, and particularly on post-quantum constructions of chameleon hash functions.

In the second part of this thesis, presented in Chapter 4, we analyze the Bitcoin protocol. Rather than conducting a systematic mapping on this topic, we built upon existing surveys, such as (ZHANG, D. et al., 2021), which highlighted challenges with the immutability of blockchains. While foundational to blockchain technology, this property is not suitable for all applications and can expose the system to attacks where malicious users upload illegal content to the blockchain network.

Drawing insights from the survey and the Bitcoin protocol analysis, we proposed a new construction for blockchains using chameleon hash functions for alternative linking. This approach enables redactable blockchains that are simpler and potentially faster than existing designs. To validate our proposal, we implemented the construction and conducted experiments comparing the performance of the hash, verification, and redact algorithms against similar literature-based proposals using chameleon hash functions. The results, presented in tables, include the mean and margin of error for each case, assuming a Gaussian distribution.

In the third and final part of this thesis, we analyzed the TLS 1.3 protocol. Building on works such as (SIKERIDIS et al., 2020) and (PAQUIN et al., 2020), we focused on addressing the challenges of transitioning from TLS 1.3 to a post-quantum version. These challenges include the significant performance impact of larger key sizes and TLS messages required by post-quantum algorithms, as well as the complexity of migrating an ecosystem that includes clients, servers, certificate authorities, certificate validation systems, revocation checks, and trust in root certificate authorities.

To ease this transition, we developed a novel approach that avoids introducing post-quantum keys into the existing infrastructure. Instead, we leveraged identity-based encryption to integrate post-quantum security into the protocol without modifying current certificates, certification protocols, or revocation mechanisms. To evaluate the feasibility and performance of our solution, we measured protocol timings in both a simulated environment and real-world tests between machines located on different continents. We also compared these results with other existing hybrid solutions.

#### **1.4.1 Exploratory Work on Chameleon Hashes for Signatures with Enhanced Guarantees**

At the early stages of this research, one of the initial objectives was to explore the application of chameleon hash functions to the construction of digital signature schemes with novel security and functionality properties. Specifically, we aimed to extend traditional digital signatures by incorporating mechanisms that would offer additional guarantees beyond standard verification and non-repudiation, such as more granular accountability or context-aware validation.

However, despite extensive theoretical investigation and prototype development, the envisioned cryptographic properties could not be achieved as intended. After successfully modeling the security requirements for the proposed signature scheme, it became evident that constructing a scheme satisfying these properties was significantly more challenging than anticipated. Although we managed to design constructions that met weaker variants of our requirements, these results proved less compelling, as similar guarantees could be achieved through simpler approaches that did not rely on chameleon hash functions. Consequently, this line of inquiry did not yield contributions

substantial enough to be included among the core results of this thesis.

Interestingly, as a byproduct of this research trajectory, we discovered an alternative method of embedding custom metadata within digital signatures using a different set of cryptographic tools. While this approach does not realize the original goals of enhanced verification guarantees, it enables a flexible and potentially valuable mechanism for associating auxiliary data with signed messages. This auxiliary technique is currently undergoing the patenting process. However, since it does not constitute a core contribution of this dissertation and does not integrate with the main technical framework presented herein, we have opted to omit its detailed discussion from the manuscript.

## 1.5 METHODOLOGICAL LIMITATIONS

While this thesis strives for comprehensive analysis and rigorous evaluation, several methodological limitations should be acknowledged to contextualize its contributions and findings:

### 1.5.1 Experimental Constraints

The experimental evaluations presented in this work, while designed to reflect realistic deployment scenarios, are subject to certain limitations:

- **Network Simulation Fidelity:** Our TLS protocol evaluations rely on network emulation to simulate various latency conditions. Although NetEm provides a reasonable approximation of network characteristics, it cannot fully capture the complex, dynamic behaviors of the global Internet, including traffic bursts, congestion patterns, and routing anomalies that might affect protocol performance in unpredictable ways;
- **Implementation Maturity:** The prototype implementations developed to validate our approaches prioritize functional correctness over optimization. Production-ready implementations would likely achieve better performance through code optimization, hardware acceleration, and implementation-specific enhancements;
- **Scale Limitations:** Our blockchain evaluations focused on isolated redaction operations rather than continuous operation in large-scale networks with high transaction volumes. The cumulative effects of multiple redactions on overall system performance may warrant further investigation.

### 1.5.2 Security Model Simplifications

Our security analyses necessarily employ certain simplifications and assumptions:

- **Idealized Cryptographic Primitives:** Some security proofs rely on idealized models such as random oracles or generic group models, which provide strong theoretical guarantees but may not perfectly reflect the behavior of implemented constructions;
- **Bounded Adversarial Capabilities:** Our security models assume computationally bounded adversaries. While this is standard practice in cryptographic research, it is important to acknowledge that the exact bounds of quantum computational capabilities remain uncertain as the technology evolves;
- **Isolated Protocol Analysis:** The security analyses typically consider protocols in isolation rather than in composition with other system components. In practice, subtle interactions between protocols and system elements can sometimes introduce vulnerabilities not captured in isolated analyses.

### 1.5.3 Implementation and Deployment Considerations

The transition from theoretical constructions to practical deployment introduces additional complexities:

- **Side-Channel Vulnerabilities:** While our constructions are provably secure within their formal models, implementations may be vulnerable to side-channel attacks. The post-quantum primitives employed may have different side-channel characteristics compared to classical alternatives, potentially requiring new mitigation strategies;
- **Deployment Economics:** Our work focuses primarily on technical feasibility rather than detailed economic analysis of deployment costs. A comprehensive transition strategy would need to account for implementation, integration, testing, and maintenance costs across diverse infrastructure;
- **Organizational Adoption Barriers:** The institutional and organizational challenges of cryptographic transitions extend beyond technical considerations to include governance, risk management, compliance, and human factors that may influence adoption timelines and strategies.

### 1.5.4 Scope Limitations

While comprehensive within its defined scope, this research necessarily excludes certain dimensions:

- **Limited Protocol Coverage:** Our investigation focuses on two significant but specific protocols (blockchain and TLS). The applicability of our approaches to other cryptographic protocols may require additional validation and adaptation;

- **Evolving Quantum Threat Models:** The quantum threat landscape continues to evolve as quantum computing research progresses. This work necessarily relies on the current state of knowledge regarding the capabilities of quantum computing capabilities, which are expected to evolve as new breakthroughs emerge.
- **Post-Standardization Developments:** Our research precedes the final standardization of many post-quantum algorithms. Future standards may introduce refinements or alternatives that could influence optimal implementation strategies.

These limitations do not diminish the contributions of this thesis, but rather delineate boundaries of the current work and highlight opportunities for future research. Throughout the thesis, we attempt to acknowledge these limitations where relevant and discuss their implications for the interpretation and application of our results.

## 1.6 THESIS STRUCTURE

This thesis explores methodologies for enhancing cryptographic protocols in post-quantum environments through two main research directions:

**Chapter 2: Preliminaries** - Presents fundamental concepts, notation, and cryptographic schemes used throughout the thesis.

**Chapter 3: Literature Review** - Presents a systematic mapping of chameleon hash functions, analyzing their definitions, properties, security assumptions, and applications.

**Chapter 4: A New Protocol for Redactable Blockchains** - Proposes a novel construction for post-quantum redactable blockchains utilizing specialized chaining mechanisms based on chameleon hash functions with preimage properties, enabling controlled mutability while maintaining cryptographic integrity.

**Chapter 5: Easing Post-Quantum Migration on TLS Protocol** - Addresses challenges of transitioning TLS 1.3 to post-quantum security by introducing a hybrid mechanism that integrates identity-based encryption with classical cryptography, preserving compatibility with current certificate chains and authentication mechanisms.

**Chapter 6: Discussion and Conclusion** - Summarizes the contributions, discusses their implications, and outlines potential directions for future research.

The thesis introduces practical implementations of both proposals, conducts rigorous experimental evaluations across various network environments, and provides comprehensive security analyses. The methodologies presented have been implemented as open-source solutions to encourage further research and practical adoption in the field of post-quantum cryptography.

## 2 PRELIMINARIES

This chapter establishes the fundamental notation, definitions, and concepts that serve as the foundation for the remainder of this thesis. We begin by introducing the notational conventions adopted throughout our work, ensuring clarity and consistency in the presentation of algorithms, cryptographic schemes, and security proofs.

Subsequently, we provide formal definitions of the core cryptographic primitives relevant to our research. These include chameleon hash functions, key encapsulation mechanisms (KEMs), identity-based encryption schemes, message authentication codes, and digital signature schemes. For each primitive, we present the algorithms that constitute the scheme, along with the security models that formalize their expected properties.

Finally, we describe the protocols that form the basis of our case studies: the Bitcoin backbone protocol, which underlies the most widely used blockchain implementation, and the Transport Layer Security (TLS) protocol version 1.3, which secures a significant portion of internet communications. We also introduce the KEMTLS protocol, a post-quantum alternative to TLS that replaces digital signatures with key encapsulation mechanisms.

The definitions and concepts presented in this chapter provide the necessary technical background to understand the security analyses and protocol modifications proposed in subsequent chapters. Readers familiar with these preliminaries may choose to skip directly to the contributions beginning in Chapter 3.

### 2.1 NOTATION

We use uppercase letters to represent cryptographic schemes (e.g., *KEM* for key encapsulation mechanisms and *SIG* for signature schemes). Each algorithm in these schemes is prefixed by the scheme name for easy identification. For example, *KEM.KeyGen* and *SIG.KeyGen* refer to different key generation algorithms, even though they share the same name after the prefix. Algorithms not associated with a specific cryptographic scheme are denoted without a prefix, with the first letter capitalized.

Sets are represented by single uppercase letters, such as *A* and *M*, while generic set elements are denoted by lowercase names like *msg* and *k*. The symbol  $\lambda$  is used to denote security parameters in cryptographic schemes.

When describing algorithms or attack games, the symbol “ $\leftarrow$ ” denotes an assignment. We also use “ $\xleftarrow{\$}$ ” to indicate that an element is sampled uniformly at random from the set on the right side of the operator and assigned to the variable on the left. If a returned value is irrelevant and should be discarded, it is assigned to the variable “ $-$ ”.

In attack games, we use  $\mathcal{A}$  to represent an adversary attempting to break the security of a cryptographic scheme. The adversary is modeled as a tuple of algorithms,

depending on the rules of the specific attack game. Subscripts, such as  $\mathcal{A}_1$  and  $\mathcal{A}_2$ , are used to distinguish between the different algorithms that make up  $\mathcal{A}$  and to differentiate stages of the attack game when  $\mathcal{A}$  is invoked. If  $\mathcal{A}$  is able to query oracles during the game, the oracles are represented as superscripts over  $\mathcal{A}$ , for example,  $\mathcal{A}^{f(\cdot)}(x)$  denotes an adversary which gets  $x$  as input and is able to query an oracle about the output of some function or algorithm  $f$ . For more information about how attack games are used in cryptographic proofs, check (SHOUP, 2004).

The notation  $Pr[A]$  denotes the probability of a particular event  $A$  occurring. When discussing events related to some adversary breaking the security of a primitive, we may instead use the notation  $Adv(\mathcal{A})$ . This denotes the probability of a given adversary  $\mathcal{A}$  breaking the security of a specific cryptographic primitive, with subscripts indicating which primitive is being broken and superscripts indicating the security definition. We say that a given probability is **negligible** if, when modeled as a function  $f : \mathbb{Z}_{\geq 1} \rightarrow \mathbb{R}$  that takes a security parameter as input, it satisfies  $\lim_{n \rightarrow \infty} f(n) \cdot n^c = 0$  for all  $c > 0$ .

## 2.2 CRYPTOGRAPHIC SCHEMES

### 2.2.1 Hash Functions

A hash function  $H : M \rightarrow D$  is an efficiently computable function that maps a (typically large) domain  $M$  into a much smaller digest space  $D$ .

We define a family of hash functions as a set of efficiently computable functions parameterized by a security parameter  $1^\lambda$ . The security of such a family is modeled according to the Collision Resistance Attack Game below. We say that a family of hash functions is collision-resistant if, for all efficient adversaries  $\mathcal{A}$ , the probability that the game outputs 1 is negligible.

---

#### Collision Resistance Game (Hash Functions)

---

```

1:  $(msg_1, msg_2) \xleftarrow{\$} \mathcal{A}(1^\lambda)$ 
2: if  $msg_1 \neq msg_2$  and  $H(1^\lambda, msg_1) = H(1^\lambda, msg_2)$  then
3:   return 1
4: else
5:   return 0
6: end if

```

---

There are weaker security notions for hash functions. We say that a family of hash functions is second-preimage resistant if, for all efficient adversaries  $\mathcal{A}$ , the probability that the Second-Preimage Resistance Attack Game outputs 1 is negligible.

Finally, we say that a family of hash functions parametrized by the security parameter  $1^\lambda$  is (first) preimage resistant if, for all efficient adversaries  $\mathcal{A}$ , the probability that the First-Preimage Resistance Attack Game outputs 1 is negligible.

---

**Second-Preimage Resistance (Hash Functions)**


---

```

1:  $msg_1 \xleftarrow{\$} M$ 
2:  $msg_2 \xleftarrow{\$} \mathcal{A}(1^\lambda, msg_1)$ 
3: if  $msg_1 \neq msg_2$  and  $H(1^\lambda, msg_1) = H(1^\lambda, msg_2)$  then
4:   return 1
5: else
6:   return 0
7: end if

```

---



---

**First-Preimage Resistance (Hash Functions)**


---

```

1:  $msg_1 \xleftarrow{\$} M$ 
2:  $dgt \leftarrow H(1^\lambda, msg_1)$ 
3:  $msg_2 \xleftarrow{\$} \mathcal{A}(1^\lambda, dgt)$ 
4: if  $H(1^\lambda, msg_2) = dgt$  then
5:   return 1
6: else
7:   return 0
8: end if

```

---

Notice that an adversary capable of finding preimages for a given hash function could also break its second-preimage resistance, provided that the digest space is significantly smaller than the message space, which is typically the case for most practical hash functions. Moreover, breaking the second-preimage resistance would directly enable the adversary to find collisions.

From this point onward, we omit the security parameter  $1^\lambda$  when discussing hash functions, assuming by default that a hash function with a sufficiently large security parameter is used to ensure security in our applications. Therefore, we should write  $H(msg)$  instead of  $H(1^\lambda, msg)$ .

### 2.2.2 Traditional Chameleon Hash Scheme

A traditional chameleon hash scheme CH is a tuple of three efficient algorithms ( $CH.KeyGen$ ,  $CH.Hash$ ,  $CH.Collision$ ) defined over three sets:  $M$  (the message space),  $R$  (the randomness space) and  $D$  (digest space). We define the three algorithms as:

- $CH.KeyGen(1^\lambda) \rightarrow (ek, tk)$ : Given an input a security parameter  $1^\lambda$  and outputs a pair of keys: an evaluation key ( $ek$ ) and trapdoor key ( $tk$ );
- $CH.Hash(ek, msg, rnd) \rightarrow dgt$ : Given an evaluation key  $ek$ , some message  $msg \in M$  and a randomness  $rnd \in R$ , this algorithm returns a digest  $dgt \in D$ ;
- $CH.Collision(tk, msg, rnd, msg') \rightarrow rnd'$ : Given a trapdoor key  $tk$ , a message  $msg \in M$ , a randomness  $rnd \in R$  and a second message  $msg' \in M$ , this algorithm returns a  $rnd' \in R$  such that  $CH.Hash(ek, msg, rnd) = CH.Hash(ek, msg', rnd')$ .

The primitive was originally defined in (KRAWCZYK; RABIN, 1998).

Note that  $\text{CH.Collision}$  computes an arbitrary second-preimage. There is no consensus on how to name this algorithm in the literature. Some authors use the name “*Forge*” (ATENIESE; MEDEIROS, 2004a) because it allows for creating forgeries in signatures using the primitive. Others call it “*Coll*” (BLAZY et al., 2015) as the abbreviation of “collision” and some others use  $H^{-1}$  (MOHASSEL, 2010) meaning that it is the inverse of the hash function  $H$ . Here the name “*Coll*” was preferred, but it is used in the non-abbreviated form.

A secure chameleon hash should be collision-resistant, which means that given an evaluation key  $ek$ , no adversary should be able to find different inputs for  $\text{CH.Hash}(ek, \cdot, \cdot)$  that produces the same output. More formally, the collision-resistance means that for all adversaries  $\mathcal{A}$ , the probability of the Chameleon Hash Collision Resistance Attack Game return 1 is bounded by a negligible value. This probability, given some adversary  $\mathcal{A}$  is denoted by  $\text{Adv}_{\text{CH}}^{\text{ColRes}}(\mathcal{A})$ .

---

#### Chameleon Hash Collision Resistance Attack Game

---

```

1:  $(ek, tk) \xleftarrow{\$} \text{CH.KeyGen}(1^\lambda)$ 
2:  $(msg_1, rnd_1, msg_2, rnd_2) \xleftarrow{\$} \mathcal{A}(1^\lambda, ek)$ 
3: if  $(msg_1, rnd_1) \neq (msg_2, rnd_2)$  and  $\text{CH.Hash}(ek, msg_1, rnd_1) =$   

    $\text{CH.Hash}(ek, msg_2, rnd_2)$  then
4:   return 1
5: else
6:   return 0
7: end if
```

---

The attack game models the security in a scenario where attackers is not given access to sample collisions. The only information given to it is the security parameter and the evaluation key  $ek$ . This security definition is insufficient if we want to use chameleon hash functions in scenarios where sample collisions may be leaked. However, still, we want that no adversary could find new collisions other than the leaked ones.

A stronger version of collision-resistance is the **full collision-resistance**. It requires that an adversary cannot find new collisions in the chameleon hash even if it has access to an oracle that shows him the result of  $\text{CH.Collision}(tk, \cdot, \cdot, \cdot)$ . More formally, this property requires that for all adversaries  $\mathcal{A}$ , the probability of the game below return 1 is bounded by a negligible value. The game is similar to the previous one, but in this new version the adversary is allowed to make queries for the algorithm *Collision* (described below), which produces sample collisions and updates a list of queried inputs. In the end, the adversary must produce a collision using at least one pair not sent or produced by collision queries. This ensures adaptive security for the chameleon hash, protecting it against adaptive attacks where the adversary tailors its queries based on

prior responses and leaked information.

---

**Chameleon Hash Full Collision Resistance Attack Game**

---

```

1:  $L \leftarrow \emptyset$  ▷  $L$  will store a list of queried chameleon hash inputs.
2:  $(ek, tk) \xleftarrow{\$} \text{CH.KeyGen}(1^\lambda)$ 
3:  $(msg_1, rnd_1, msg_2, rnd_2) \xleftarrow{\$} \mathcal{A}^{\text{Collision}(L, tk, \cdot, \cdot, \cdot)}(1^\lambda, ek)$ 
4: if  $(msg_1, rnd_1) \neq (msg_2, rnd_2)$  and  $\text{CH.Hash}(ek, msg_1, rnd_1) =$   

    $\text{CH.Hash}(ek, msg_2, rnd_2)$  and  $(msg_1, rnd_1) \notin L$  then
5:   return 1
6: else
7:   return 0
8: end if

```

---



---

**Algorithm 6** Collision Query for Chameleon Hash Full Collision Resistance Attack Game

---

```

1:  $\text{Collision}(L, tk, msg_1, rnd_1, msg_2)$ :
2:    $rnd_2 \xleftarrow{\$} \text{CH.Collision}(tk, msg_1, rnd_1, msg_2)$ 
3:    $L \leftarrow L \cup \{(msg_1, rnd_1), (msg_2, rnd_2)\}$  ▷ Stores queried inputs in  $L$ .
4:   return  $rnd_2$ 

```

---

### 2.2.3 Secret-Coin Chameleon-Hash

A secret-coin chameleon hash scheme SCH is a tuple of efficient algorithms (SCH.KeyGen, SCH.Hash, SCH.Verify, SCH.Collision) defined over two sets:  $M$  (the message space) and  $D$  (digest space). We define the four algorithms as:

- SCH.KeyGen( $1^\lambda$ )  $\rightarrow (ek, tk)$ : On input a security parameter from  $\mathbb{N}$ , this algorithm outputs an evaluation key  $ek$  and a trapdoor key  $tk$ ;
- SCH.Hash( $ek, msg$ )  $\rightarrow (dgt, str)$ : Given an evaluation key  $ek$  and a message  $msg \in M$ , this algorithm outputs a digest  $dgt \in D$  and a verification string  $str$ ;
- SCH.Verify( $ek, msg, dgt, str$ )  $\rightarrow b$ : Given an evaluation key  $ek$ , a message  $msg \in M$ , a digest  $dgt \in D$  and a verification string  $str$ , this algorithm outputs the  $b = 1$  if  $dgt$  is a legitimate digest for  $msg$  or outputs  $b = 0$  otherwise;
- SCH.Collision( $tk, msg_1, dgt, str_1, msg_2$ )  $\rightarrow str_2$ : Given a trapdoor key  $tk$ , messages  $msg_1, msg_2 \in M$ , a digest  $dgt \in D$  and a verification string  $str_1$  this algorithm outputs a second verification string  $str_2$  such that

$$\text{SCH.Verify}(ek, msg_1, dgt, str_1) = 1 \implies \text{SCH.Verify}(ek, msg_2, dgt, str_2) = 1.$$

The term “secret-coin chameleon hash” was introduced in (ATENIESE et al., 2017), although earlier constructions can be found in (CHEN, X. et al., 2004).

The main difference in this scheme, compared to a traditional chameleon hash, is that the user does not control the randomness of the hash function. Instead, the source of randomness is external and independent. This prevents users from verifying the hash simply by recomputing it with the same inputs. Consequently, the scheme requires a dedicated *Verify* algorithm. The motivation for using this alternative scheme is to offer properties similar to those of a traditional chameleon hash, but in a form that is easier to construct in practice while achieving full-collision resistance.

The full collision-resistance for secret-coin chameleon hash also assumes that no adversary is able to find new collisions, except with negligible probability, even if given access to an oracle that provides sample collisions. More formally, the full collision-resistance for secret-coin chameleon hash requires that for all adversaries, the probability of the following attack game return 1 is bounded by a negligible value.

---

**Secret-Coin Chameleon Hash Collision Resistance Attack Game**

---

```

1:  $L \leftarrow \emptyset$ 
2:  $(ek, tk) \xleftarrow{\$} SCH.KeyGen(1^\lambda)$ 
3:  $(msg_1, str_1, msg_2, str_2, dgt) \xleftarrow{\$} \mathcal{A}^{SCollision(L, tk, \cdot, \cdot, \cdot)}(1^\lambda, ek)$ 
4: if  $msg_1 \neq msg_2$  and  $SCH.Verify(ek, msg_1, dgt, str_1) =$   

    $SCH.Verify(ek, msg_2, dgt, str_2) = 1$  and  $(msg_1, dgt) \notin L$  then
5:   return 1
6: else
7:   return 0
8: end if

```

---



---

**Algorithm 8** Collision Query for Secret Coin Chameleon Hash Collision Resistance Attack Game

---

```

1:  $Collision(L, tk, msg_1, dtg, str_1, msg_2)$ :
2:    $L \leftarrow L \cup \{(msg_1, dgt), (msg_2, dgt)\}$ 
3:   return  $SCH.Collision(tk, msg_1, dgt, str_1, msg_2)$ 

```

---

## 2.2.4 Key Encapsulation Mechanisms (KEMs)

A Key Encapsulation Mechanism (*KEM*) is defined by the following algorithms:

- **Key Generation:**  $KEM.KeyGen(1^\lambda) \rightarrow (pk, sk)$ : Given the security parameter  $1^\lambda$ , this probabilistic polynomially-bounded algorithm generates a secret key  $sk$  and a public key  $pk$ .
- **Encapsulation:**  $KEM.Encaps(pk) \rightarrow (k, c)$ : Given a public key  $pk$ , this probabilistic polynomially-bounded algorithm generates a secret  $k \in K$  and a ciphertext  $c \in C$ , where  $K$  is the key space and  $C$  the ciphertext space.

- **Decapsulation:**  $KEM.Decaps(sk, c) \rightarrow k$ : Given a secret key  $sk$  and a ciphertext  $c \in C$ , this deterministic polynomially-bounded algorithm recovers the corresponding secret  $k \in K$ .

This primitive was first introduced in (SHOUP, 2001).

The security of a public-key  $KEM$  is evaluated through two attack games, referred to as Game 0 and Game 1, detailed below. For any adversary  $\mathcal{A}$ , let  $W_0$  represent the event where  $\mathcal{A}$  returns 1 in Attack Game 0, and  $W_1$  represent the event where  $\mathcal{A}$  returns 1 in Attack Game 1. The advantage of adversary  $\mathcal{A}$  is defined as  $Adv_{KEM}^{IND-CCA}(\mathcal{A}) = |Pr[W_0] - Pr[W_1]|$ . A  $KEM$  is considered secure if this advantage is negligible for all adversaries  $\mathcal{A}$ .

---

#### KEM Attack Game 0

---

- 1:  $(sk, pk) \leftarrow KEM.KeyGen(1^\lambda)$
  - 2:  $(c^*, k_0) \leftarrow KEM.Encaps(pk)$
  - 3:  $b \leftarrow \mathcal{A}^{KEM.Decaps(sk, \cdot)}(pk, c^*, k_0) \triangleright$  Run adversary  $\mathcal{A}$ , allowing decapsulation queries.
  - 4:  $\triangleright$  The adversary controls the ciphertext to be decapsulated.
  - 5:  $\triangleright$  The secret key  $sk$  remains hidden and cannot be accessed.
  - 6: **return**  $b$
- 

---

#### KEM Attack Game 1

---

- 1:  $(sk, pk) \leftarrow KEM.KeyGen(1^\lambda)$
  - 2:  $(c^*, -) \leftarrow KEM.Encaps(pk)$
  - 3:  $k_1 \xleftarrow{\$} K \quad \triangleright$  Randomly chosen key from key space  $K$
  - 4:  $b \leftarrow \mathcal{A}^{KEM.Decaps(sk, \cdot)}(pk, c^*, k_1)$
  - 5: **return**  $b$
- 

#### 2.2.4.1 KEM Attack Games

The adversary's objective is to distinguish between two scenarios, and the advantage measures its ability to do so, thereby compromising the  $KEM$ 's security based on the indistinguishability under chosen-ciphertext attacks (IND-CCA).

In both games, the adversary is allowed to query an oracle for decapsulation using the secret key  $sk$ , but cannot request the decapsulation of the specific ciphertext  $c^*$ .

#### 2.2.4.2 Diffie-Hellman KEM

A common Key Encapsulation Mechanism ( $KEM$ ) is the Diffie-Hellman  $KEM$  (DHKEM), which operates over a multiplicative group  $G$  with generator  $g$  and order  $q$ , typically derived from an elliptic curve. The security of DHKEM relies on the computational difficulty of the Computational Diffie-Hellman problem: given  $g$ ,  $g^a$ , and  $g^b$ ,

compute  $g^{ab}$ . This *KEM* uses a hash function  $H : \{0, 1\}^* \rightarrow K$ , where  $K$  is the secret space, and assumes efficient algorithms exist to verify solutions for the Diffie-Hellman problem.

The secret key is a random exponent  $x$  such that  $0 < x < q$ , and the corresponding public key is  $y = g^x$ . The encapsulation and decapsulation algorithms are provided in Algorithms 11 and 12.

This *KEM* is described in further detail and has a proposed standardization in Section 3.1 of RFC 9180 (BARNES et al., 2022).

---

**Algorithm 11** DHKEM.Encaps( $y$ )

---

```

1: Input: Public key  $y$ 
2: Output: Ciphertext  $c$ , Shared secret  $k$ 
3:  $x' \xleftarrow{\$} \mathbb{Z}_q$ 
4:  $y' \leftarrow g^{x'}$ 
5:  $s \leftarrow y^{x'}$ 
6:  $k \leftarrow H(s \parallel y' \parallel y)$ 
7:  $c \leftarrow y'$ 
8: return  $(c, k)$ 

```

---



---

**Algorithm 12** DHKEM.Decaps( $x, c$ )

---

```

1: Input: Secret key  $x$ , Ciphertext  $c$ 
2: Output: Shared secret  $k$ 
3:  $y' \leftarrow c$ 
4:  $s \leftarrow y'^x$ 
5: return  $H(s \parallel y' \parallel y)$ 

```

---

The security of the scheme derives from the fact that, given the public information  $(y, y')$ , only the sender, who computes  $y'$  from a secret  $x'$ , and the recipient, who knows the secret key  $x$ , are able to compute  $s = g^{x \cdot x'} = y^{x'} = y'^x$ .

### 2.2.4.3 Combiners and Hybrid Constructions

KEM combiners, introduced in (GIACON et al., 2018), provide a method for constructing a new KEM by integrating two or more existing KEMs. The primary security goal of KEM combiners is to ensure that the resulting combined KEM remains secure even if one of the original KEMs is compromised. This robustness is particularly valuable in developing hybrid schemes that combine classical and post-quantum algorithms, thereby enhancing resilience against future vulnerabilities.

Several approaches to KEM combiners have been explored in the literature. Below, we mention two that satisfy the IND-CCA security requirements for KEMs:

- **XOR-then-MAC Combiner:** This approach XORs the outputs of the combined KEMs to generate an initial shared secret. The shared secret is then authenticated

using a MAC (they will be defined in Section 2.2.6) to protect against mix-and-match attacks, where an adversary could combine parts of different ciphertexts to forge valid combinations, thus compromising more naive combiners;

- **Dual-PRF Combiner:** In this method, both KEMs are executed to produce a pair of secrets  $(k_1, k_2)$  and a pair of ciphertexts  $(c_1, c_2)$ . The secrets are then combined using a pseudo-random function (PRF) to generate a key  $k$ , which is used to instantiate another PRF. The final secret is produced by running  $PRF(k, (c_1, c_2))$ .

Both combiners are further analyzed, and their security is proven in (BINDEL et al., 2019). A comprehensive analysis of KEM combiners and their security properties is provided by Giacon et al. (GIACON et al., 2018), which formalizes the security definitions and constructions for these combiners. Their study emphasizes the importance of properly defining and constructing combiners to ensure the desired security guarantees.

In our proposal, we specifically evaluate the performance and security of the XOR-then-MAC combiner. The reason for this choice is that MACs are already required to ensure the security of the TLS handshake. They mitigate attacks such as the renegotiation attack, reported in (RAY; DISPENSA, 2009), and truncation attacks discussed in (SMYTH; PIRONTI, 2013). Therefore, the MAC required by this combiner is already integrated into standard TLS handshakes, eliminating the need for additional cryptographic primitives.

### 2.2.5 Identity Based KEMs

An identity-based Key Encapsulation Mechanism (*IDKEM*) consists of the following algorithms:

- **Key Generation:**  $IDKEM.KeyGen(1^\lambda) \rightarrow (mpk, msk)$ : Given the security parameter  $1^\lambda$ , this probabilistic polynomially-bounded algorithm generates a master public key  $mpk$  and a master secret key  $msk$ ;
- **Key Extraction:**  $IDKEM.Extract(msk, id) \rightarrow sk_{id}$ : Given the master secret key  $msk$  and a string  $id$ , this probabilistic polynomially-bounded algorithm generates a secret key  $sk_{id}$  associated with identity  $id$ ;
- **Encapsulation:**  $IDKEM.Encaps(mpk, id) \rightarrow (k, c)$ : Given the master public key  $mpk$  and an identity string  $id$ , this probabilistic polynomially-bounded algorithm produces a secret  $k$ , and a ciphertext  $c$ ;
- **Decapsulation:**  $IDKEM.Decaps(sk_{id}, c) \rightarrow k$ : Given a secret key  $sk_{id}$  associated with identity  $id$  and a ciphertext  $c$ , this deterministic polynomially-bounded algorithm recovers the secret  $k$ .

The security of an *IDKEM* is evaluated through attack games similar to those used for traditional KEMs. These games are depicted in *IDKEM* Attack Game 0 and 1, and in Algorithm 15.

---

**IDKEM Attack Game 0**


---

- 1:  $(msk, mpk) \leftarrow \text{IDKEM.KeyGen}(1^\lambda)$
  - 2:  $(id^*, st) \leftarrow \mathcal{A}_1^{\text{Decaps}(msk, \cdot), \text{IDKEM.Extract}(msk, \cdot)}(mpk)$
  - 3:  $(c^*, k_0) \leftarrow \text{IDKEM.Encaps}(mpk, id^*)$
  - 4:  $b \leftarrow \mathcal{A}_2^{\text{Decaps}(msk, \cdot), \text{IDKEM.Extract}(msk, \cdot)}(st, c^*, k_0) \quad \triangleright \text{Queries to Decaps are defined in Algorithm 15.}$
  - 5: **return**  $b$
- 

The adversary's objective is to distinguish between the two scenarios presented in the attack games. The adversary's advantage, denoted as  $\text{Adv}_{\text{IDKEM}}^{\text{IND-CCA}}(\mathcal{A})$ , measures their ability to differentiate between the games. It is defined as  $|\Pr[W_0] - \Pr[W_1]|$ , where  $W_0$  and  $W_1$  represent the events where the adversary outputs 1 in Attack Game 0 and Attack Game 1, respectively. A secure *IDKEM* ensures that this advantage is negligible for any adversary  $\mathcal{A}$ .

---

**IDKEM Attack Game 1**


---

- 1:  $(msk, mpk) \leftarrow \text{IDKEM.KeyGen}(1^\lambda)$
  - 2:  $(id^*, st) \leftarrow \mathcal{A}_1^{\text{Decaps}(msk, \cdot), \text{IDKEM.Extract}(msk, \cdot)}(mpk)$
  - 3:  $(c^*, -) \leftarrow \text{IDKEM.Encaps}(mpk, id^*)$
  - 4:  $k_1 \xleftarrow{\$} K \quad \triangleright \text{Randomly chosen key from key space } K$
  - 5:  $b \leftarrow \mathcal{A}_2^{\text{Decaps}(msk, \cdot), \text{IDKEM.Extract}(msk, \cdot)}(st, c^*, k_1)$
  - 6: **return**  $b$
- 

---

**Algorithm 15** Decapsulation Query for *IDKEM* Attack Games

---

- 1:  $\text{Decaps}(msk, id, c)$ :
  - 2:  $sk_{id} \leftarrow \text{IDKEM.Extract}(msk, id)$
  - 3: **return**  $\text{IDKEM.Decaps}(sk_{id}, c)$
- 

In the context of identity-based *KEM* attack games, the adversary can select the identity it wants to target. Initially, the adversary is given the master public key ( $mpk$ ) and is allowed to send decapsulation queries to an oracle, as defined in Algorithm 15. Additionally, the adversary can query the oracle for the output of *IDKEM.Extract*, obtaining the secret key for any identity. After gathering this information, the adversary selects a target identity  $id^*$ . The adversary then receives a ciphertext associated with the chosen identity and either the secret encapsulated in the ciphertext or an unrelated secret, chosen uniformly at random. The adversary's goal is to distinguish between these two scenarios. An example of identity-based *KEM* used in this article is the DLP-IBE proposed in (DUCAS et al., 2014) by Ducas, Lyubashevsky and Prest.

### 2.2.6 Message Authentication Codes (MACs)

A Message Authentication Code (MAC) ensures the integrity and authenticity of a message. Defined over the sets  $(K, M, T)$ , a MAC consists of the following algorithms:

- **Key Generation:**  $MAC.KeyGen(1^\lambda) \rightarrow key$ : Given the security parameter  $1^\lambda$ , this algorithm generate a key  $key \in K$ .
- **Signing:**  $MAC.Sign(key, msg) \rightarrow tag$ : Given inputs  $key \in K$  and  $msg \in M$ , this algorithm generates a tag  $tag \in T$ .
- **Verifying:**  $MAC.Verify(key, msg, tag) \rightarrow b$ : Given a key  $key \in K$ , a message  $msg \in M$ , and a tag  $tag \in T$ , this algorithm outputs a value  $b$ , indicating either accept or reject.

A tag generated by  $MAC.Sign$  for a message is always accepted when verified with the same message and key.

Later, in Chapter 5, when presenting our hybrid construction, we assume that a MAC is secure if no adversary can forge an accepted tag for a chosen message, even when allowed to manipulate half of the bits of the key  $k$ , query a single signature, and perform a polynomial number of verification queries, manipulating and changing half of the key bits they control. The advantage of an adversary  $A$  is the probability of forging an accepted tag in the attack game illustrated below.

---

#### MAC Attack Game

---

```

1:  $k \leftarrow MAC.KeyGen(1^\lambda)$ 
2:  $k_1 \parallel k_2 \leftarrow k$ 
3:  $(msg^*, b, k_b, st) \leftarrow \mathcal{A}_1(1^\lambda)$   $\triangleright$  First execution of adversary. Its state can be stored in  $st$ .
4: if  $b = 0$  then
5:    $k^* \leftarrow k_b \parallel k_2$ 
6: else
7:    $k^* \leftarrow k_1 \parallel k_b$ 
8: end if
9:  $tag^* \leftarrow MAC.Sign(k^*, msg^*)$ 
10:  $(msg', tag') \leftarrow \mathcal{A}_2^{Verify(\cdot, \cdot)}(st, tag^*)$   $\triangleright$  Now adversary tries to produce a forgery.
11: return  $MAC.Verify(k^*, msg', tag')$ 

```

---

In the MAC attack game, the adversary submits a message in a query and selects half of the bits of the MAC key. In response, the adversary receives the computed tag for the queried message. With this information, and the ability to verify MAC tags, the adversary's goal is to produce a forgery – a valid tag for a new message. The adversary's advantage is the probability of successfully generating such a valid tag.

**Algorithm 17** Verification Query for MAC Attack Game

---

```

1: Verify( $k_b, msg, tag$ ):
2: if  $b = 0$  then
3:    $k' \leftarrow k_b \parallel k_2$ 
4: else
5:    $k' \leftarrow k_1 \parallel k_b$ 
6: end if
7: return MAC.Verify( $k', msg, tag$ )

```

---

**2.2.7 Signature Schemes**

A signature scheme is defined by the following algorithms, operating over the sets  $(M, S)$ , where  $M$  is the set of all possible messages, and  $S$  is the set of signatures:

- **Key Generation:**  $SIG.KeyGen(1^\lambda) \rightarrow (pk, sk)$ : A probabilistic algorithm that, given a security parameter  $1^\lambda$ , generates a cryptographic key pair  $(pk, sk)$ , where  $pk$  is the public key and  $sk$  is the secret key;
- **Signing:**  $SIG.Sign(sk, msg) \rightarrow sig$ : A probabilistic algorithm that, given a secret key  $sk$  and a message  $msg \in M$ , produces a signature  $sig \in S$ ;
- **Signature Verification:**  $SIG.Verify(pk, msg, sig) \rightarrow b$ : A deterministic algorithm that, given a public key  $pk$ , a message  $msg \in M$ , and a signature  $sig \in S$ , returns a binary value  $b$ . If  $b = 1$ , the signature is valid (accept); otherwise, it is invalid (reject).

A signature generated by `Sign` for a given message is always accepted when verified with the same message and key.

The security of a signature scheme is evaluated under the Unforgeability under Chosen Message Attacks (UF-CMA) model, formalized in (GOLDWASSER et al., 1988). In this model, the adversary's goal is to produce a forged signature. The adversary can adaptively request signatures for any message, but the forged signature must be for a new message – one that has not been previously queried. The adversary's advantage is the probability of successfully returning 1 in the attack game described below:

**Signature Attack Game**


---

```

1:  $(pk, sk) \leftarrow SIG.KeyGen(1^\lambda)$ 
2:  $Q \leftarrow \emptyset$ 
3:  $(msg^*, sig^*) \leftarrow \mathcal{A}^{SigningQuery(sk, \cdot)}(1^\lambda, pk)$ 
4: if  $SIG.Verify(pk, msg^*, sig^*) = \text{accept}$  and  $msg^* \notin Q$  then
5:   return 1
6: else
7:   return 0
8: end if

```

---

**Algorithm 19** Signing Query for Signature Attack Game

---

```

1: SigningQuery( $sk, msg$ ):
2:  $Q \leftarrow Q \cup \{msg\}$ 
3: return  $SIG.Sign(sk, msg)$ 

```

---

A signature scheme is considered secure under the UF-CMA model if, for any adversary  $\mathcal{A}$ , their advantage is always a negligible function of the security parameter  $\lambda$ . This probability is denoted as  $Adv_{SIG}^{UF-CMA}(\mathcal{A})$ .

## 2.2.7.1 ECDSA Signature Scheme

An example of a signature scheme is the ECDSA, which operates over a group  $G$  derived from an elliptic curve. We treat  $G$  as a multiplicative group with order  $q$ , and  $g$  as one of its generators. ECDSA signatures are defined over  $\mathbb{Z}_q^2$ , where  $q$  is the order of the elliptic curve. A hash function  $Hash : M \rightarrow \mathbb{Z}_q$  is used, where  $M$  represents the message space. The algorithms are as follows:

**Algorithm 20** ECDSA.Sign( $sk, msg$ )  $\rightarrow (r, s)$ 


---

```

1:  $x \leftarrow sk$ 
2: repeat
3:    $k \xleftarrow{\$} \mathbb{Z}_q$ 
4:    $y' \leftarrow g^k$ 
5:   Let  $r$  be the  $x$ -coordinate of the point  $y'$  on the elliptic curve  $G$  modulo  $q$ 
6:    $s \leftarrow (Hash(msg) + rx) k^{-1} \pmod{q}$ 
7: until  $r \neq 0$  and  $s \neq 0$ 
8: return  $(r, s)$ 

```

---

**Algorithm 21** ECDSA.Verify( $pk, msg, (r, s)$ )  $\rightarrow b$ 


---

```

1:  $y \leftarrow pk$ 
2: if  $s = 0$  or  $r = 0$  then
3:   return reject
4: end if
5:  $a \leftarrow Hash(msg) \cdot s^{-1} \pmod{q}$ 
6:  $b \leftarrow r \cdot s^{-1} \pmod{q}$ 
7:  $y' \leftarrow g^a \cdot y^b \pmod{q}$ 
8: if  $y'$  is the point at infinity in  $G$  then
9:   return reject
10: end if
11: if the  $x$ -coordinate of  $y'$  on  $G$  modulo  $q$  equals  $r$  then
12:   return accept
13: else
14:   return reject
15: end if

```

---

## 2.3 PROTOCOLS

### 2.3.1 Bitcoin Backbone Protocol

We present here a basic notation to describe a blockchain using the model from (GARAY et al., 2015) that describes the Bitcoin backbone protocol.

The blockchain is described with the help of two hash functions  $H$  and  $G$ . Both are modeled as random oracles. Each query to the oracle representing  $H$  must pass two values: a counter and the data to be hashed. In each round, the parties that participate in the protocol must initialize their counters to zero and increment it after each new query to  $H$ . A maximum of  $q$  queries is allowed for each party in each round. The query limit is only for new queries: verifying previously computed hashes (even when initially computed by other parties) does not require the counter to be incremented. Therefore, it is not subject to the limit  $q$ . This is required to model a limit to the number of hash queries when mining new blocks.

A block is a triple  $B = (s, x, ctr)$ . The element  $s$  is taken from the set of possible digests for the hash function  $H$ . The element  $x \in \{0, 1\}^*$  is the block content. And  $ctr \in \mathbb{N}$  is the counter value for  $H$ .

We say that a block is valid if:

$$\text{validblock}_q^d(B) := (H(ctr, G(s, x)) < d) \wedge (ctr < q) = 1$$

The value  $d$  models how difficult the puzzle is to be solved as proof of work and is updated after each round. The value  $q$  models the expected time for mining a block.

A list of blocks is a blockchain:

$$C = ((s_0, x_0, ctr_0), (s_1, x_1, ctr_1), \dots)$$

The last block of a blockchain is its head, denoted by  $\text{Head}(C)$ . Any blockchain  $C$  with a head  $(s, x, ctr)$  can be extended to a longer chain  $C' := C || B'$  appending a new valid block  $B' = (s', x', ctr')$  such that  $s' = H(ctr, G(s, x))$ . A blockchain is valid if, for all blocks except the first one, we have that  $s_i = H(ctr_{i-1}, G(s_{i-1}, x_{i-1}))$ .

### 2.3.2 The Transport Layer Security 1.3 Protocol

Transport Layer Security (TLS) is a widely used cryptographic protocol that ensures secure communication over computer networks. It plays a vital role in protecting web pages and enabling secure machine-to-machine interactions. TLS 1.3, specified in RFC 8446 (RESCORLA, 2018), enables secure message exchange over untrusted networks by establishing a shared secret between communicating parties. Unlike protocols that rely solely on symmetric encryption, TLS combines multiple cryptographic techniques.

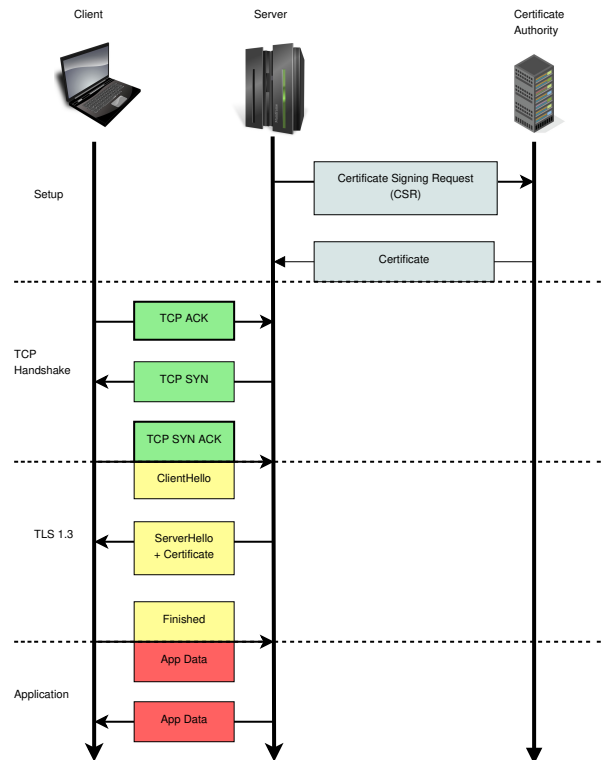


Figure 1 – Diagram illustrating the usage of TLS 1.3. The server must first obtain a certificate from a Certificate Authority before initiating the protocol, typically by issuing a Certificate Signing Request (CSR) containing its public key and demonstrating possession of the corresponding private key as part of the certification process. The Transmission Control Protocol (TCP) handshake, typically managed by the operating system, establishes the connection between the client and server. TLS 1.3 then secures the connection, enabling secret sharing between the client and server, which allows encryption of application data.

The TLS standard allows for many variations. In some cases, only the server is authenticated during client-server communication, while in others, mutual authentication is used. Additionally, extensions exist that enable clients to cache server certificates. For simplicity, this thesis focuses on a simplified description of TLS 1.3 that aligns with its most common use case. We assume that only the server is authenticated, the client initiates the communication, and no cached certificates are used. Figure 1 illustrates an example of TLS usage under these assumptions. In this work, we also simplify the protocol by omitting certain additional messages exchanged during the handshake, such as the Encrypted Extensions message, when they are not relevant to our proposal.

Before using the protocol, the server performs a setup step. During this step, the server generates ECDSA signature keys: a secret key  $x$ , which is a randomly chosen scalar, and a public key  $g^x$ , which is the result of scalar multiplication of  $x$  with the elliptic curve generator point  $g$ . The server then connects with a Certification Authority (CA), where it proves its identity by providing documentation or credentials. It sends its public key to the CA and signs a challenge or certificate signing request (CSR) provided

by the CA, demonstrating possession of the corresponding secret key. In response, the CA signs the server public key, producing the server certificate.

The protocol relies on the principle of transitive trust: if a client trusts a particular Certification Authority (CA), it will also trust other CAs certified by that trusted CA. This approach enables the creation of a hierarchical trust structure, often referred to as a 'tree of trust,' starting from a small number of root CAs that are widely trusted by many entities. A CA can demonstrate its trustworthiness by providing a certificate chain, where each certificate in the chain is issued by another CA, and the chain ultimately leads back to a trusted root CA. Therefore, in TLS 1.3 usually each server must send to the client not a single certificate, but a certificate chain to prove to the client that it is a member of his tree of trust.

For a certificate to be considered valid, it must not appear on a revocation list issued by the CAs. To eliminate the need for clients to check these lists directly, servers can provide short-term declarations, known as Online Certificate Status Protocol (OCSP) stapling or similar mechanisms, signed by the CA. These declarations confirm that the certificate has not been revoked within a specified time interval.

Assuming the server holds a valid, non-revoked certificate, a typical TLS 1.3 handshake proceeds using the Diffie Hellman Key-Exchange as follows, after a transport-layer connection (e.g., TCP) has been established:

1.  $C \rightarrow S$ : "ClientHello"

The client sends a message to the server containing a list of supported cryptographic algorithms. It also selects a random private exponent  $x_c$  and computes the corresponding elliptic curve point  $y = g^{x_c}$ . This curve point, along with the supported algorithm list, is included in the ClientHello message.

2.  $S \rightarrow C$ : "ServerHello"

If the server supports one of the client's proposed algorithms, it selects a compatible algorithm and computes using the client's curve point  $y$  and the server's private key  $x$ , the secret value  $y^x$ . The server then sends its certificate (containing its public key  $g^x$ ) to the client and signs the handshake transcript to prove ownership of the private key associated with the certificate, thereby authenticating itself.

After this single round-trip, the client computes the same secret value  $y^x = g^{x \cdot x_c}$  using the server's public key  $g^x$  together with  $x_c$ . This shared secret is used to derive keys for a symmetric encryption scheme, ensuring that all subsequent communication between the client and server is encrypted and authenticated. Both parties also compute a MAC over the transcript of the messages to prove the liveness and ownership of the derived symmetric key, sending this MAC in a final "Finished" message.

### 2.3.3 The KEMTLS Protocol

KEMTLS is a proposed protocol designed to establish secure connections using post-quantum algorithms, with the potential to replace the current TLS 1.3 standard. At a high level, the protocol operates as follows (SCHWABE et al., 2020):

1.  $C \rightarrow S$  : “ClientHello”

The Client sends a “ClientHello” message containing a newly generated ephemeral public key  $pk_e$ , created by running  $KEM.KeyGen(1^\lambda)$  (KEM algorithms were presented in Section 2.2.4).

2.  $S \rightarrow C$  : “ServerHello”

The server responds with a “ServerHello” message that includes:

- An ephemeral key and ciphertext  $(k_e, c_e)$  generated using the  $KEM$  encapsulation function with the client’s ephemeral public key  $(pk_e)$ ;
- A certificate chain that includes the server’s public key  $(pk_S)$ .

3.  $C \rightarrow S$  : “ClientKeyExchange”

The client replies with a “ClientKeyExchange” message containing  $c_S$ , which encapsulates a secret  $k_S$  computed by running  $KEM.Encaps(pk_S)$ .

The final shared secret is derived by combining  $k_e$  and  $k_S$ . With both secrets known, the client can begin sending encrypted application data alongside the “ClientKeyExchange” message, even before the handshake is fully completed.

4.  $S \rightarrow C$  : “Finished”

The server sends a “Finished” message to confirm the successful exchange of secrets. At this point, both parties have established the shared secret. Optionally, the server can send encrypted application data along with the “Finished” message.

Although KEMTLS requires two round trips to complete the full handshake – seemingly less efficient than the single round trip of TLS 1.3 – this limitation can be mitigated. The client has the option to send application data concurrently with the “ClientKeyExchange” message, reducing overall latency before the handshake is fully completed.

Additionally, if the client already possesses the server’s public key, it can include the  $c_S$  encapsulation in the initial “ClientHello” message. This allows the server to respond with both the “ServerHello” and the “Finished” message within the second round trip, achieving secret sharing in a single round trip.

While KEMTLS supports additional features, such as client authentication, this thesis focuses on the core mechanics of the protocol described above. The principles introduced can be extended to incorporate these additional functionalities as needed.

## 2.4 CONCLUSION

This chapter has established the mathematical and cryptographic foundations necessary for understanding the post-quantum adaptations proposed in subsequent chapters. We have presented formal definitions for a diverse set of cryptographic primitives, including chameleon hash functions with their various security properties, key encapsulation mechanisms, identity-based encryption schemes, message authentication codes, and digital signature schemes.

The exploration of these primitives revealed important distinctions between classical and post-quantum approaches to cryptography. Traditional schemes based on discrete logarithm and integer factorization problems offer computational efficiency and compact representations but remain vulnerable to quantum attacks. Meanwhile, post-quantum alternatives provide security against quantum adversaries at the cost of increased computational overhead and larger key sizes.

We have also examined two protocols that will serve as case studies throughout this thesis: the Bitcoin backbone protocol, which provides the foundation for blockchain technology, and the Transport Layer Security protocol, which secures a significant portion of internet communications. By understanding their structures and security requirements, we can better appreciate the challenges involved in transitioning these protocols to post-quantum security.

The KEMTLS protocol, introduced as a post-quantum alternative to TLS 1.3, demonstrates how cryptographic primitives can be reconfigured to maintain security in a post-quantum environment while addressing practical implementation concerns. This protocol serves as an important reference point for our own contributions in enhancing post-quantum security without disrupting existing infrastructure.

With these preliminaries established, we now proceed to examine the specific challenges in adapting legacy cryptographic systems to post-quantum security, beginning with a systematic literature review of chameleon hash functions and their applications in Chapter 3.

### 3 LITERATURE REVIEW

#### Executive Summary

This chapter presents a comprehensive systematic mapping of chameleon hash functions, a specialized cryptographic primitive central to our proposed solutions. Through rigorous analysis of 353 relevant publications, we categorize and evaluate the evolution, security properties, and applications of these functions.

#### Key Findings:

- Identification of diverse chameleon hash variants (traditional, identity-based, labelled, secret-coin) and their security foundations;
- Discovery of the underutilized *preimage property* in certain constructions, which enables efficient computation of first preimages with trapdoor knowledge;
- Analysis of practical limitations in existing constructions that restrict their application in scenarios requiring adaptive security in the collision resistance.

**Research Impact:** This systematic review reveals critical research gaps in chameleon hash constructions and applications, directly informing our blockchain redactability proposal in Chapter 4 and establishing the theoretical foundation for our post-quantum adaptation strategies. The identified preimage property emerges as a particularly promising avenue for novel cryptographic applications in post-quantum contexts.

#### 3.1 INTRODUCTION

Chameleon hash functions represent a specialized cryptographic primitive with unique properties that make them particularly valuable in post-quantum cryptographic adaptations. These functions combine the collision-resistance of traditional cryptographic hash functions with the additional capability of computing collisions when in possession of a trapdoor key. This distinctive characteristic enables numerous cryptographic applications, from redactable blockchains and sanitizable signatures to identity-based encryption schemes and secure message authentication codes.

Despite their growing importance in modern cryptographic systems, the literature on chameleon hash functions has developed in a fragmented manner across multiple research communities. Different constructions, security assumptions, and application scenarios have emerged without a comprehensive framework to organize this knowledge. This fragmentation has impeded both theoretical advancement and practical implementation of chameleon hash-based solutions, particularly in the context of post-quantum security adaptations.

To address this critical gap in the cryptographic literature, we conducted a Systematic Mapping Study (SMS), a methodologically rigorous approach to synthesizing

existing scientific knowledge and identifying research opportunities. Unlike traditional literature reviews, which often focus on narrow research questions, a SMS provides a broader categorization of the available literature, identifying clusters of evidence and areas where primary research is lacking. This approach is particularly well-suited to emerging fields with heterogeneous research outputs, such as chameleon hash functions.

Our methodology follows the established SMS framework developed by (KITCHENHAM et al., 2011) and refined for software engineering and computer science by (PETERSEN et al., 2015). This systematic approach ensures comprehensive coverage and minimizes selection bias through a transparent, replicable process consisting of four distinct phases:

- **Definition of the Research Protocol:** We carefully formulated search keywords, directives, and expressions to identify relevant academic works across digital libraries and repositories. This phase also established precise inclusion and exclusion criteria to ensure systematic selection of publications for analysis;
- **Search Execution:** We implemented the defined search protocol across six major academic databases, gathering a comprehensive set of 740 initial results to ensure broad coverage of the literature;
- **Filtering:** The collected publications underwent methodical screening against our predetermined selection criteria, resulting in a refined dataset of 353 relevant works that make substantive contributions to understanding chameleon hash functions;
- **Analysis and Classification:** The filtered results underwent detailed analysis and systematic classification according to defined taxonomies, enabling identification of patterns, research trends, and knowledge gaps.

The comprehensive mapping presented in this chapter serves multiple purposes within our broader research on post-quantum cryptographic adaptations. First, it establishes a taxonomic framework for understanding the diverse definitions, security properties, and construction methods of chameleon hash functions. Second, it identifies specific properties, such as the preimage capability, that can be leveraged for novel applications in post-quantum settings. Finally, it provides the theoretical foundation necessary to develop the redactable blockchain and TLS protocol adaptations presented in subsequent chapters.

By organizing the disparate knowledge on chameleon hash functions into a coherent framework, this chapter not only addresses a significant gap in the cryptographic literature but also establishes the conceptual groundwork for the original contributions of this thesis. The insights gained from this systematic mapping directly inform our

approach to creating more efficient and secure post-quantum adaptations for legacy cryptographic systems.

### 3.2 RESEARCH QUESTIONS

In our systematic mapping, we aimed to address the following four research questions:

**RQ1: What are the different definitions of chameleon hash schemes described in the literature?**

A cryptographic scheme consists of a set of algorithms designed to fulfill specific requirements. Chameleon hash schemes typically include algorithms for key generation, hash computation, and collision generation using a secret key. However, some studies introduce alternative definitions, adding algorithms or varying the inputs for each algorithm. Identifying these variations allows us to catalog the distinct chameleon hash schemes in the literature.

**RQ2: What are the different constructions of chameleon hash functions proposed in the literature?**

Chameleon hash functions can be constructed using diverse methods, each grounded in specific cryptographic assumptions. This question focuses on organizing these constructions based on their underlying security assumptions, enabling a clearer understanding of the theoretical foundations of each approach.

**RQ3: What use cases are proposed for chameleon hash functions?**

To answer this question, we reviewed each selected study to identify applications proposed for chameleon hash functions. Documenting these use cases provides insights into their practical utility and adoption.

**RQ4: What properties are required of chameleon hash schemes for each identified application?**

For articles proposing specific use cases, we analyzed the required properties of the chameleon hash functions. These properties were documented to highlight the functional and security needs of each application.

### 3.3 SEARCH AND SELECTION STRATEGY

The search string used in this work is:

```
"trapdoor hash" OR "chameleon hash" OR "chameleon hashing" OR  
"chameleon hashes"
```

This research was conducted at the beginning of 2022 and was limited to results published up to 2021. The search was performed across six academic databases:

- **IEEE XPlore:** This database gave us 69 results. The URL for this search was:

```
https://link.springer.com/search?date-facet-mode=between&
facet-start-year=1900&sortOrder=oldestFirst&facet-end-year=2021&
showAll=true&query=%28%22trapdoor+hash%22+OR++%22chameleon+hash%
22+OR++%22chameleon+hashing%22+OR+%22chameleon+hashes%22%29
```

- **ACM Database:** This database gave us 58 results. The URL for this search was:

```
https://dl.acm.org/action/doSearch?fillQuickSearch=false&
target=advanced&expand=dl&field1=AllField&text1=%22trapdoor+
hash%22+OR+%22chameleon+hash%22+OR+%22chameleon+hashing%22+
OR+%22chameleon+hashes%22&AfterMonth=1&AfterYear=1908&
BeforeMonth=12&BeforeYear=2021&startPage=0&sortBy=Ppub_asc
```

We also selected an option to exclude posters from the search.

- **Springer:** This database gave us 467 results. The URL for this search was:

```
https://link.springer.com/search?date-facet-mode=between&
facet-start-year=1900&sortOrder=oldestFirst&
facet-end-year=2021&showAll=true&query=%28%22trapdoor+hash%22
+OR++%22chameleon+hash%22+OR++%22chameleon+hashing%22+OR+%22
chameleon+hashes%22%29
```

- **Science Direct:** This database gave us 99 results. The URL for this search was:

```
https://www.sciencedirect.com/search?qs=%22trapdoor%20hash
%22%20OR%20%22chameleon%20hash%22%20OR%20%22chameleon%20
hashing%22%20OR%20%22chameleon%20hashes%22&pub=&date=&
authors=&affiliations=&volume=&issue=&page=&tak=&title=&
references=&docId=&sortBy=date&show=100
```

- **Cryptology Eprint Archive:** This database gave us 43 results. The URL for this search was:

<https://eprint.iacr.org/search?q=%22trapdoor+hash%22+OR+%22chameleon+hash%22+OR+22chameleon+hashing%22+OR+22chameleon+hashes%22&title=&authors=&category=&submittedafter=&submittedbefore=&revisedafter=&revisedbefore=>

- **MDPI Open Access Journals**> This database gave us 4 results. The URL for this search was:

<https://www.mdpi.com/search?q=%22trapdoor+hash%22+OR+%22chameleon+hash%22+OR+%22chameleon+hashing%22+OR++%22chameleon+hashes%22>

The exclusion criteria for the study were as follows:

**Initial Exclusion Criteria:**

1. Duplicate papers (37 works were excluded under this criteria).
2. Papers not in English (no work was excluded under this criteria).
3. Secondary or tertiary works (32 works were excluded under this criteria).
4. Editorials, poster sessions, prefaces, opinion letters, slide presentations, keynotes, and book chapters that were not original research articles (8 works were excluded under this criteria).
5. Master's theses or doctoral dissertations (no work was excluded under this criteria).

**Secondary Exclusion Criteria:**

After the initial filtering, a second round of revision was conducted to exclude:

1. Papers mentioning chameleon hash schemes only in the introduction or references, without employing or proposing new applications (267 works were excluded under this criteria).
2. Papers offering criticism or cryptanalysis of previous works, without a new application (5 works were excluded under this criteria).
3. Papers defining a chameleon hash as a protocol rather than a cryptographic scheme (1 work was excluded under this criteria).
4. Papers that primarily present opinions, narrate experiences, or advocate positions without proposing new constructions or conducting original research (4 works were excluded under this criteria).

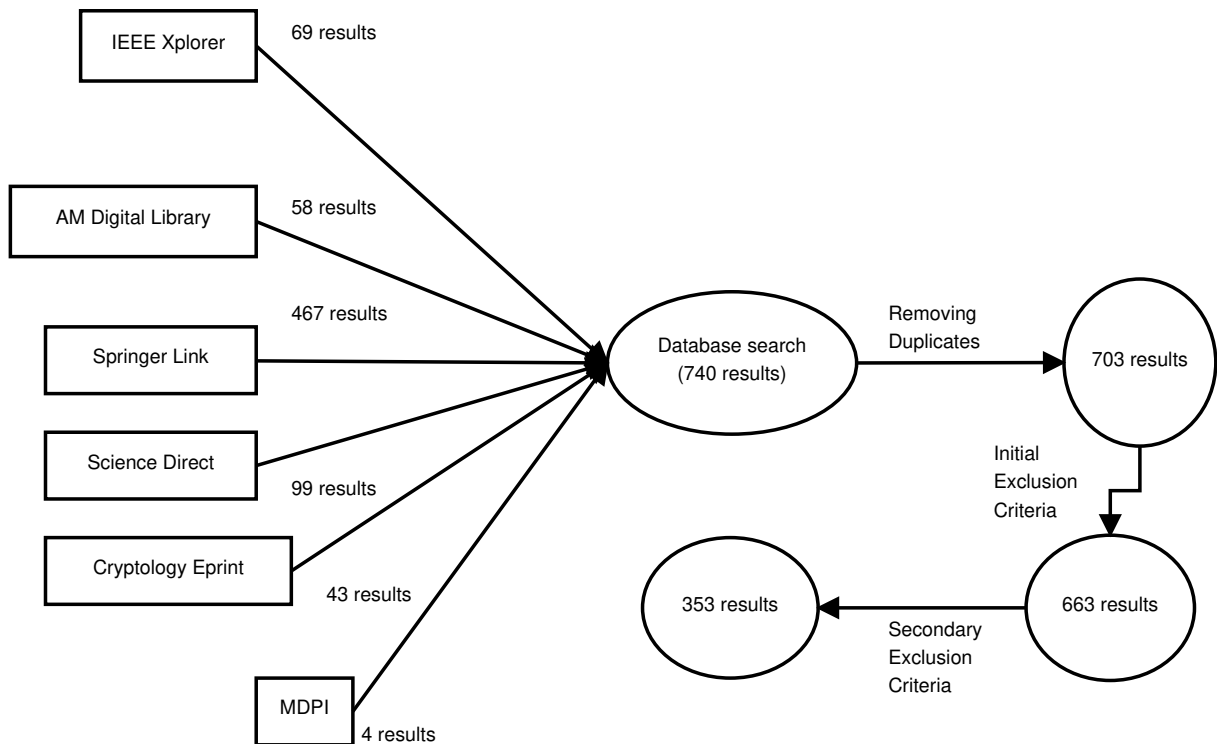


Figure 2 – Diagram with number of works selected by our systematic mapping in each step.

5. Papers outside the fields of computer science, engineering, or mathematics (5 works were excluded under this criteria).
6. False positives: papers that after initial reading revealed that they did not talk about chameleon hash functions, but were incorrectly filtered because of similar words or because they use another primitive similarly named (28 works were excluded under this criteria). Most of the works excluded under this criteria used a primitive called “trapdoor hash”, which was first defined in 2019 in (DÖTTLING et al., 2019). But several older works used this name as a synonymous of chameleon hash functions, which justifies including it in the search string.

Papers that did not meet any of these exclusion criteria were included in the study.

The Figure 2 illustrates the number of articles remaining after each step of the selection process.

### 3.4 RESULTS

In Figure 3, we illustrate the number of selected results by year. We begin in 1998, as this marks the first publication on chameleon hash schemes, where the concept was initially introduced.

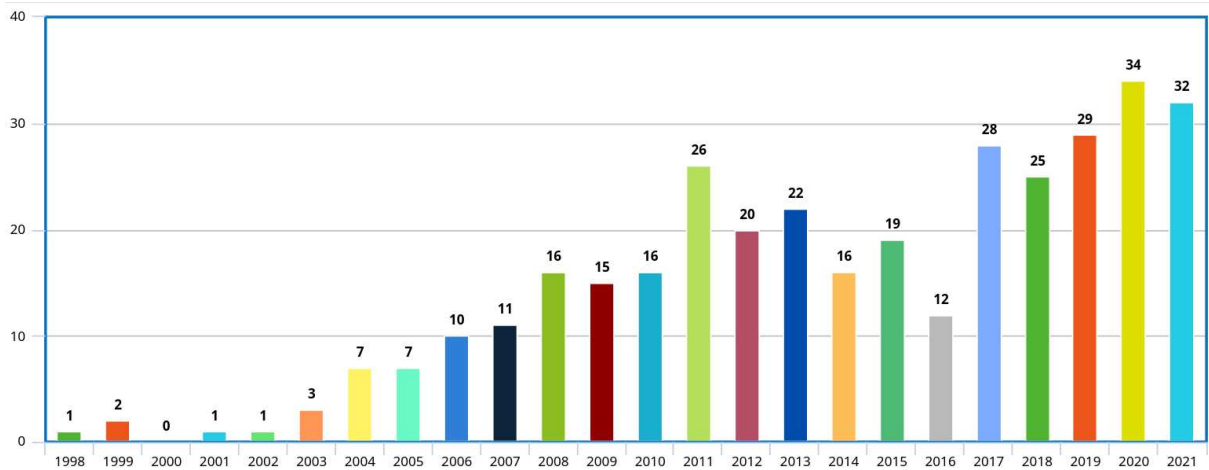


Figure 3 – Selected publications by year.

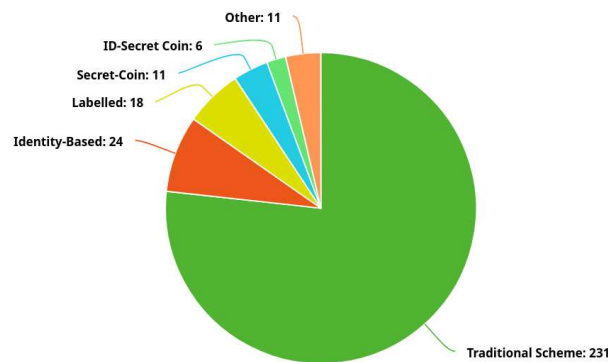


Figure 4 – Different chameleon hash scheme definitions by publication.

### 3.4.1 Scheme Definitions

Figure 4 highlights the variety of cryptographic schemes referred to as 'chameleon hash,' each with distinct formal definitions. Below, we categorize and summarize these schemes:

- **Traditional Chameleon Hash:** This widely used definition is detailed in Section 2.2.2 and forms the basis for many subsequent schemes. The difficulty in ensuring collision resistance for this scheme in scenarios where collisions produced with a trapdoor are leaked is the main motivation for almost all other schemes.
- **Identity-Based Chameleon Hash:** This scheme incorporates a unique string, termed an "identity," as an additional parameter for the hashing and collision algorithms. The collision computation is split into two stages: generating an identity-specific trapdoor using the trapdoor key and subsequently computing collisions with the identity-specific trapdoor.

A central authority can issue identity trapdoors to authenticated parties, ensuring the correct trapdoor is provided only to verified identities.

- **Labelled Chameleon Hash:** Similar to the identity-based scheme, users supply unique strings (referred to as 'tags' or 'labels') for the hashing algorithm. Unlike the identity-based scheme, trapdoor computation cannot be delegated to a central authority; instead, users generate and manage their own unique strings.
- **Secret Coin Chameleon Hash:** This is the scheme described in Section 2.2.3.
- **ID-Secret Coin:** This is the same scheme than the Secret Coin Chameleon Hash, except that the hashing and collision algorithms require an "identity" as additional parameter, and the collision is also split in two computations exactly like in the Identity-Based Chameleon Hash.
- **Other Variations:**
  - **Ephemeral-Trapdoor Chameleon Hash:** Unlike other schemes, this variation requires cooperation between the trapdoor owner and the original hash generator to compute collisions. Five articles explore this scheme.
  - **Certificateless Chameleon Hash Scheme:** Addressing security concerns in identity-based schemes, this approach employs an interactive protocol between a central authority and the trapdoor recipient to ensure the authority cannot reconstruct trapdoors. The papers (HOU, 2012) and (XU et al., 2020) propose this scheme.
  - **Hybrid Identity-Based and Secret-Coin Scheme:** Three articles combine these two approaches to achieve specific functionalities.
  - **Dual Long-Term Trapdoor Scheme:** This scheme uses two pairs of keys for hash computations, allowing one key pair to be periodically discarded and replaced. A single article presents this concept.

### 3.4.2 Security Assumptions

We can also categorize the various chameleon hash constructions based on the security assumptions underpinning their security. Figure 5 illustrates the number of articles associated with each security assumption. Works that treat chameleon hash functions as black-box primitives without specifying particular constructions are omitted from this count.

The main assumptions are as follows:

- **Discrete Logarithm:** This is one of the simplest and most widely used constructions. A traditional chameleon hash can be defined as:

$$CH.Hash(y, m, r) = g^m y^r$$

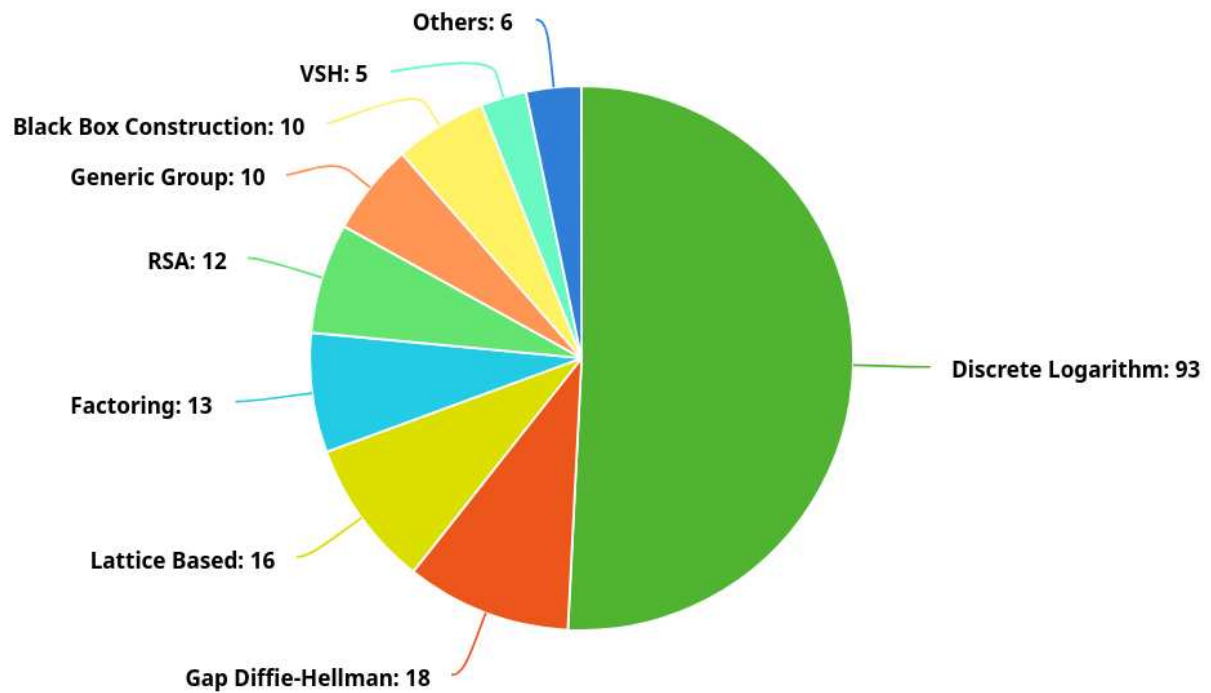


Figure 5 – Different security assumptions by publication. Publications that did not suggest a specific chameleon hash construction were omitted.

where  $g$  is the generator of a multiplicative group, and  $y$  is the public key with the corresponding secret key  $x$  such that  $g^x = y$ . Variations of this definition exist, but all rely on using a discrete logarithm as the secret key and a group where computing discrete logarithms is hard. While this assumption is predominantly used for traditional schemes, labelled and secret-coin schemes have also been built on it.

- **Lattice-Based Assumptions:** These post-quantum constructions derive their security from the hardness of certain lattice-based problems. Since quantum computers are not believed to break these assumptions, they provide quantum-resistant security. All constructions found under this assumption were traditional schemes.
- **Generic Group Model:** These constructions rely on the Generic Group Model, a non-standard model that assumes random encoding of group elements. While the underlying security is based on the discrete logarithm assumption, it also depends on attackers being unable to extract meaningful information from the encoding. All constructions found using this model were traditional schemes.
- **Factoring Assumption:** Security is derived from the difficulty of factoring semiprime numbers. This assumption has been used to create traditional, labelled, and identity-based schemes.

- **RSA Assumption:** These schemes base their security on the RSA problem. Specifically, given a composite  $n$  and a fixed exponent  $e$ , the problem is to find  $d$  such that  $(M^e)^d = M$ . Both traditional and identity-based constructions have been built on this assumption.
- **Gap-Diffie Hellman:** These constructions use bilinear pairings and assume that the Computational Diffie-Hellman (CDH) problem is hard in a multiplicative group, even if the Decisional Diffie-Hellman (DDH) problem is not. All constructions under this assumption are identity-based schemes.
- **Black-Box Constructions:** These assume the existence of common cryptographic primitives (e.g., secure encryption schemes and non-interactive zero-knowledge proofs) and use them as building blocks. Combining these components allows the construction of schemes like the Secret-Coin Chameleon Hash.
- **Computational Diffie-Hellman:** These are the constructions based on the hardness of solving the Computational Diffie-Hellman Problem.
- **Others:** A few less common assumptions were also used in some constructions:
  - **Paillier’s Cryptosystem:** Relies on the difficulty of solving modular equations like  $z \equiv y^n \pmod{n^2}$ . Two works proposed traditional schemes under this assumption.
  - **VSH:** A family of fast chameleon hash constructions based on the difficulty of finding modular square roots of “smooth” numbers modulo a composite. They are a variation of the VSH (Very Smooth Hash) family of hash functions with trapdoor. Six works used this assumption for traditional schemes.
  - **Unique Assumptions:** Certain works relied on rare assumptions, such as modular bivariate polynomials (BONEH; CORRIGAN-GIBBS, 2014), the Symmetric External Diffie-Hellman Assumption (WANG, Y.; TANAKA, 2015), or the Decision Linear Assumption (ISHIZAKA; MATSUURA, 2018). Other examples include homomorphic one-way functions (ALAMATI et al., 2019), the Strong RSA Assumption (CHEVALLIER-MAMES; JOYE, 2006), and homomorphic hash functions (THANALAKSHMI et al., 2021).

### 3.4.3 Applications and Relevant Properties

In Figure 6, we illustrate the number of works proposing different applications for chameleon hash functions. To answer research questions 3 and 4, we will enumerate the found applications and their corresponding properties. All chameleon hash applications require at least the collision resistance property, so we omit this property when describing specific properties for each application.

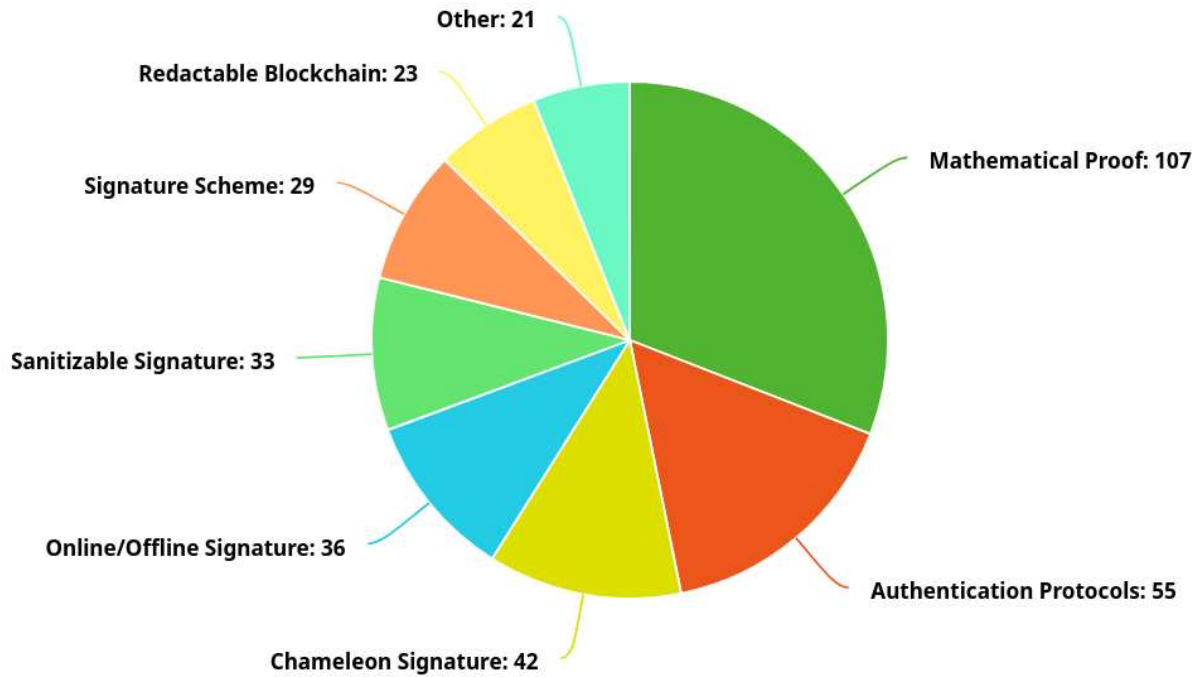


Figure 6 – Different applications for chameleon hash schemes by publication. Publications that did not suggest a specific application were omitted.

- **Mathematical Proofs:** Proving the security of a signature scheme or other cryptographic scheme often requires non-standard models, such as the random oracle model, which assumes that a given hash function behaves like an idealized random function. In certain cases, replacing the hash function with a chameleon hash function allows mathematical proofs to work without relying on idealized random functions. This substitution can either prove the security of a cryptographic primitive or enhance its security when combined with the chameleon hash. Mathematical proofs that rely on concrete, non-idealized functions are generally more desirable.

These proofs typically depend on the following essential properties of chameleon hash functions:

- **Weak Uniformity:** A Chameleon Hash Scheme  $CH$  defined over  $(M, R, D)$  has the **weak uniformity property** if given any pair  $(ek, tk)$  returned by  $CH.KEYGEN(1^\lambda)$ , for all pair of messages  $msg_1, msg_2 \in M$ , both functions  $f_1 = CH.HASH(ek, msg_1, \cdot)$  and  $f_2 = CH.HASH(ek, msg_2, \cdot)$  have the same probability distribution if the input is chosen uniformly at random from  $R$ .
- **Strong Uniformity:** A chameleon hash scheme  $CH$  defined over  $(M, R, D)$  has the **strong uniformity property** if given  $(ek, tk) \xleftarrow{\$} CH.KEYGEN(1^\lambda)$ , for all messages  $msg \in M$ , the probability distribution of  $f = CH.HASH(ek, msg, \cdot)$  if the randomness input is chosen uniformly at random from  $R$  is the same

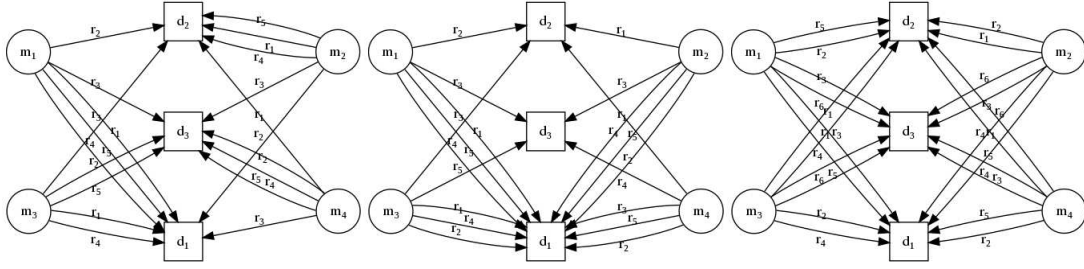


Figure 7 – Graphs representing toy examples of chameleon hash functions. Each message  $m_i$ , represented by a circle, is connected to a digest  $d_i$  by an edge labeled  $r_i$  if, for some fixed key  $ek$ , we have  $CH.Hash(ek, m_i, r_i) = d_i$ . In the first image, the chameleon hash does not have the uniformity property. In the second image, it satisfies the weak uniformity property. The third image illustrates the strong uniformity property.

than the probability distribution of choosing a digest from  $D$  uniformly at random.

Figure 7 illustrates the difference between non-uniform, weakly uniform, and strongly uniform chameleon hash functions.

- **Preimage Property:** Enables the computation of both first and second preimages using the trapdoor key ( $tk$ ). This property is crucial for constructing programmable hash functions often required in proofs within the random oracle model.
- **Online/Offline Signatures:** Online/offline signatures allow a signer to precompute parts of the signature process during idle times (offline) and finalize the signature for a specific message later (online). To create an online/offline signature, the signer first generates a random message, computes its chameleon hash, and signs the resulting digest. When the actual message needs to be signed, the signer computes a collision that makes the precomputed signature valid for the true message. Almost all works follow this approach, with an interesting exception being (ZHANG, Y. et al., 2011), that uses chameleon hash to help precomputing verifications.

This approach is beneficial when generating collisions is significantly faster than computing a hash in the chameleon hash scheme. In such cases, the signer can precompute signatures during idle periods, allowing for faster finalization when a message needs to be signed.

Chameleon hash schemes used for online/offline signatures must satisfy the following properties:

- **Weak Uniformity:** Ensures that any message produces digests with the same probability distribution. This guarantees that there is no discernible dif-

ference between the precomputed signature (based on a random message) and the final signature for the actual message.

- **Fast Collisions:** The *CH.Collision* algorithm must be significantly faster than the *CH.Hash* algorithm, which ensures that running the second part of the signature computation will be faster than running a regular signing algorithm and keeps most work in the computation in the first part, when the computer is idle.
- **Chameleon Signatures:** It is derived by replacing the hash function in a signature scheme with a chameleon hash function, similar to online/offline signatures. However, the key distinction is that the trapdoor key ( $tk$ ) is given to the verifier rather than the signer.

In this scheme, the signature acquires unique properties:

- To sign a message, the signer must choose the intended recipient and use the recipient's chameleon hash key ( $ek$ ) to hash the message.
- The signature is considered legitimate by the recipient, who can verify its authenticity. However, the recipient cannot convince others of the signature's legitimacy because their ability to compute collisions undermines its trustworthiness when presented to third parties.

This ability to compute collisions introduces interesting applications:

- **Content Anonymization:** The recipient can replace the originally signed message with another colliding message, effectively anonymizing the content.
- **Delegated Signing:** In some proposals, the recipient's ability to compute collisions enables them to act as a delegated signer, producing valid signatures on behalf of the original signer.

Chameleon hash schemes used in chameleon signatures must satisfy the following properties:

- **Enhanced Collision Resistance:** While an adversary with access to a sample collision for a given digest may find new collisions for the same digest, they cannot generate collisions for a new, unrelated digest without a sample collision.
- **Weak Uniformity:** This property ensures that digests produced by the chameleon hash have the same probability distribution, regardless of whether they are generated by the signer or forged by the recipient. This indistinguishability guarantees that the signature cannot be externally verified as either original or forged.

- **Message Hiding:** This property allows anyone who finds a collision in the chameleon hash (even without the trapdoor key) to produce a new colliding message for the same digest. It is particularly useful in legal or dispute scenarios, enabling the signer to prove they did not sign a specific message if challenged.
- **Sanitizable Signatures:** This signature allows certain parts of a document to be altered by a trusted party without invalidating the signature. This is achieved by dividing the document into two parts during the signing process:
  - One part is hashed using a standard hash function.
  - The other part is hashed using a chameleon hash function, with the trapdoor key ( $tk$ ) securely held by a trusted anonymizer.

This setup enables the trusted anonymizer to modify specific, predefined parts of the document (hashed with the chameleon hash) while preserving the overall validity of the signature.

Some applications are:

- **Content Customization:** Adapting signed content for individual users while maintaining its authenticity.
- **Sensitive Information Redaction:** Anonymizing sensitive parts of documents without invalidating the signature, e.g., for privacy or compliance purposes.
- **Dynamic Certificates:** Creating certificates where users are permitted to update certain fields, such as their public keys, within a specified timeframe without invalidating the certificate.

When using ephemeral trapdoor chameleon hash schemes, it is possible to create invisible sanitizable signatures. These signatures conceal which parts of the document are sanitizable and which are immutable, enhancing the privacy and security of the sanitization process.

Sanitizable signatures rely on chameleon hash schemes with the following properties:

- **Full Collision-Resistance:** An attacker, even after observing sample collisions, must be unable to find new collisions for the same or other digests. This ensures that only the trusted anonymizer with the trapdoor key can modify the predefined parts of the document.

- **Weak Uniformity:** The digest must not leak information about the originally signed message, even if parts of the document are redacted or modified by the trusted anonymizer. This ensures privacy and indistinguishability of redacted content.
- **Signature Schemes:** In this type of application, the chameleon hash scheme is not merely combined with an existing signature scheme; instead, it serves as the foundation for designing entirely new signature schemes. By leveraging the properties of chameleon hash functions, both regular signatures as various specialized and innovative signature schemes can be constructed. For example:
  - **Group and Ring Signatures:** These allow a user to sign a message on behalf of a group to which they belong. The identity of the signer remains hidden within the group, providing anonymity while ensuring legitimacy.
  - **Message-Hiding Signatures:** These signatures allow a signer to prove the validity of a signature without revealing the original signed message. Such schemes are particularly useful in privacy-sensitive applications.
  - **Aggregate Signatures:** In aggregate signatures, multiple individuals can sign the same document without significantly increasing the total size of the combined signature. This is especially efficient for large-scale applications where multiple parties must jointly sign a document.

Besides the standard collision resistance, the following properties may be useful for some signature proposals:

- **D-Times Collision-Resistance:** Some applications require a chameleon hash function where up to  $d$  collisions can be revealed without compromising security. This property enables the creation of limited-use signatures, which can only be used up to  $d$  times. If the signature is used more than this, the user is penalized, as its trapdoor key is revealed, or identifying the signer in anonymized schemes. **Example of use case:** A user purchases a license to use a software suite on  $dd$  devices. The signature allows the user to sign up to  $dd$  installations. If the user exceeds this limit, their trapdoor key or identity could be revealed, ensuring accountability.
- **Redactable Blockchains:** A blockchain is a linked list of signed elements where each block references the previous one through hash pointers. It functions as a mirrored database, with a consensus protocol ensuring agreement on its content. The integrity of the entire blockchain is verified by checking the hash in the last block, as each block securely links to the one before it.

This structure ensures that blockchain content cannot be modified without recalculating the digests for all subsequent blocks. Most blockchain protocols make this recalculation intentionally impractical to achieve immutability, a desirable property in many applications. However, immutability can sometimes be problematic in specific scenarios, such as:

- **Illegal Content:** An attacker could upload prohibited or harmful content to the blockchain.
- **Storage Efficiency:** As the blockchain grows, the storage requirements may become excessive, making it necessary to compress older blocks.

To address these challenges, replacing standard hash functions with chameleon hash functions allows a trusted authority to redact blocks in the blockchain. This enables the removal of illegal content or compression of outdated blocks while maintaining the integrity and usability of the blockchain.

For redactable blockchains, the previously described property of **full collision resistance** is desirable. This ensures the security of the blockchain remains intact after multiple redactions. Alternatively, the previously presented **enhanced collision resistance** also could be used instead, but in this case redacted blocks must be authenticated with an additional signature scheme.

- **Authentication Protocols:** Chameleon hash functions can be employed as a tool for authentication. Since only the owner of a specific trapdoor key can compute collisions in a chameleon hash scheme, proving the ability to compute a collision serves as evidence of trapdoor ownership. This creates a mechanism for verifying identity or granting access.

However, each authentication event may reveal a collision, which can weaken the security of many chameleon hash constructions over time. Some proposed solutions for this are:

- **Multi-Key Trapdoors:** Chameleon hash schemes with multi-key trapdoors allow collisions to be computed across different keys, provided the user knows both trapdoor keys. This enables the user to generate and update their keys for each authentication, ensuring that no two collisions involve the same key and preventing security degradation.
- **Chameleon Hash Chains:** In this approach, the user authenticates by revealing a preimage generated during the previous authentication, rather than exposing a collision. This avoids directly publicizing collisions and protects the integrity of the trapdoor key.

In certain proposals, the focus is on limiting authentication to a single instance per key. After authenticating once, the user's secret key might be intentionally compromised or revealed. This forces the user to request permission for each new authentication attempt, obtaining a fresh, certified pair of trapdoor keys for future use.

Authentication protocols based on chameleon hash functions are particularly appealing for mobile devices or other low-power systems. Many chameleon hash constructions allow for extremely efficient collision computation, which reduces energy consumption and computational overhead. This makes them ideal for environments with resource constraints, such as IoT devices or lightweight cryptographic applications.

- **Other Applications:** Chameleon hash functions have found use in a variety of cryptographic protocols and systems. Below, we summarize key applications and their relevance:
  - **Fair Exchange Protocols:** By integrating chameleon hash functions into signature schemes and storing the trapdoor key  $tk$  with a trusted server rather than with the signer or recipient, we can create a Fair Exchange Protocol. In such a protocol, two users exchange signed messages after agreeing on the terms and exchange signed messages by a specified deadline. If one party fails to provide their signed message, the other party can request the trusted server to generate the required signature on behalf of the non-compliant user. This approach ensures fairness and accountability in digital exchanges. Relevant works include (YANG et al., 2008) and (SUN et al., 2010).
  - **Searchable Encryption:** Chameleon hash functions can be used to generate search tokens for querying encrypted homomorphic data. This application leverages the **weak uniformity** property of chameleon hashes, ensuring that digests do not leak any information about the hashed message. A detailed exploration of this concept can be found in (YAO et al., 2018).
  - **Algorithmic Tamper-Proof Counter-Units:** In (KOMANO et al., 2009), chameleon hash functions are applied to secure sensitive software memory regions. Specifically:
    1. Memory regions are signed after being hashed using a chameleon hash.
    2. During normal operation, intentional updates (e.g., incrementing a counter) can preserve the signature's validity by computing collisions in the chameleon hash.
    3. Unauthorized modifications, such as those from buffer overflow attacks, would invalidate the signature. In such cases, the system could detect

the tampering and execute countermeasures, such as self-destruction, to prevent sensitive information leaks. This mechanism is referred to as Algorithmic Tamper-Proof Counter-Units.

- **Oblivious Transfer Protocols:** Chameleon hash functions can enhance oblivious transfer protocols, which involve interaction between a user and a server storing  $n$  data items. These protocols ensure that the user can retrieve exactly one of the  $n$  items, and that the server remains unaware of which item was chosen. Relevant works include (BLAZY et al., 2019), (BLAZY; CHEVALIER, 2015), and (BLAZY et al., 2016).

- **Lossy Trapdoor Functions:** Chameleon hash functions have been utilized to construct lossy trapdoor functions. These functions have two indistinguishable modes: a one-to-one mode which behaves like a one-way trapdoor function and a many-to-one mode which behaves like a lossy function that obscures information about the input.

Lossy trapdoor functions are useful in encryption schemes and other cryptographic applications. Examples of works in this area include (LIU, S. et al., 2011), (LAI et al., 2011), (QIN; LIU, S., 2013), (QIN; LIU, S., 2014), (WANG, Z.; YIU, 2015), (NISHIMAKI, 2020), (JIA; LIBERT, 2021), (HOFHEINZ, 2012), (HANSER; SLAMANIG, 2013), and (LIBERT; QIAN, 2019).

- **Chameleon Encryption and Hierarchical Identity-Based Encryption:** Some constructions combine chameleon hash functions with encryption schemes to create chameleon encryption, which introduces a homomorphic relationship between the original message and the digest during encryption and decryption.

This concept has been proposed for hierarchical identity-based encryption, enabling efficient key management and delegation in complex cryptographic systems. Relevant research includes (DÖTTLING; GARG, 2017) and (YOSHIDA et al., 2019).

### 3.5 SUPPLEMENTARY INFORMATION

The raw data containing the list of all works, which were accepted, which were discarded, the reasons for discarding can be checked at the GitHub repository [https://github.com/thiagoharry/mapeamento\\_sistematico](https://github.com/thiagoharry/mapeamento_sistematico). The results are organized in a CSV format together with a list of works and references mentioned in the CSV file in the BibTex format. The CSV format also presents the publication year, how do we classify each paper according chameleon hash scheme, usage and security assumptions.

### 3.6 THREATS TO VALIDITY OF THE SYSTEMATIC MAPPING

The systematic mapping presented in this work was conducted as a preliminary effort to guide the early stages of our research. Its main goal was to provide an initial taxonomy of chameleon hash function applications, properties and constructions, rather than to produce an exhaustive survey. As such, it has several limitations that may affect its completeness and generalizability.

First, the review was conducted in 2021 and was not continuously updated, meaning that more recent publications may not have been considered. Second, the scope of the search was limited to selected journals and flagship conference proceedings in cryptography and security. Broader sources such as Google Scholar or preprint repositories were not included, and we did not apply snowballing techniques to extend the set of references. These decisions were made to keep the effort focused and manageable, in line with the exploratory nature of this review.

As a result, the taxonomy derived from the review may omit relevant constructions or application domains that have emerged more recently or were published in alternative venues. Nonetheless, we believe that the reviewed literature captures the main conceptual distinctions and recurring patterns in the design and use of chameleon hashes, which were sufficient to support the initial development of our proposals.

Improving and expanding this initial mapping remains an open task. As future work, we plan to revisit the literature using a broader range of sources, apply snowballing strategies to identify overlooked studies, and include recent publications to better capture the current state of the art in this area.

### 3.7 DISCUSSION OF RESULTS

The systematic mapping revealed that, while most papers adhere to the classical definition of chameleon hash, the practical usage of this construction is often restricted to specific, narrow scenarios. In many instances, chameleon hashes are employed in mathematical proofs to avoid reliance on the random oracle model and to present security arguments in the standard model. However, beyond these proofs, the collision-finding algorithm is rarely used in practice. Instead, its mere existence functions as a theoretical construct within the proof, without any practical role in the protocol or cryptographic scheme being analyzed.

Excluding this predominant use case in formal proofs, most remaining applications are concentrated in areas such as online/offline signature schemes—where the collision algorithm is invoked but the resulting collisions are never exposed to external parties—or in certain forms of one-time signature constructions. These applications share a common limitation: once a sample collision is leaked, the security of the scheme is compromised, thereby requiring the generation of fresh keys.

The challenges of securely adapting traditional chameleon hash constructions to a broader range of applications are well documented. For instance, in (CAMENISCH et al., 2017), the author initially proposed a new type of sanitizable signature using a traditional chameleon hash as a black-box. However, they later acknowledged that no existing construction met the required security guarantees and ultimately replaced it with a more specialized design based on non-interactive zero-knowledge proofs. Similarly, in (ATENIESE et al., 2017), the authors justified their decision not to adopt a traditional chameleon hash scheme due to the absence of constructions offering both the necessary security properties and proofs in the standard model. Despite numerous efforts, the task of designing a traditional chameleon hash with stronger collision resistance and provable security in the standard model has remained an open problem for years.

While traditional chameleon hash schemes are generally faster and simpler than alternative constructions, their limitations often render them unsuitable for many practical applications. Addressing these limitations—even without solving the difficult problem of achieving stronger forms of collision resistance—became a central focus of our research. Rather than proposing entirely new, non-traditional schemes, we explored how the inherent properties of traditional chameleon hashes could be leveraged to overcome these shortcomings. We also examined how such properties could enhance specific applications or enable new functionalities.

In particular, our work identified a specific property of certain chameleon hash constructions that significantly expands their practical utility: in some schemes, the trapdoor key  $tk$  enables not only the computation of second preimages but also the efficient derivation of first preimages. Although this property is not novel, it has previously been used almost exclusively in the context of mathematical proofs and security reductions. Notably, prior work has shown that chameleon hashes with this feature can transform signature schemes with weaker security guarantees into fully UF-CMA-secure schemes.

Interestingly, there is no universally accepted terminology for this stronger variant of chameleon hash functions or its associated property. In (SHAMIR; TAUMAN, 2001), it is referred to as the inversion property; (SCHMIDT-SAMOA; TAKAGI, 2005) describes it as strong altering; and (LU et al., 2019a) introduces the term chameleon hash plus for constructions that exhibit this capability. For the sake of consistency and clarity, we adopt the term **preimage property** to describe this feature and refer to chameleon hash functions that possess it as **preimage chameleon hash functions**.

We hypothesize that the preimage property can also be harnessed in novel ways to enable the design of more efficient and secure cryptographic protocols—particularly in post-quantum settings. This idea forms the foundation of our work in Chapter 4, where we propose a new approach to redactable blockchain protocols. By leveraging

this property, we aim to overcome key limitations of traditional chameleon hashes regarding the lack of full collision-resistance in the standard model and expand their practical applicability.

### 3.8 CONCLUSION

This chapter has presented a comprehensive systematic mapping study of chameleon hash functions, addressing a significant gap in the literature. Through a rigorous methodological approach examining 353 relevant publications, we have categorized the diverse definitions, security assumptions, and applications of these cryptographic primitives, providing a unified framework for understanding their evolution and potential.

Our analysis reveals two important insights. First, while the traditional chameleon hash remains the most widely studied variant, numerous specialized constructions have emerged to address specific limitations, including identity-based, labelled, secret-coin, and ephemeral-trapdoor variants. Second, the security foundations of these schemes span a broad spectrum of cryptographic assumptions, from classical discrete logarithm and factoring assumptions to post-quantum lattice-based constructions, offering various trade-offs between security guarantees and computational efficiency.

Perhaps most significantly, our mapping has identified that chameleon hash functions, while theoretically versatile, face practical limitations in many proposed applications. Their primary use remains confined to mathematical proofs and limited scenarios where collision revelation is controlled or inconsequential. More complex applications, particularly those requiring resistance to multiple collision exposures, often necessitate specialized constructions or additional cryptographic primitives, highlighting a fundamental tension between theoretical capabilities and practical deployment.

The preimage property, the ability of certain chameleon hash constructions to compute first preimages efficiently with knowledge of the trapdoor key, presented in Section 3.4.3, emerges as a particularly promising avenue for expanding their practical utility. This property, variously termed “inversion property,” “strong altering,” or “chameleon hash plus” in the literature, has been primarily exploited in mathematical proofs but offers untapped potential for novel cryptographic protocols, especially in post-quantum settings.

This systematic review establishes the foundation for our subsequent contributions by identifying key research opportunities: (1) leveraging the preimage property to address limitations in traditional chameleon hash applications, particularly for redactable blockchains; and (2) exploring how alternative cryptographic primitives might overcome similar challenges in post-quantum protocol transitions. The insights gained from analyzing chameleon hash functions, particularly regarding collision resistance, key management, and security in dynamic environments, will inform our approach to

both blockchain redactability and TLS protocol evolution in the following chapters.

By mapping the landscape of chameleon hash research, we not only contribute a valuable resource to the cryptographic community but also establish a clear context for the novel constructions and applications proposed in this thesis. The patterns and limitations identified will guide our design decisions as we develop practical solutions for post-quantum cryptographic adaptation in legacy systems.

## 4 A NEW PROTOCOL FOR REDACTABLE BLOCKCHAINS

### Executive Summary

This chapter introduces a novel redactable blockchain protocol that maintains cryptographic integrity while enabling authorized content modifications. Our approach leverages the preimage property of chameleon hash functions to solve the fundamental tension between blockchain immutability and necessary mutability.

#### Key Contributions:

- A redaction mechanism that prevents collision exposure, maintaining security even after multiple modifications;
- Transparent block structure that makes redactions visible and auditable;
- Both classical and post-quantum implementations with comprehensive security proofs.

**Experimental Results:** Our construction demonstrates superior performance compared to existing approaches, only 13.5-20.5% slower than non-redactable blockchains versus 38%+ penalties in prior designs. The protocol enables regulatory compliance and content moderation applications while preserving blockchain's fundamental integrity guarantees.

### 4.1 INTRODUCTION

This chapter introduces our novel contribution to blockchain technology: a protocol that enables controlled mutability while preserving cryptographic integrity. Traditional blockchains, by design, are immutable ledgers where once data is recorded, it cannot be altered without invalidating subsequent blocks. While this immutability is a cornerstone feature that ensures trust in many applications, it presents significant challenges in scenarios where legitimate modifications may be necessary, such as removing illegal content or implementing data privacy regulations.

Based on our systematic mapping of chameleon hash functions (Chapter 3), we exploit the property of some constructions that allows computing first preimages with the trapdoor to design a more efficient and secure redactable blockchain. Our protocol differs from existing approaches by enabling block redaction without exposing collision information, a common vulnerability in previous designs.

We begin with an overview of redactable blockchain technology and its importance, followed by an analysis of Ateniese's seminal work in this field ((ATENIESE et al., 2017)). We then introduce preimage chameleon hash functions, presenting both classical and post-quantum constructions that serve as building blocks for our protocol. The core of this chapter details our new construction for redactable blockchains, including its design principles, security properties, and operational mechanics.

To validate our approach, we present extensive experimental results comparing our protocol against existing redactable blockchain implementations. These experiments evaluate critical performance metrics including hashing time, verification efficiency, and redaction overhead under various cryptographic assumptions. Our findings demonstrate that the proposed protocol not only enhances security by preventing collision exposure but also achieves superior performance in several key aspects.

The innovations presented in this chapter contribute to addressing one of the fundamental tensions in blockchain technology: maintaining the integrity and trustworthiness of the ledger while enabling necessary modifications under controlled circumstances. Our solution strikes a balance between these competing requirements, offering a practical framework for applications where absolute immutability may be problematic or legally untenable.

## 4.2 INTRODUCTION TO REDACTABLE BLOCKCHAIN TECHNOLOGY

A blockchain is a linked list of signed elements where each block links the previous one using hash pointers. They can act as a mirrored database where some protocol is used to achieve consensus over its content. The application that popularized this concept was Bitcoin, a digital currency designed for decentralized operation proposed by Satoshi Nakamoto in (NAKAMOTO, 2008).

In most digital currencies, blockchains are designed to be immutable. Once a transaction is recorded on the blockchain, it cannot be altered or removed without significant changes to the system. The only practical way to remove a block is through a hard fork, which involves altering the blockchain protocol and requiring participants to agree on invalidating and ignoring certain blocks. However, hard forks are highly disruptive, as they invalidate all subsequent transactions, making them unsuitable for addressing the need to modify or remove an older block.

This immutability, while often a feature, can be problematic in certain contexts. For example, a blockchain can be exploited by malicious users who embed illegal or harmful content within its blocks, potentially creating serious legal and ethical issues. Such scenarios are not merely theoretical; for instance, attacks involving the storage of malware and illicit images, including child abuse material, have been documented on the Bitcoin blockchain (ATENIESE et al., 2017). These incidents can deter users from storing local copies of the blockchain and undermine its broader utility.

A mutable blockchain could provide defenses against such attacks and some additional advantages, like allowing old blocks to be compressed or merged, making the chain smaller. However, such changes should be sporadic and only in exceptional circumstances, following precise rules. Otherwise, this could damage the trust in the chain.

The first paper that proposed a redactable blockchain was (ATENIESE et al.,

2017). The article's idea was to use a chameleon hash instead of a usual hash function in the hash pointer. However, as the only traditional chameleon hash with a sufficiently strong collision-resistance to be used in this context, the author introduced the secret-coin chameleon hash scheme and proceeded to use them instead of a traditional chameleon hash.

After this, a number of different proposals for redactable blockchain building were proposed. Not all of them use chameleon hash functions. According to (ZHANG, D. et al., 2021), besides redactable blockchains based on chameleon hash functions, we also have:

1. Consensus-based mechanisms: When changes in the blockchain are made following some kind of protocol where some participants can vote and decide how eventual redactions are performed. This mechanism is decentralized and, because of this, cannot use a typical chameleon hash function.
2. Meta-transaction-based redaction mechanisms: These mechanisms act on transactions, not in entire blocks. Each transaction can have more than one possible version, but only one of the versions is active at a given time. Special meta-transactions are sent to the chain to add new versions and change the active version of a given transaction. Inactive blocks can be permanently removed or encrypted if one needs to hide them. Their original hash is preserved to ensure the blockchain can be correctly verified even after redaction.
3. Pruning-based redaction mechanisms: This method is used on specific blockchains where the entire transaction history is not expected to be permanently stored. Old blocks can be expired, and they can be replaced by new blocks containing a snapshot with all information from expired blocks that are still relevant.

The different redaction techniques can also be divided between those that redact entire blocks of the blockchain, and those that redact only specific transactions inside the blocks. In addition to content-redaction measures, there are also techniques designed to prevent or hinder the insertion of non-legitimate content into blockchains, thereby aiming to mitigate the problem rather than remedy it after the fact. Some examples of such techniques are listed in (MATZUTT et al., 2018).

The proposal that we will describe in this chapter is a redactable blockchain technology that uses chameleon hash functions and focuses on redacting entire blocks of the blockchain.

### 4.3 ATENIESE'S REDACTABLE BLOCKCHAIN

Here we describe the changes in the Bitcoin backbone protocol, presented in Section 2.3.1, made by Ateniese in (ATENIESE et al., 2017) to create a simple

redactable blockchain using a classical chameleon hash.

Ateniese's blockchain uses the same hash function  $H$ , but replaces the hash function  $G$  with a classical chameleon hash  $CH.Hash$ . A trusted party with the authority to redact the blockchain knows the trapdoor key  $tk$ . The evaluation key  $ek$  is public.

A block is a tuple  $B = (s, x, ctr, rnd)$ , where  $rnd$  is an element from the chameleon hash randomness space in the case of a classical chameleon hash. The other elements are defined as in the Bitcoin Backbone Protocol presented in Section 2.3.1:  $s$  is the digest,  $x$  is the block content, and  $ctr$  is the counter value for the block, used to model timing in the protocol.

A block is valid if:

$$\text{validblock}_q^D(B) := (H(ctr, CH.Hash(ek, (s, x), rnd)) < D) \wedge (ctr < q) = 1$$

It is possible to extend a chain  $C$  with  $Head(c) = (s, x, ctr, rnd)$  attaching a valid block  $B' = (s', x', ctr', rnd')$  such that  $s' = H(ctr, CH.Hash(ek, (s, x), rnd))$ .

The party who knows the chameleon hash trapdoor  $tk$  can redact a block finding a collision in the chameleon hash. Both  $s$  and  $x$  can be redacted, keeping the block and the blockchain valid. This construction can be further customized. We could let only  $x$  to be redacted, replacing the hash computation by  $H(ctr, s, CH.Hash(ek, x, rnd))$ . We could separate redactable and non-redactable parts in a block with the same idea. The chameleon hash processes the redactable parts, and the non-redactable part is processed just by  $H$ .

Each time a redaction is required, the trusted party who knows the chameleon hash trapdoor, after computing the collision, must broadcast a message requiring all the clients to update and redact a given block or sequence of blocks. Not necessarily all clients will comply with the redaction. Some of them could still store old versions of the blocks in their copies of the blockchain. However, the possibility of redaction allows well-behaved users to comply with the law when illegal content is detected in the block and is redacted by the trusted party.

Notice that when this is done, a sample collision is always revealed. Given the old redacted block  $(s, x, ctr, rnd)$  and the new block  $(s, x', ctr, rnd')$ , the pairs  $(x, rnd)$  and  $(x', rnd')$  would produce a collision in the chameleon hash. This means that we would need a chameleon hash with enhanced collision-resistance, or full collision-resistance. For this reason, the author who proposed this construction opted for using a secret-coin chameleon hash scheme, as this property is hard to be achieved with traditional chameleon hash schemes.

#### 4.4 PREIMAGE CHAMELEON HASH

Our main idea to propose a faster redactable blockchain using chameleon hash functions is leveraging the preimage property, described in Section 3.4.3. We can

model chameleon hash schemes with such property as having the following additional algorithm:

- $CH.Preimage(tk, msg, dgt) \rightarrow rnd$ : Given the trapdoor  $tk$ , a message  $msg$  and a digest  $dgt$ , this algorithm returns a randomness  $rnd$  such that  $CH.Hash(ek, msg, rnd) = dgt$ .

Besides the weak collision-resistance, a very important property for our constructions the **strong uniformity**, also described in more details in Section 3.4.3. This property means that there is a distribution  $D_r$  from which we can pick  $rnd \in R$  such that for all pair of keys  $(ek, tk)$  generated by  $CH.KeyGen$  and for all  $msg \in M$ , no adversary can distinguish between  $(rnd, CH.Hash(ek, msg, rnd))$  when  $rnd$  is chosen from distribution  $D_r$  and the tuple  $(CH.Preimage(tk, msg, dgt), dgt)$ , when  $dgt$  is chosen uniformly at random from  $D$ .

Here we present some different constructions of chameleon hash functions with the required properties based on our systematic mapping and literature review.

#### 4.4.1 Classical Construction Based on Factoring and Modular Square Root

We will describe here the chameleon hash proposed by (BELLARE; RISTOV, 2014, Section 4.5). This chameleon hash is of particular interest because its hash can be computed using only multiplications and therefore is fast to compute.

The chameleon hash scheme is defined over sets  $(M, R, D)$  where  $M = \{0, 1\}^k$  and both  $R$  and  $D$  are the set of quadratic residues modulo  $n$ . The value  $k$  is a constant. This scheme requires that all hashed messages have the same length. If necessary, this is achieved first by passing the data to a regular hash function like SHA256 and then passing the output to the chameleon hash function. Doing this, we still have a chameleon hash, as the composition of a regular hash function and a chameleon hash function is also a chameleon hash function.

To generate the pair of keys, a user selects a pair of primes  $(p, q)$  and set  $n = pq$ . It also chooses a vector  $\mathbf{e}$  with  $k$  different prime numbers that are also quadratic residues modulo  $n$ . The trapdoor key is  $tk = (p, q, \mathbf{e})$ . The evaluation key is  $ek = (n, \mathbf{e})$ .

To compute the chameleon hash, following formula is used, recalling that the message  $\mathbf{m}$  is a binary vector:

$$CH.Hash(ek, \mathbf{m}, r) = r^2 \cdot \prod_{0 \leq i < k} \mathbf{e}[i]^{\mathbf{m}[i]} \pmod{n} \quad (1)$$

Computing a hash means multiplying at most  $k$  different numbers with the square of the randomness  $r$ .

Finding the correct  $r \in R$  such that any possible message  $\mathbf{m} \in M$  results in a target digest  $d \in D$  can be done computing:

$$CH.Preimage(tk, \mathbf{m}, d) = \sqrt{\left(\prod_{\mathbf{m}[i]=1} \mathbf{e}[i]\right)^{-1} d} \pmod n \quad (2)$$

Knowing the trapdoor  $(p, q)$  it is possible to compute the modular square root in Equation 2 using, for example, Tonelli–Shanks algorithm (SHANKS, 1973). Without the trapdoor, computing modular square roots like this is a hard problem, provided that the factoring problem is also hard.

**Theorem 1.** *The chameleon hash construction above based on multiplication modulo  $n$  has weak collision resistance, provided that factoring  $n$  is hard.*

*Proof.* To prove this, first, we show that if we could find collisions in that chameleon hash, we could, with non-negligible probability, find the modular square root of a random integer  $x$  modulo  $n$ .

To compute this, we first create a vector  $\mathbf{e}$  with  $k$  primes that are quadratic residues modulo  $n$ . We compute them by generating random numbers, squaring them modulo  $n$ , and then storing the prime results in the vector. By doing this, we will know the modular square root for all primes in  $\mathbf{e}$ , and we could store them in a separate vector. After this, we randomly choose one of the positions in the vector  $\mathbf{e}$  and store  $x$  there, overwriting any previous value. This will let us produce a valid evaluation key  $ek = (n, \mathbf{e})$ .

We pass this evaluation key to the collision-finder and obtain a collision  $(m, r)$ ,  $(m', r')$ . If these pairs are a collision, we can obtain an equation in the form below:

$$\mathbf{e}[a_1] \cdot \mathbf{e}[a_2] \dots \mathbf{e}[a_n] \cdot r^2 \equiv \mathbf{e}[b_1] \cdot \mathbf{e}[b_2] \dots \mathbf{e}[b_m] \cdot r'^2 \pmod n \quad (3)$$

Depending on the bits of the colliding message, one of the values  $\mathbf{e}[i]$  from Equation 3 could be the value  $x$  whose modular square root we want to discover, as we placed it in the evaluation key  $\mathbf{e}$ .

If, for example,  $x$  appears on the left side of Equation 3, but not on the right side, we can obtain its modular square root solving:

$$\sqrt{x} = \sqrt{\frac{\mathbf{e}[a_1] \cdot \mathbf{e}[a_2] \dots \mathbf{e}[a_n] \cdot r^2}{\mathbf{e}[b_1] \cdot \mathbf{e}[b_2] \dots \mathbf{e}[b_m] \cdot r'^2}} \pmod n \quad (4)$$

As we know the square root of all numbers in  $\mathbf{e}$  (except  $x$ ), we can solve the right side of the above equation, and then we obtain the square root of  $x$ . The case when  $x$  is on the right side but not the left side of Equation 3 is analogous.

The probability of having  $x$  on one side but not on the other side of the equation is the probability that  $m$  and  $m'$  have different bits in the position chosen randomly to store  $x$ . If they are part of a valid collision, then necessarily  $m \neq m'$ . Therefore, in the

worst case, both messages differ in at least 1 bit. The two messages have  $k$  bits. Then the probability that we have  $x$  in only one side of the equation is at least  $1/k$ , provided that our collision-finder found a valid collision.

Given access to an oracle that solves a single modular square root modulo  $n$ , it is known that we can factorize  $n$  with probability  $1/2$ . Therefore, being  $\text{Adv}_{CH}^{CR}(\lambda)$  the advantage of a given adversary to find a collision in  $CH$ , the probability that we will factorize  $n$  using this method, and this adversary, represented by  $\text{Adv}_{FACT}$ , is:

$$\text{Adv}_{FACT} \geq \frac{1}{2k} \text{Adv}_{CH}^{CR}(\lambda) \quad (5)$$

Therefore, if we could find collisions in this chameleon hash, then we could factorize  $n$  with non-negligible probability, which contradicts our assumption.  $\square$

**Theorem 2.** *The chameleon hash described in this section has strong uniformity.*

*Proof.* To prove this, it suffices to show that both the set of valid randomness and the set of valid digests have the same number of elements (because they are the same). Therefore, there is a one-to-one correspondence between each possible randomness and possible digest for all messages. The strong uniformity property is achieved if we choose  $\text{rnd} \in R$  from a uniform distribution.  $\square$

In (BELLARE; RISTOV, 2014), Bellare noticed that on practice, computing modular square roots modulo composite numbers appears to be a hard problem not only when the numbers are random: it is also unknown how to compute modular square roots for small primes. Therefore, the performance of this construction can be greatly improved if, instead of filling vector  $\mathbf{e}$  with random primes, we fill it with small primes that are quadratic residues modulo  $n$  at the cost of this stronger assumption.

#### 4.4.2 Post-Quantum Construction Based on Lattices

Lattice-based chameleon hash functions were first proposed in (CASH et al., 2010). The main technique of the construction is using what was presented as “preimage sampleable (trapdoor) functions” (PSF) in (GENTRY et al., 2008). These functions can be implemented in several ways and are denoted by  $f_A : X \rightarrow Y$ , where  $A$  is a public parameter that defines the function. The function has the property that, if given a “small” value sampled from a Gaussian distribution (being small usually means being a vector with a small euclidean norm), the output is statistically close to uniform. This property, among others, is required for the security of most constructions using such functions.

The public parameter  $A$  usually is generated with a trapdoor  $T$ . Using this trapdoor, for any element  $y \in Y$ , it is possible to sample a preimage  $x \in X$  such that  $f_A(x) = y$  and that if  $y$  was chosen from a uniform distribution, then  $x$  is small and produced following a Gaussian distribution. We will denote by  $\text{SAMPLEPRE}$  this algorithm. Computing  $f_A$  preimages without the trapdoor must be infeasible.

Let  $\mathcal{R}_q = \mathbb{Z}_q[x]/(f(x))$  be a polynomial ring with coefficients modulo  $q$  where  $f(x)$  is a polynomial with degree  $n$ . It is possible to build a preimage sampleable function  $f_{\hat{\mathbf{a}}}$  that maps vectors from  $\mathcal{R}_q^m$  to  $\mathcal{R}_q$  computing the following dot product:

$$f_{\hat{\mathbf{a}}}(\hat{\mathbf{x}}) = \hat{\mathbf{a}} \cdot \hat{\mathbf{x}}, \quad (6)$$

where  $\hat{\mathbf{a}}$  is indistinguishable from a vector chosen uniformly at random from  $\mathcal{R}_q^m$ . Producing a suitable  $\hat{\mathbf{a}}$  with a trapdoor can be achieved using the technique from (MICCIANCIO; PEIKERT, 2012) generalized to the ring setting.

The security of this construction can be reduced to the hardness of solving the following problem:

**Definition 1.  $R$ -SIS $_{m,q,\beta}$  Problem:** Given a vector  $\hat{\mathbf{a}} \in \mathcal{R}_q^m$  chosen uniformly at random, find  $\hat{\mathbf{x}} \in \mathcal{R}_q^m$ , such that  $\hat{\mathbf{a}} \cdot \hat{\mathbf{x}} = 0$  and such that  $\|\hat{\mathbf{x}}\| < \beta$ .

Using this ring definition for  $f_{\hat{\mathbf{a}}}$ , we can build a post-quantum chameleon hash based on the one proposed in (CASH et al., 2010). For the key generation, we produce a suitable  $\hat{\mathbf{a}}$  indistinguishable from a random vector from  $\mathcal{R}_q^m$  and a corresponding trapdoor  $T$ . The chameleon hash is computed with the following equation:

$$CH.Hash(\hat{\mathbf{a}}, msg, \hat{\mathbf{r}}) = H(msg) + \hat{\mathbf{a}} \cdot \hat{\mathbf{r}}, \quad (7)$$

where  $H$  is a regular hash function modeled as a random oracle that produces elements from  $\mathcal{R}_q$  and  $\hat{\mathbf{r}}$  is small and sampled from a discrete Gaussian distribution. To link the security of this construction with the  $R$ -SIS $_{m,q,\beta}$  problem, we also require that sampled elements  $\hat{\mathbf{r}}$  are sufficiently small, such that if we add four of them, we would still get an element with norm smaller than  $\beta$  with overwhelming probability.

The algorithm to compute the preimage for this chameleon hash can be defined as below:

$$CH.Preimage(tk, msg, \mathbf{d}) = \text{SAMPLEPRE}((\hat{\mathbf{a}}, T), \mathbf{d} - H(msg)) \quad (8)$$

**Theorem 3.** The chameleon hash construction presented in Equation 7 is collision-resistant in the random oracle model, assuming that the  $R$ -SIS problem is hard for a given set of parameters  $(m, q, \beta)$  shared with the chameleon hash construction and that  $f_{\hat{\mathbf{a}}}(\hat{\mathbf{x}}) = \hat{\mathbf{a}} \cdot \hat{\mathbf{x}}$  is a secure preimage sampleable function.

*Proof.* We show how a collision finder for the presented chameleon hash can be used to create an algorithm to solve  $R$ -SIS $_{m,q,\beta}$  with non-negligible probability.

The algorithm simulates the challenger in the following way.

**Setup:** Our collision-finder algorithm receives the random vector  $\hat{\mathbf{a}}$  and the system parameters  $(m, q, \beta)$ . The algorithm passes the system parameter to the collision-finder and also the evaluation key  $ek = \hat{\mathbf{a}}$ .

**Random Oracle Query:** For each query  $msg_i$ , we sample a small vector  $\hat{\mathbf{r}}_i$  from a Gaussian distribution in  $\mathcal{R}_q$ . Next, we compute  $f_{\hat{\mathbf{a}}}(\hat{\mathbf{r}}_i) = \hat{\mathbf{a}} \cdot \hat{\mathbf{r}}_i$  and send this as a response to the query. By the properties of the preimage sampleable function, given an adequate standard deviation, the result would be indistinguishable from random and uniform.

**Output:** Our collision-finder will produce as output  $((msg', \hat{\mathbf{r}}'), (msg'', \hat{\mathbf{r}}''))$ . Without loss of generality, we will assume that  $msg'$  and  $msg''$  were queried in the random oracle query. Therefore, for known values  $\hat{\mathbf{r}}_i$  and  $\hat{\mathbf{r}}_j$ , we have:

$$\hat{\mathbf{a}}\hat{\mathbf{r}}_i + \hat{\mathbf{a}}\hat{\mathbf{r}}' = \hat{\mathbf{a}}\hat{\mathbf{r}}_j + \hat{\mathbf{a}}\hat{\mathbf{r}}'' \quad (9)$$

And with this equation, we can find a solution to the  $R$ -SIS problem:

$$\hat{\mathbf{a}}(\hat{\mathbf{r}}_i + \hat{\mathbf{r}}' - \hat{\mathbf{r}}_j - \hat{\mathbf{r}}'') = 0 \quad (10)$$

Therefore, if we successfully could compute collisions with roughly the same probability, we could obtain a solution to the  $R$ -SIS problem.

□

For a construction based on  $R$ -SIS that does not require the random oracle model, see the full version of (DUCAS; MICCIANCIO, 2014). Here we opted for this construction because it avoids one additional matrix multiplication, and the applications where we tested the construction already assumed a random oracle model.

We can also use a construction for a preimage sampleable function that maps  $\mathcal{R}_q \times \mathcal{R}_q$  to  $\mathcal{R}_q$ , which is used by the lattice-based Falcon signature:

$$f_{\mathbf{a}}((\mathbf{x}_0, \mathbf{x}_1)) = \mathbf{x}_0 + \mathbf{a}\mathbf{x}_1, \quad (11)$$

where  $\mathbf{a} \in \mathcal{R}_q$ .

This preimage sampleable function uses a different technique to obtain  $\mathbf{a}$  that requires extra cryptographic assumptions besides the hardness of the  $R$ -SIS problem, namely the NTRU assumption. The first chameleon hash using this function was proposed by (LU et al., 2019b, Section 4.2). That chameleon hash equation is:

$$CH.Hash(ek = (\mathbf{a}, \mathbf{h}), \mathbf{m}, (\mathbf{r}_0, \mathbf{r}_1)) = (\mathbf{r}_0 + \mathbf{a}\mathbf{r}_1) + \mathbf{h}\mathbf{m}, \quad (12)$$

where  $\mathbf{h}$  is an element from  $\mathcal{R}_q$  chosen uniformly at random,  $\mathbf{a}$  is indistinguishable from a randomly chosen element from  $\mathcal{R}_q$  and is part of the preimage sampleable function and  $\mathbf{m}$  is an element from  $\mathcal{R}_q$  with binary coefficients.

Like in the previous scheme, the chameleon hash randomness is processed by the preimage sampleable function and, using the corresponding SAMPLEPRE algorithm, it is possible to implement a  $CH.Preimage$  algorithm.

**Theorem 4.** *The chameleon hash construction presented in Equation 12 is collision-resistant, assuming that solving the  $R\text{-}SIS_{3,q,\beta}$  problem is hard.*

*Proof.* Here we restate the proof present in (LU et al., 2019b). If we had an algorithm that can find collisions in that chameleon hash, we could use it to solve the  $R\text{-}SIS_{3,q,\beta}$  problem proceeding as below.

Given an instance  $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$  of the  $R\text{-}SIS_{3,q,\beta}$  problem, first we check if one of these polynomials is invertible. This will be true if all its coefficients are different than zero, which will happen with probability  $(1 - 1/q)^n$ . Let's assume that  $\mathbf{e}_1$  is the invertible polynomial chosen.

Next, we produce a valid chameleon hash evaluation key computing  $\mathbf{a} = \mathbf{e}_2/\mathbf{e}_1$  and  $\mathbf{h} = \mathbf{e}_3/\mathbf{e}_1$ . This produces a key  $ek = (\mathbf{a}, \mathbf{h})$  with uniform distribution.

We pass this key for our collision-finder, that would return as collision the values  $(\mathbf{m}, \mathbf{r}_0, \mathbf{r}_1)$  and  $(\mathbf{m}', \mathbf{r}'_0, \mathbf{r}'_1)$ . If this is a valid collision, by Equation 12, we have:

$$(\mathbf{r}_0 - \mathbf{r}'_0) + \mathbf{a}(\mathbf{r}_1 - \mathbf{r}'_1) + \mathbf{h}(\mathbf{m} - \mathbf{m}') = 0 \quad (13)$$

Which, by the way that we produced the keys, is equal:

$$\mathbf{e}_1/\mathbf{e}_1(\mathbf{r}_0 - \mathbf{r}'_0) + \mathbf{e}_2/\mathbf{e}_1(\mathbf{r}_1 - \mathbf{r}'_1) + \mathbf{e}_3/\mathbf{e}_1(\mathbf{m} - \mathbf{m}') = 0 \quad (14)$$

We can rewrite this as  $1/\mathbf{e}_1$  multiplied by a sum. As this is a multiplication that results in zero, either  $1/\mathbf{e}_1$  or the other multiplicand is zero. As it is impossible for the inverse of a polynomial be zero, this means that the second multiplicand below is zero:

$$\mathbf{e}_1(\mathbf{r}_0 - \mathbf{r}'_0) + \mathbf{e}_2(\mathbf{r}_1 - \mathbf{r}'_1) + \mathbf{e}_3(\mathbf{m} - \mathbf{m}') = 0 \quad (15)$$

Therefore,  $(\mathbf{r}_0 - \mathbf{r}'_0, \mathbf{r}_1 - \mathbf{r}'_1, \mathbf{m} - \mathbf{m}')$  is a solution for the  $R\text{-}SIS_{3,q,\beta}$  problem.  $\square$

**Theorem 5.** *Both post-quantum chameleon hash constructions presented in this Section, based on  $f_{\hat{\mathbf{a}}}$  and  $f_{\mathbf{a}}$  preimage sampleable functions, have the strong uniformity property.*

*Proof.* This is ensured by the properties of the preimage sampleable function and the corresponding SAMPLEPRE algorithm. Provided that the randomness is chosen following a sufficiently narrow Gaussian distribution, the output of  $f_{\hat{\mathbf{a}}}$  or  $f_{\mathbf{a}}$  will be statistically close to uniform over its range. In both cases, the result is added with another element derived from the message. Therefore, the final result also will be statistically close to uniform.  $\square$

#### 4.5 REDACTED BLOCKCHAIN BASED ON PREIMAGE CHAMELEON HASH FUNCTIONS

Using the presented preimage chameleon hash functions, we can now overcome the limited collision-resistance defining the redactable blockchain rules in a way that redactions do not leak collisions. We describe here the full details of our proposal.

A non-redacted block now is a tuple  $(s, x, ctr, rnd)$ . The elements  $s$ ,  $x$ , and  $ctr$  are the same used in a regular blockchain. The element  $rnd$  is a chameleon hash randomness, like in Ateniese's redactable blockchain.

Redacted blocks will be different and distinguishable from non-redacted blocks. A redacted block is a tuple  $(s, x, ctr, rnd', dgt, rnd)$ . The two randomness values  $rnd'$  and  $rnd$  represent, respectively, the new randomness in the redacted block and the old randomness before the block was redacted. The new value  $dgt$  represents the old chameleon hash digest in the block before the redaction. These additional elements will ensure that even after redaction, we still could check the validity of the blocks.

A non-redacted block  $B = (s, x, ctr, rnd)$  is valid if:

$$\mathbf{validblock}_q^D(B) := (H(ctr, rnd, CH.Hash(ek, (s, x), rnd)) < D) \wedge (ctr < q) = 1$$

A redacted block  $B' = (s, x, ctr, rnd', dgt, rnd)$  is valid if:

$$\mathbf{validblock}_q^D(B') := (H(ctr, rnd, dgt) < D) \wedge (ctr < q) = 1$$

The rules to append a new non-redacted block in the blockchain are basically the same. (See Section 4.3.)

Given a block  $B = (s, x, ctr, rnd)$ , to create a redacted block  $B' = (s', x', ctr, rnd', dgt, rnd)$  to replace it, the party with the trapdoor  $tk$  can choose a desired  $s'$  and  $x'$  and should store in  $dgt$  the value  $CH.Hash(ek, (s, x), rnd)$ . Next, it should compute the new  $rnd'$  with  $CH.Preimage(tk, (s', x'), dgt')$ , where  $dgt' = H(ctr, t, dgt)$ . See Figure 8 and 9 for an example.

This way, a redacted block is chained differently than a non-redacted block. As in a regular blockchain, non-redacted blocks are chained, storing the double computed hash in the next block. Meanwhile, redacted blocks are chained, resulting in a single computed hash matching the digest stored in the next block after the block is changed. This can be done because the hash function is a preimage chameleon hash.

$$(s_0, x_0, ctr_0, rnd_0) \leftarrow (s_1, x_1, ctr_1, rnd_1) \leftarrow (s_2, x_2, ctr_2, rnd_2)$$

Figure 8 – Before redacting block  $B_1$ : The next block is correctly chained when  $s_2 = H(ctr_1, rnd_1, CH.Hash(ek, (s_1, x_1), rnd_1))$ .

$$(s_0, x_0, ctr_0, rnd_0) \leftarrow (s'_1, x'_1, ctr_1, rnd'_1, dgt_1, rnd_1) \leftarrow (s_2, x_2, ctr_2, rnd_2)$$

Figure 9 – After redacting block  $B_1$ : The next block is correctly chained when  $s_2 = CH.Hash(ek, (s'_1, x'_1), rnd'_1)$ . The old values  $(dgt_1, rnd_1)$  are still stored in  $B_1$  to allow users to compute the function  $\mathbf{validblock}_q^D$ .

To make a redaction, the authorized party broadcasts the new redacted block to all clients and asks them to change their copy in the blockchain. A proposed redaction

should be rejected if the redacted block is not valid, if the values  $(ctr, rnd_1)$  are different in the redacted and non-redacted block, or if the  $dgt$  stored in the redacted block is not the same value that was computed before the block was redacted.

Ideally, one could also rely on timestamps in each block and establish a consensus rule that any recently mined redacted block must be rejected. In this way, even if the chameleon hash trapdoor were leaked, it could not be exploited to mine new blocks.

A redacted block can easily be identified by the different format and also because it is chained differently. This blockchain construction forbids secret changes in the blockchain: all redactions leave a “scar” in the chain, in the sense that redacted blocks can be easily distinguished from non-redacted ones. This is done to increase the trust in the blockchain and prevent misusing redactions. This is also why we use the chameleon hash randomness as input for  $H$  when checking the block validity. This ensures that the party with a trapdoor cannot make invisible redactions computing collisions instead of preimages.

#### 4.6 SECURITY DISCUSSION

The different chaining in redacted and non-redacted blocks prevents redactions from leaking sample collisions for the chameleon hash, provided that we redact each block with the same key at most once. In this case, a collision would be leaked by a redaction only if one could produce and store in the blockchain values  $(x, rnd)$  such that:

$$H(ctr, t, rnd, CH.Hash(ek, (s, x), rnd)) = CH.Hash(ek, (s, x), rnd)$$

In the Bitcoin backbone protocol,  $H$  is modeled as a random oracle, and therefore, computing this is as hard as performing a preimage attack against the chameleon hash. Likewise, finding a collision during normal usage, without using the trapdoor is guaranteed to be infeasible, provided that the chameleon hash has weak collision-resistance.

However, if a block is redacted twice with the same key, a collision is leaked, and our construction is not secure anymore. To prevent this, we require a trustworthy central authority, which is authorized to make redactions. It should publish a list of trusted keys for the chameleon hash. If it needs to redact twice a block, it could generate a new pair of keys and append the new evaluation key to the list of trusted keys. Likewise, keys also could be revoked, and any redaction made with a revoked key should be ignored.

It can be argued that while we do not reveal collisions, we reveal preimages after each redaction. Thanks to the chameleon hash’s weak collision resistance, the input for  $H$  in each block is always distinct, except with negligible probability. Therefore, when considering  $H$  as a random oracle, the information revealed by each redaction is always a random digest whose distribution is indistinguishable from the uniform and

its respective preimage composed by a specific message (the redacted block) and its randomness  $rnd$ .

To ensure that the chameleon hash is secure under this circumstance, we can define a new kind of collision-resistance where we give the adversary access to an oracle that, given any message  $msg_i$  chosen by the adversary, returns a digest  $dgt$  chosen uniformly at random and the output for  $CH.Preimage(tk, msg_i, dgt)$ .

We present and prove the following theorem which shows that all chameleon hash schemes with weak collision-resistance and strong uniformity properties keep their collision-resistance even in this scenario where the adversary can communicate with such an oracle.

**Theorem 6.** *No efficient adversary can find collisions (except with negligible probability) in a chameleon hash with weak collision-resistance and strong uniformity, even if given access to an oracle that outputs  $CH.Preimage(tk, msg_i, dgt_i)$  for a random digest  $dgt_i$  chosen uniformly at random and a message  $msg_i$  chosen by the adversary.*

*Proof.* Let us prove by contradiction, assuming that  $CH$  is collision-resistant and has strong uniformity, but assuming that we have an efficient adversary  $\mathcal{A}$  that, given access to such oracle, can find a collision in  $CH$ .

We can then build a new adversary  $\mathcal{B}$  that is initialized with an evaluation key  $ek$  and then try to find a collision in  $CH$ .

Adversary  $\mathcal{B}$  works initializing adversary  $\mathcal{A}$  with  $ek$  and then waiting for its queries for the oracle.

For each query  $msg_i$ , adversary  $\mathcal{B}$  chooses a random  $rnd_i$  using the standard distribution from  $CH$  defined by the strong uniformity property. Then, it computes  $dgt_i = CH.Hash(ek, msg_i, rnd_i)$  and send as response for the query the pair  $(rnd_i, dgt_i)$ . Note that by the strong uniformity property, the digest  $dgt_i$  is indistinguishable from a digest chosen uniformly at random and  $rnd_i$  is indistinguishable from a value produced by  $CH.Preimage$ .

After making a polynomial number of queries, adversary  $\mathcal{A}$  outputs a possible collision. Adversary  $\mathcal{B}$  outputs the same values and halts.

Note that if  $\mathcal{A}$  succeeds with a non-negligible probability of finding a collision, then adversary  $\mathcal{B}$  also succeeds with the same probability. Moreover, this contradicts the weak collision-resistance assumption.  $\square$

Also, notice that the digest for  $H$  and for the chameleon hash  $CH$  have the same number of bits. In practice, digests for chameleon hash functions are much larger than digests for SHA256. Therefore, to use our construction, we usually need to use a hash function that allows us to adapt the digest size, like SHAKE256. Depending on the particular hash function and chameleon hash, a conversion function also could be needed to allow us to compare both values.

## 4.7 EXPERIMENTAL RESULTS

To test the viability and performance of our construction, we implemented it using different preimage chameleon hash schemes known in the literature.

Our tests were run in a Dell notebook with a Quad-Core Intel Core i5-3210M 2.50GHz with 4GB of memory and running Ubuntu 20.04.3. While measuring running time, the tests were performed 1000 times, and the mean was extracted. Given the measured standard deviation, we computed the error margin given an interval of confidence of 95%, assuming a normal distribution. We aimed in all constructions for a security level near to 128 bits.

To compare our construction performance, first we measured the time needed to perform known blockchain operations from the literature. The first measured value is the time needed to compute the hash for a regular non-redactable blockchain. For this, we generated a random 1MB block and then measured the time to perform the SHA256 function over the random block, and then over the computed digest. The result is given in the first line of Table 1.

Next, we tested the performance of two existing redactable blockchains proposed in the literature. First, we implemented Ateniese's redactable blockchain described in Section 2.4 using the chameleon hash described in Section 4 from (ATENIESE; MEDEIROS, 2004b) over an elliptic curve group. This construction was chosen because of its simplicity: it is built using a classical chameleon hash instead of requiring the more expensive secret-coin chameleon hash. However, its security proof relies on the non-standard generic group model. We used a 384-bit curve to account for the security impact of leaked collisions during redaction.

The second existing construction that we tested was Ateniese's redactable blockchain built using the secret-coin chameleon hash proposed in (DERLER et al., 2020). That construction was instantiated with the Elgamal signature, as suggested in that article. We choose this construction because it is relatively simple compared with other secret-coin chameleon hash functions. At the same time, it is one of the few with security proof for its full collision-resistance.

For these constructions, we measured the hashing time and the redaction time. For the secret-coin chameleon hash construction, we also measured the verification time. To better present the result, we also put between parenthesis how much slower the hashing time is when compared with the hashing of a simple non-redactable blockchain. We also present in bytes the additional data that must be appended to each block of the blockchain in each construction to allow for redactions. The result is presented in Table 1.

Having these previous values for comparison, we then tested our construction using four different preimage chameleon hash functions. The first construction was the one described in Section 4.4.1 based on the Factoring assumption. We aimed to

Table 1 – Testing existing constructions proposed in the literature. The confidence interval for the timings is 95%.

| Blockchain Type                                                       | Hashing Time (ms)            | Verification Time (ms) | Redaction Time (ms) | Added Data (bytes) |
|-----------------------------------------------------------------------|------------------------------|------------------------|---------------------|--------------------|
| Regular Non-Redactable Blockchain                                     | 3.491 $\pm$ 0.023            | Same than hashing      | -                   | 0                  |
| Ateniese's with chameleon hash from (ATENIESE; MEDEIROS, 2004b)       | 4.849 $\pm$ 0.023 (+38.0%)   | Same than hashing      | 5.488 $\pm$ 0.046   | 64                 |
| Ateniese's with secret-coin chameleon hash from (DERLER et al., 2020) | 11.589 $\pm$ 0.037 (+232.0%) | 10.889 $\pm$ 0.019     | 10.345 $\pm$ 0.045  | 1272               |

achieve 128 bits of security and choose  $k = 256$  and a 2982-bit integer as  $n$  to account for the non-tight security proof.

The second preimage chameleon hash was the one described in the Subsection 4.4.1, but has the optimization described in the last paragraph. We used the additional assumption that it is hard to compute the modular square root of small primes. Other than that, we used exactly the same parameters from the first preimage chameleon hash.

The third preimage chameleon hash was the one described in Subsection 4.4.2 in Equation 7 based solely in the  $R$ - $SIS$  assumption. We built the chameleon hash using the (ROHLOFF et al., 2020) library and using the same parameters presented in (EL BANSARKHANI; BUCHMANN, 2014) to achieve a security level of about 100 bits.

The last tested implementation was the second construction described in Subsection 4.4.2 in Equation 12, using the  $R$ - $SIS$  and NTRU assumption. We used the same implementation as (LU et al., 2019b), with parameters that, according to that article, achieved 114 bits of security against classical attackers and 103 bits of security against quantum attackers.

The results for these tests can be seen in Table 2. We measured the hashing and redaction time and the additional data that need to be stored in each block. Furthermore, as our proposed scheme needs bigger digest sizes, while the other schemes use a SHA256 digest to represent each block, Table 2 also presents the different digest size for each construction. The digest sizes will have the same number of bytes than the chameleon hash digest.

It should be noted that in the last two preimage chameleon hash from Table 2, the size of the additional data appended to each block of the blockchain can be reduced if we run a compression algorithm during the hashing. Therefore, there is a possible trade-off between the hashing time and the size of this data.

The results show that our proposed scheme, when combined with the post-quantum chameleon hash from (LU et al., 2019b) or the classical construction from (BELLARE; RISTOV, 2014), can create a redactable blockchain with better performance

Table 2 – Testing our proposed construction with different preimage chameleon hash functions. The confidence interval for the timings is 95%.

| Preimage Hash                                          | Chameleon | Hashing Time (ms)           | Time | Redaction Time (ms) | Added Data (bytes) | Digest Size (bytes) |
|--------------------------------------------------------|-----------|-----------------------------|------|---------------------|--------------------|---------------------|
| Based on Factoring Assumption                          |           | 5.405 $\pm$ 0.063 (+54.8%)  |      | 16.025 $\pm$ 0.031  | 373                | 373                 |
| Based on Modular Square Root Assumption (small primes) |           | 3.964 $\pm$ 0.005 (+13.5%)  |      | 16.032 $\pm$ 0.021  | 373                | 373                 |
| Based on <i>R-SIS</i> Assumption                       |           | 11.749 $\pm$ 0.043 (+ 236%) |      | 32.208 $\pm$ 0.111  | 128383             | 4191                |
| Based on NTRU and <i>R-SIS</i> Assumption              |           | 4.208 $\pm$ 0.036 (+20.5%)  |      | 4.514 $\pm$ 0.029   | 8192               | 896                 |

than all other tested proposals. They achieve a performance penalty of respectively just 20.5% and 13.5% when compared with a regular non-redactable blockchain.

By comparison, when using Ateniese’s construction with a chameleon hash based on the generic group model from (ATENIESE; MEDEIROS, 2004b), the performance penalty is about 39%. Moreover, our scheme works without needing the generic group model and also can be used with more implementations of chameleon hash functions.

#### 4.8 CONCLUSION

This chapter has presented a novel approach to blockchain redactability that addresses fundamental limitations in existing solutions. By leveraging the preimage property of certain chameleon hash functions, we have demonstrated that it is possible to enable controlled mutability in blockchain systems without compromising their security guarantees or revealing collision information that could potentially undermine the system’s integrity.

Our proposed construction offers several significant advantages over previous redactable blockchain designs. First, by using preimage chameleon hash functions, we enable trustworthy authorities to redact blockchain content without exposing collision information that attackers could exploit. This advancement preserves the collision resistance property of the underlying cryptographic primitives even after multiple redactions, a substantial improvement over traditional approaches that become increasingly vulnerable with each modification.

Second, our approach maintains transparency in the redaction process by creating structurally different blocks that visibly indicate when content has been modified. These “scars” in the blockchain serve as permanent records of redaction events, enhancing accountability while preserving the system’s overall trustworthiness. This transparency stands in contrast to approaches that might allow for surreptitious alterations.

Third, the experimental results demonstrate that our construction offers superior

performance compared to existing redactable blockchain designs. When implemented with optimized chameleon hash constructions, our approach introduces only a modest overhead (13.5% to 20.5%) compared to non-redactable blockchains, significantly better than alternative solutions that incur penalties of 38% or more. This efficiency makes our approach particularly suitable for practical deployment in resource-constrained environments.

Furthermore, we have provided both classical and post-quantum instantiations of our construction, ensuring its viability in future cryptographic landscapes. The post-quantum variant based on lattice assumptions offers security against quantum adversaries while maintaining reasonable performance characteristics, particularly when implemented with NTRU-based preimage chameleon hash functions.

The security analysis confirms that our construction maintains its integrity even when redacted blocks reveal preimages, provided the underlying chameleon hash functions possess the weak collision-resistance and strong uniformity properties. This analysis extends existing security models to account for the unique characteristics of our construction, offering formal assurances of its safety.

While our primary focus has been on redacting entire blocks within a blockchain, the principles and techniques introduced here could be adapted to more granular redaction scenarios, such as modifying specific transactions within blocks. Additionally, the transparent nature of our redaction mechanism could be particularly valuable in regulatory contexts where accountability for content modification is essential.

In conclusion, this chapter addressed a critical tension in blockchain technology between immutability and the occasional necessity for controlled modifications. Our solution provides a cryptographically sound, efficient, and transparent mechanism for enabling trusted authorities to redact problematic content while preserving the overall integrity and trustworthiness of the blockchain system. The methodology demonstrates that with careful cryptographic design, seemingly contradictory requirements, immutability and controlled mutability, can be reconciled without compromising the fundamental security properties that make blockchain technology valuable.

## 5 EASING POST-QUANTUM MIGRATION ON TLS PROTOCOL

### Executive Summary

This chapter presents a novel approach to transitioning the Transport Layer Security (TLS) protocol to post-quantum security without disrupting existing infrastructure. Our solution addresses the critical challenge of migrating millions of certificates and complex trust chains to quantum-resistant algorithms.

#### Key Contributions:

- A hybrid KEMTLS protocol that integrates identity-based encryption with classical cryptography, providing quantum resistance while maintaining compatibility with existing certificates;
- A security framework for combining identity-based and traditional key encapsulation mechanisms with rigorous security proofs;
- A practical migration pathway that eliminates the need for widespread certificate regeneration during initial quantum-resistant deployment.

**Experimental Results:** Our implementation demonstrates minimal performance impact in realistic network environments, only 2.6-7.1% additional latency compared to classical TLS 1.3 before application data transmission. These results validate the practicality of our approach for immediate deployment across the internet's existing security infrastructure.

### 5.1 INTRODUCTION

Transport Layer Security (TLS) constitutes a cornerstone of modern internet security, safeguarding trillions of transactions annually across the global digital infrastructure. As the primary cryptographic protocol protecting sensitive communications, from financial transactions and healthcare data to routine web browsing and machine-to-machine interactions, TLS forms a critical component of our digital trust architecture. This chapter addresses the urgent challenge of transitioning this essential protocol to post-quantum security without disrupting the complex ecosystem that depends on it.

The anticipated development of Cryptographically Relevant Quantum Computers (CRQCs) represents an existential threat to current TLS implementations. These quantum systems could efficiently execute Shor's algorithm, effectively breaking the public-key cryptography that underpins TLS authentication and key exchange mechanisms. This vulnerability necessitates a carefully orchestrated transition strategy to quantum-resistant algorithms before such computing capabilities materialize, a scenario that security experts increasingly view as a matter of "when" rather than "if."

While numerous approaches to post-quantum TLS have been proposed, most solutions require substantial modifications to the existing certificate infrastructure and

Public Key Infrastructure (PKI). These approaches typically demand regeneration of certificate chains, updates to certificate authorities, and widespread deployment of new cryptographic primitives across millions of systems, creating significant technical, logistical, and economic barriers to timely migration. The complexity of this transition threatens to create a dangerous security gap between the emergence of quantum computing capabilities and the widespread deployment of quantum-resistant protocols.

This chapter introduces a novel hybrid solution that addresses these migration challenges through a strategic integration of identity-based encryption with classical cryptography. Our approach enables post-quantum security while maintaining full compatibility with existing certificates and PKI systems, dramatically simplifying the transition process. By eliminating the need for post-quantum certificates in the initial migration phase, we provide a practical pathway for incrementally securing TLS against quantum threats without disrupting the vast ecosystem built upon this critical protocol.

## 5.2 TRANSPORT LAYER SECURITY FUNDAMENTALS

TLS 1.3, specified in RFC 8446 (RESCORLA, 2018), establishes secure communication channels over untrusted networks by creating a shared secret between communicating parties. Unlike protocols that rely solely on symmetric encryption, TLS combines multiple cryptographic techniques to achieve its security properties:

- **Elliptic Curve Diffie-Hellman (ECDH) Key Agreement Protocol:** TLS employs the ECDH key agreement mechanism to establish a shared secret without direct transmission. Through asymmetric cryptography, both parties independently compute this shared secret using only public components exchanged during the handshake process.
- **Digital Signatures:** The protocol uses digital signatures to authenticate the server and, optionally, the client. This authentication relies on public-key cryptography, with Certification Authorities (CAs) issuing digital certificates that validate the authenticity of public keys, thereby establishing trust between communicating parties.

Once the shared secret is established, TLS employs efficient symmetric encryption for bulk data transfer, ensuring both confidentiality and integrity. This hybrid approach, combining asymmetric and symmetric cryptography, creates a robust and efficient security foundation. Figure 1, on page 46 of this thesis, illustrates the TLS 1.3 connection establishment process.

### 5.3 QUANTUM THREAT AND POST-QUANTUM INITIATIVES

Quantum computing poses a fundamental threat to TLS 1.3's security model. With the ability to efficiently execute Shor's algorithm (SHOR, 1994), Cryptographically Relevant Quantum Computers could factor large integers and solve discrete logarithm problems exponentially faster than classical computers. This capability directly compromises the security of widely deployed public-key cryptographic algorithms, including RSA and Elliptic Curve Cryptography (ECC), which form the backbone of TLS authentication and key establishment.

Recognizing this impending threat, the National Institute of Standards and Technology (NIST) initiated a comprehensive post-quantum cryptography standardization program in 2016 (ALAGIC et al., 2020). By 2022, NIST had selected several candidate algorithms for further development and standardization, establishing a framework for post-quantum secure communications.

Adapting TLS 1.3 to resist quantum attacks typically involves replacing the Diffie-Hellman key agreement with a post-quantum Key Encapsulation Mechanism (KEM) and substituting classical signatures with post-quantum alternatives. However, these modifications significantly increase message sizes due to the larger key and signature dimensions required by quantum-resistant algorithms (PAQUIN et al., 2020; SIKERIDIS et al., 2020).

One notable solution addressing these challenges is the Key Encapsulation Mechanism for Transport Layer Security (KEMTLS) protocol (SCHWABE et al., 2020). This innovative approach leverages post-quantum KEMs for both key exchange and authentication, reducing message sizes compared to traditional TLS handshakes and mitigating many performance penalties typically associated with post-quantum cryptography. However, KEMTLS introduces an additional round trip for secret sharing, potentially affecting latency in certain network conditions. Empirical studies comparing KEMTLS to TLS 1.3 with post-quantum algorithms demonstrate that KEMTLS generally requires less memory and exhibits marginally improved performance metrics (GONZALEZ; WIGGERS, 2022; GIRON et al., 2023).

### 5.4 HYBRID APPROACHES AND TRANSITION CHALLENGES

The transition to post-quantum cryptography represents a complex, global-scale migration that cannot occur instantaneously. Industry consensus currently favors hybrid approaches that combine classical and post-quantum algorithms to achieve security levels matching the strongest component. Examples of such deployments include Google Chrome, iMessage, OpenSSH, and AWS. This strategy mitigates the risks posed by early-stage post-quantum algorithms, which require thorough cryptanalysis before full deployment.

The recent cryptanalytic break of SIKE, despite its advanced progress in the NIST standardization process, underscores the necessity for caution and continued evaluation (CASTRYCK; DECRU, 2023; JAO et al., 2021). This event has reinforced support for hybrid constructions; the Open Quantum Safe project explicitly recommends hybrid algorithms during the migration period (OPEN QUANTUM SAFE PROJECT, 2024), and NIST acknowledges their compatibility with emerging standards (BARKER et al., 2018). Recent drafts of proposed TLS standards have formalized proposals for integrating hybrid algorithms (STEBILA et al., 2020).

The TLS ecosystem extends far beyond the protocol itself, encompassing a complex network of certification authorities, certificate validation software, revocation checking mechanisms, root CA trust stores, and diverse client and server applications. Transitioning this ecosystem to post-quantum technologies requires coordinated updates across multiple components, including:

- Certification authorities must implement and support post-quantum algorithms;
- Certificate Revocation Lists (CRLs) require modification;
- Root certificate stores must be updated across all platforms;
- Client and server software must implement new cryptographic primitives.

Several factors compound these challenges. Certification authorities must maintain compatibility with both classical and post-quantum algorithms during the transition period. Furthermore, generating new keys and certificates for millions of TLS servers represents a logistical challenge of remarkable complexity. While major organizations with significant resources may navigate this transition efficiently, smaller entities and resource-constrained Internet-of-Things (IoT) devices face substantial obstacles and potential delays.

Our initial research explored leveraging chameleon hash functions to address these migration challenges. While these functions offer theoretical advantages in discrete logarithm-based constructions, enabling lower-power devices to exchange keys without computationally expensive modular exponentiations, these benefits diminish when transitioning to post-quantum constructions. Furthermore, while chameleon hash functions could potentially enable adaptable certificates (where certificate holders possessing the chameleon hash trapdoor could update their keys without invalidating the certificate), implementing this approach would require substantial modifications to certificate formats and CA infrastructure, making it impractical for general TLS deployment.

## 5.5 OTHER NOTABLE RESEARCH EFFORTS AIMING TO EASE MIGRATION

We found a few previous research efforts addressing post-quantum migration challenges. Notably, Paul et al. (PAUL et al., 2022) highlight the difficulty of regenerating certificate chains to authenticate post-quantum keys. Their approach proposes a gradual migration method where root certificates initially remain unchanged, using hash-based signatures to accommodate different signature algorithms within the same certificate chain. While innovative, this method still requires generating new keys and updating various certificates in the chain, thus preserving substantial migration complexity.

## 5.6 OUR PROPOSED SOLUTION

We introduce a novel hybrid KEMTLS variant that combines post-quantum and classical key encapsulation mechanisms to provide quantum resistance while maintaining compatibility with existing infrastructure. Our approach offers three key advantages:

1. **Existing Certificate Compatibility:** The classical component leverages the existing PKI, while the post-quantum component employs identity-based encryption (IBE), eliminating the need for post-quantum keys in TLS certificates.
2. **Dual-Purpose Key Usage:** Current Elliptic Curve Digital Signature Algorithm (ECDSA) based leaf certificates can serve for both message signing in TLS 1.3 and message encapsulation in our proposed protocol. We provide a rigorous security proof demonstrating that using the same key for both ECDSA signatures and specific Diffie-Hellman-based KEMs maintains security within established models.
3. **Simplified Deployment:** By avoiding the need to regenerate certificates or modify the existing PKI, our solution enables a more rapid transition to post-quantum security.

In our identity-based approach, a server's domain name or another identifier functions as its public key, eliminating the need to transmit or certify post-quantum public keys. Secret keys for individual identities are generated by a Private Key Generator (PKG) and securely delivered to the corresponding servers.

Our proposal complements existing migration strategies, including the approach from Paul et al. (PAUL et al., 2022). While their strategy addresses root certificate updates, our method tackles the challenge of regenerating keys and certificate chains, making it suitable for an earlier phase in the migration process. According to the taxonomy proposed by Alnahawi et al. (ALNAHAWI et al., 2024), we offer a hybrid post-

quantum key exchange solution that specifically targets the key exchange aspect of TLS, deferring authentication components to later migration phases.

## 5.7 ADDRESSING IBE CHALLENGES IN TLS

Our proposal systematically addresses two fundamental challenges associated with integrating identity-based encryption into TLS:

### 5.7.1 Revocation Problem

Using a server's domain name as its identity creates a significant revocation challenge: revoking a server's secret key would effectively revoke the entire domain, which is impractical in most operational scenarios. Traditional solutions include appending timestamps or additional strings to the identity, but these approaches often necessitate additional certification steps.

Our solution integrates the classical public key from the existing certificate into the identity itself, aligning the IBE component with established revocation mechanisms. This innovative approach maintains compatibility with existing certificate revocation processes while enabling post-quantum security.

### 5.7.2 Key Escrow Problem

Identity-based encryption inherently relies on a Private Key Generator to create secret keys for all participating parties, introducing a key escrow risk that is particularly concerning in a widely deployed protocol like TLS. Our hybrid approach mitigates this risk by combining IBE with traditional public key cryptography.

By implementing cryptographic combiners, we ensure that the security of the overall scheme remains at least as strong as the most robust component. Even if the PKG were compromised, the system would maintain security equivalent to classical TLS 1.3, provided the traditional public key algorithms remain secure.

In the worst-case scenario, where a server's IBE key is leaked, the protocol would revert to a classical Diffie-Hellman-based system, maintaining TLS 1.3 security levels while temporarily forfeiting post-quantum resistance. This trade-off, which we consider acceptable, enables a smoother transition to full post-quantum security while preserving backward compatibility. As the ecosystem evolves, the IBE component could be replaced with more conventional post-quantum PKI mechanisms in subsequent migration phases.

## 5.8 OUR HYBRID KEMTLS PROPOSAL

### 5.8.1 The Proposed Hybrid KEM Scheme (HKEM)

By combining public key KEMs with IDKEMs, we can leverage the strengths of both cryptographic primitives. Our hybrid scheme integrates a PKI-based KEM with an identity-based IDKEM using KEM combiners, creating a robust and flexible hybrid KEM. Our hybrid scheme consists of the following algorithms:

- $HKEM.MasterKeyGen(1^\lambda) \rightarrow (mpk, msk)$ : This algorithm generates the master public key  $mpk$  and master secret key  $msk$ , similar to  $IDKEM.KeyGen$ ;
- $HKEM.KeyGen(1^\lambda) \rightarrow (pk, sk)$ : Each user runs this algorithm locally to generate a public-private key pair  $(pk, sk)$ , identical to  $KEM.KeyGen$ ;
- $HKEM.Extract(msk, id) \rightarrow sk_{id}$ : Given the master secret key  $msk$  and an identity  $id$ , this algorithm generates a secret key  $sk_{id}$  corresponding to the identity, similar to  $IDKEM.Extract$ ;
- $HKEM.Encaps(mpk, pk, id) \rightarrow (k, c)$ : This probabilistic polynomial-time algorithm, given the master public key  $mpk$ , a public key  $pk$ , and an identity string  $id$ , produces a secret  $k$  and a ciphertext  $c$  that encapsulates the secret securely;
- $HKEM.Decaps(sk_{id}, sk, c) \rightarrow k$ : This deterministic polynomial-time algorithm, given the secret key associated with identity  $sk_{id}$ , another secret key  $sk$ , and a ciphertext  $c$ , recovers the secret  $k$  encapsulated in the ciphertext.

While the first three algorithms mirror those of the standard KEM and IDKEM schemes, the encapsulation and decapsulation algorithms must be specifically defined for our hybrid KEM. Following the XOR-then-MAC construction from (GIACON et al., 2018), we define them as follows:

---

**Algorithm 22**  $HKEM.Encaps(mpk, pk, id)$

---

- 1:  $(c_1, k_{1a} \parallel k_{1b}) \leftarrow KEM.Encaps(pk)$
  - 2:  $(c_2, k_{2a} \parallel k_{2b}) \leftarrow IDKEM.Encaps(mpk, id)$
  - 3:  $k_{kem} \leftarrow k_{1a} \oplus k_{2a}$
  - 4:  $k_{mac} \leftarrow k_{1b} \parallel k_{2b}$
  - 5:  $c \leftarrow (c_1, c_2)$
  - 6:  $t \leftarrow MAC_{k_{mac}}(c)$
  - 7: **return**  $((c, t), k_{kem})$
- 

If we use a post-quantum identity-based KEM construction like DLP-IBE (DUCAS et al., 2014) and combine it with a classical Diffie-Hellman KEM, we can achieve a post-quantum construction without needing to modify existing keys or certificate chains,

**Algorithm 23**  $HKEM.Decaps(sk_{id}, sk, (c, t))$ 


---

```

1:  $(c_1, c_2) \leftarrow c$ 
2:  $k_{1a} \parallel k_{1b} \leftarrow KEM.Decaps(sk, c_1)$ 
3:  $k_{2a} \parallel k_{2b} \leftarrow IDKEM.Decaps(sk_{id}, c_2)$ 
4:  $k_{kem} \leftarrow k_{1a} \oplus k_{2a}$ 
5:  $k_{mac} \leftarrow k_{1b} \parallel k_{2b}$ 
6: if  $t \neq MAC_{k_{mac}}(c)$  then
7:   return  $\perp$ 
8: else
9:   return  $k_{kem}$ 
10: end if

```

---

provided the key in the leaf certificate is compatible with Diffie-Hellman KEM (i.e., it must be an elliptic curve key rather than an RSA key). KEMTLS can then be used to perform the handshake. Our construction, based on  $HKEM$ , will behave like a typical hybrid KEMTLS construction, with the following differences:

- **No post-quantum key required in the leaf certificate:** We do not need to add a post-quantum key to the leaf certificate. In fact, we can continue using the same certificates in use today, provided they are compatible with DHKEM and we assume that classical signatures remain secure. Compared to a typical hybrid KEMTLS, the data sent during the "ServerHello" message will be smaller due to the reduced size of the certificate chain. Based on the public key sizes of the finalists in NIST's round 3 post-quantum standardization process, this would save approximately 672 to 800 bytes (NIST security level 1) or 930 to 1,134 bytes (NIST security level 3) (ALAGIC et al., 2022).
- **Ciphertext size in the "ClientKeyExchange" message:** According to theoretical predictions from (BANERJEE; CHANDRAKASAN, 2020), the "ClientKeyExchange" message will need to send  $N(2\lceil \log_2 q \rceil - l) + mlen + hlen$  bits as ciphertext produced by the post-quantum part of  $HKEM$ . In this case,  $N$  is the number of dimensions of the underlying lattice, which must be at least 1024 to ensure 128 bits of security. The variable  $l$  is the compression parameter, which increases the probability of decryption failure only negligibly. According to (DUCAS et al., 2014),  $l$  should satisfy  $l \leq \lceil \log_2 q \rceil - 3$ , where  $q$  is the modulus in the Ring-LWE setting. Suggested values for  $q$  range from 23 bits (BANERJEE; CHANDRAKASAN, 2020) to 28 bits (GÜNEYSU; ODER, 2017). The variable  $mlen$  is the size of the internal message encrypted by DLP-IBE, and  $hlen$  is the digest size for the hash used in the Fujisaki-Okamoto transform to achieve IND-CCA security. Both  $mlen$  and  $hlen$  are typically 1280 (for 128-bit security) or 1408 (for 192-bit security).

Thus, the ciphertext in our proposal is expected to be between 3,488 and 4,144 bytes, depending on the parameters suggested in the literature. By comparison,

the ciphertext sizes for the NIST finalists range from 699 to 768 bytes (NIST security level 1) or 930 to 1,088 bytes (NIST security level 3). This indicates that, unfortunately, using known realistic parameters, our proposal would produce larger ciphertexts compared to other KEMs being standardized by NIST. The increase in the size of the "ClientKeyExchange" message would outweigh the savings in the "ServerHello" message.

- **Providing the PKG information:** If the client does not already know which Private Key Generator (PKG) the server uses or its master public key, this information must be provided by the server. The server could send this information either within the leaf certificate or, if the existing certificate chains cannot be modified, as an additional signed message after the certificate chain. The size of the *mpk* is  $N(\log_2 q)$  bits, which would add between 2,944 and 3,584 bytes to the handshake, based on current proposals for  $N$  and  $q$ . If the *mpk* is sent outside the leaf certificate, an additional ECDSA signature would be required for verification using the key in the leaf certificate.

### 5.8.2 Security Modelling for the Hybrid Construction

Typically, combiners mix two constructions of the same cryptographic primitive to create a new construction of the same type. Our approach differs: we combine two related but distinct primitives – public-key KEM and identity-based KEM – to create a new primitive that is neither a pure public-key KEM nor an identity-based KEM. Consequently, we require a new security definition for this hybrid primitive.

There are notable differences between the classical security model for PKI-based KEMs (Subsection 2.2.4) and identity-based KEMs (Subsection 2.2.5). Classical KEM security ensures that no adversary can break the security of a new key. In contrast, identity-based KEM security ensures that no adversary can break the security of any identity chosen by the adversary. The classical model operates in a single-user setting, while the identity-based model involves a multi-user setting. Strong security for public-key cryptography implies that schemes secure in the single-user setting are also secure in the multi-user setting (BELLARE et al., 2000). If an adversary wants information about different users, they can simulate those users by generating their keys. However, this simulation is not possible in identity-based encryption. Therefore, we must allow the adversary to query oracles to accurately model its real-world capabilities. The same holds for our hybrid scheme, where identity-based KEM security serves as the basis for the hybrid construction.

Our hybrid model introduces several differences compared to traditional identity-based KEMs, which must be reflected in the security model. Each user in our model has an identity as well as a pair of local public and secret keys. Therefore, when modeling

the attack game, we must maintain a dictionary to store each pair of keys along with their associated identities. Whenever an adversary queries information related to a specific identity, the associated keys must be generated and stored if they do not already exist.

Since public keys are not confidential, the adversary should be allowed to query the public key for any identity. Furthermore, the model must permit queries to compromise selected identities, reflecting real-world scenarios where an adversary might corrupt or control other users. After a compromise query, the adversary obtains both the local secret key  $sk$  and the identity secret key  $sk_{id}$  for the selected identity.

This model also adapts decapsulation queries, allowing the adversary to specify the identity for which they seek decapsulation information.

The *HKEM* Attack Game together with Algorithms 25, 26 and 27, illustrates the complete definition of the attack game for our hybrid *KEM*.

---

**HKEM Attack Game  $b$  (where  $b$  is either 0 or 1)**


---

```

1:  $Dict \leftarrow \text{EmptyDictionary}()$ 
2:  $(msk, mpk) \leftarrow \text{HKEM.MasterKeyGen}(1^\lambda)$ 
3:  $(id^*, st) \leftarrow \mathcal{A}_1^{\text{Decaps}(msk, \cdot), \text{Compromise}(msk, \cdot), \text{PublicKey}(\cdot)}(mpk)$ 
4:  $pk^* \leftarrow \text{PublicKey}(id^*)$ 
5:  $(c^*, k_0) \leftarrow \text{HKEM.Encaps}(mpk, pk^*, id^*)$ 
6:  $k_1 \leftarrow \mathcal{R}$  ▷ Randomly chosen key from key space  $K$ 
7:  $b \leftarrow \mathcal{A}_2^{\text{Decaps}(msk, \cdot), \text{Compromise}(msk, \cdot), \text{PublicKey}(\cdot)}(st, c^*, k_b)$ 
8: return  $b$ 

```

---



---

**Algorithm 25 Decapsulation Query for HKEM Attack Game**


---

```

1:  $\text{Decaps}(msk, id, c)$ :
2: if  $Dict[id] \neq \emptyset$  then
3:    $(pk, sk) \leftarrow Dict[id]$ 
4: else
5:    $(pk, sk) \leftarrow \text{HKEM.LocalKeyGen}(1^\lambda)$ 
6:    $Dict[id] \leftarrow (pk, sk)$ 
7: end if
8:  $sk_{id} \leftarrow \text{HKEM.Extract}(msk, id)$ 
9: return  $\text{HKEM.Decaps}(sk_{id}, sk, c)$ 

```

---

The adversary's objective is to distinguish between the two scenarios. The adversary's advantage is denoted as  $\text{Adv}_{\text{HKEM}_{\text{XIM}}}^{\text{IND-CCA}}(\mathcal{A}) = |\Pr[W_0] - \Pr[W_1]|$ , where  $W_0$  is the event in which it returns 1 in *HKEM* Attack Game 0, and  $W_1$  is the event in which they return 1 in *HKEM* Attack Game 1. The adversary cannot use  $id^*$  in a Compromise Query or  $(id^*, c^*)$  in a Decapsulation Query.

We can now present and prove a theorem, demonstrating that our hybrid construction utilizing the XOR-then-MAC combiner is secure under the security model established above.

**Algorithm 26** Compromise Query for HKEM Attack Game

---

```

1: Compromise( $msk, id$ ):
2: if  $Dict[id] \neq \emptyset$  then
3:    $(pk, sk) \leftarrow Dict[id]$ 
4: else
5:    $(pk, sk) \leftarrow HKEM.LocalKeyGen(1^\lambda)$ 
6:    $Dict[id] \leftarrow (pk, sk)$ 
7: end if
8:  $sk_{id} \leftarrow HKEM.Extract(msk, id)$ 
9: return  $(sk, sk_{id})$ 

```

---

**Algorithm 27** Public Key Query for HKEM Attack Game

---

```

1: PublicKey( $msk, id$ ):
2: if  $Dict[id] \neq \emptyset$  then
3:    $(pk, sk) \leftarrow Dict[id]$ 
4: else
5:    $(pk, sk) \leftarrow HKEM.LocalKeyGen(1^\lambda)$ 
6:    $Dict[id] \leftarrow (pk, sk)$ 
7: end if
8: return  $pk$ 

```

---

**Theorem 7.** Let  $HKEM_{XtM}$  be a hybrid key encapsulation mechanism constructed using the XOR-then-MAC combiner, with a message authentication code MAC, a public-key key encapsulation mechanism KEM, and an identity-based key encapsulation mechanism IDKEM. If, for all adversaries  $\mathcal{A}_1$ ,  $\mathcal{A}_2$ , and  $\mathcal{A}_3$ , we have that  $AdvKEM^{IND-CCA}(\mathcal{A}_1)$ ,  $AdvIDKEM^{IND-CCA}(\mathcal{A}_2)$ , and  $AdvMAC^{OT-sEUF}(\mathcal{A}_3)$  are negligible, then for all adversaries  $\mathcal{A}_4$ ,  $Adv_{HKEM_{XtM}}^{IND-CCA}(\mathcal{A}_4)$  is also negligible.

The proof is provided in Appendix A. We emphasize that our security proof follows the same structure and steps as those in (GIACON et al., 2018), with adaptations made to accommodate our slightly different security definition.

### 5.8.3 The Full Proposal and General Security Discussion

By integrating our hybrid KEM into the KEMTLS protocol alongside a post-quantum KEM with ephemeral keys generated by the client, we can create a hybrid KEMTLS that does not require including a post-quantum key in the certificate. However, we still need to indicate which Private Key Generators (PKGs) the server supports, either by extending the certificate or modifying the protocol. The server may also provide a list of supported PKGs, allowing the client to choose one it trusts for encapsulation with the hybrid KEM.

If we select a Diffie-Hellman KEM as the classical component of our hybrid construction, the public key in the certificate will be an elliptic curve point, similar to TLS

1.3. Therefore, in the case of a new certificate, the server can generate a Certificate Signing Request (CSR) using that elliptic curve point as the public key and submit it to any existing certificate authority.

CSRs require proof of knowledge, wherein applicants sign the message using the secret key corresponding to the public key being certified, typically with ECDSA. This ensures that the applicant possesses the matching secret key. However, this process poses a risk to our proposal: unlike TLS 1.3, the keys in our construction are part of a KEM, not a signature scheme. Using the same keys for signing messages and later for secret exchange in a KEM may compromise the security of both the signature scheme and the KEM.

While not all combinations of signature schemes and KEMs can securely share keys, we will demonstrate that using the same pair of keys for both an ECDSA signature and a DHKEM does not compromise the security of either cryptographic primitive. This analysis is challenging due to the lack of strong security guarantees for ECDSA signatures. Recent findings in (HARTMANN; KILTZ, 2023) suggest that proving the security of ECDSA without relying on idealized models, such as the generic group or random oracle models, is infeasible. Nevertheless, various models in the literature reduce the security of ECDSA to solving the discrete logarithm problem or similar problems.

We model the ECDSA signature as in (FERSCH et al., 2016), where only line 5 of the signing algorithm, as described in Subsection 2.2.7.1, is replaced by an idealized function. Our analysis shows that the security proofs for both ECDSA signatures and Diffie-Hellman KEM do not interfere with each other, even when these primitives share the same key.

We arrive at this conclusion by modeling security using two new SIG-KEM Attack Games: Game 0 and Game 1. These combine the Signature Attack Game with the KEM Attack Games, similar to how the two games define IND-CCA security for KEMs. The difference here is that the adversary can now query for signatures, assuming the signature scheme and KEM share the same keys. At the end of the game, the adversary outputs a bit  $b$  and a pair  $(msg^*, sig^*)$ . Let  $W_0$  represent the event where the adversary  $\mathcal{A}$  outputs  $b = 1$  in SIG-KEM Game 0, and  $W_1$  the event where it outputs  $b = 1$  in SIG-KEM Game 1. The advantage of a given adversary in the SIG-KEM game is the greater of  $|W_0 - W_1|$  or the probability of producing a valid forgery for a new message using the ECDSA signature in either game.

**Theorem 8.** *Assuming the computational Diffie-Hellman problem is hard and that we can efficiently verify solutions to the problem, no adversary can achieve a non-negligible advantage in the SIG-KEM Attack Games when we instantiate the signature with ECDSA and the KEM with DHKEM.*

**SIG-KEM Attack Game 0**


---

```

1:  $(sk, pk) \leftarrow \text{KeyGen}(1^\lambda)$   $\triangleright$  Key generation shared between KEM and signature.
2:  $Q \leftarrow \emptyset$   $\triangleright Q$ : set of messages sent to signing queries, initialized empty
3:  $(c^*, k_0) \leftarrow \text{KEM.Encaps}(pk)$ 
4:  $(b, msg^*, sig^*) \leftarrow \mathcal{A}^{\text{KEM.Decaps}(sk, \cdot), \text{SigningQuery}(sk, \cdot)}(pk, c^*, k_0)$ 
5: if  $\text{SIG.Verify}(pk, msg^*, sig^*) = \text{accept}$  and  $msg^* \notin Q$  then
6:   return  $(b, 1)$ 
7: else
8:   return  $(b, 0)$ 
9: end if

```

---

**SIG-KEM Attack Game 1**


---

```

1:  $(sk, pk) \leftarrow \text{KeyGen}(1^\lambda)$ 
2:  $Q \leftarrow \emptyset$ 
3:  $(c^*, -) \leftarrow \text{KEM.Encaps}(pk)$ 
4:  $k_1 \xleftarrow{\$} K$   $\triangleright$  Randomly chosen key from key space  $K$ 
5:  $(b, msg^*, sig^*) \leftarrow \mathcal{A}^{\text{KEM.Decaps}(sk, \cdot), \text{SigningQuery}(sk, \cdot)}(pk, c^*, k_1)$ 
6: if  $\text{SIG.Verify}(pk, msg^*, sig^*) = \text{accept}$  and  $msg^* \notin Q$  then
7:   return  $(b, 1)$ 
8: else
9:   return  $(b, 0)$ 
10: end if

```

---

The proof is provided in Appendix B. Note that, since our proposal involves the possible sharing of keys between these distinct primitives only in the classical part of the hybrid construction, the Diffie-Hellman assumption is acceptable, despite not being post-quantum.

Furthermore, as shown in the proof, information revealed by decapsulation queries does not interfere with the results or probabilities when computing responses for signing queries. Similarly, the signatures produced do not interfere with decapsulation queries, thanks to the use of different random oracles for each query type. The upper bound on the probability of breaking the KEM or the ECDSA signatures when they share the same pair of keys is simply the sum of the probabilities of an adversary breaking either the ECDSA or the KEM.

Additionally, similar proofs can be derived using alternative models for ECDSA security. Some works, such as (BROWN, 2005) and (GROTH; SHOUP, 2022), prove the security of ECDSA signatures in the generic group model. A security proof for key sharing between ECDSA signatures and DHKEM could also be made in that model, but it would require defining the gap Diffie-Hellman assumption in the generic model and extending the oracle to allow adversaries to query whether a triple of group elements forms a Diffie-Hellman triple.

Another model of ECDSA signature security is presented in (FERSCH et al., 2017), where the security proof assumes that each message will not be signed twice

(an assumption compatible with TLS). However, this security reduction is based on a weaker and non-standard assumption compared to the computational Diffie-Hellman problem. Using that model, a similar security proof to the one in the appendix could be derived, as that work simulates ECDSA signatures similarly. Adapting that proof, however, would require both the computational Diffie-Hellman assumption and the weaker intractable semi-logarithm assumption discussed in the article.

The security reduction from (FERSCH et al., 2017) applies not only to ECDSA and DSA signatures but also to Chinese SM2 (CHEN, Z. et al., 2013) and Russian GOST (TECHNICAL REGULATING; METROLOGY, 2012) signatures, suggesting that our proposal could provide equivalent security for these signature schemes as well.

#### 5.8.4 Migrating to Post-Quantum TLS

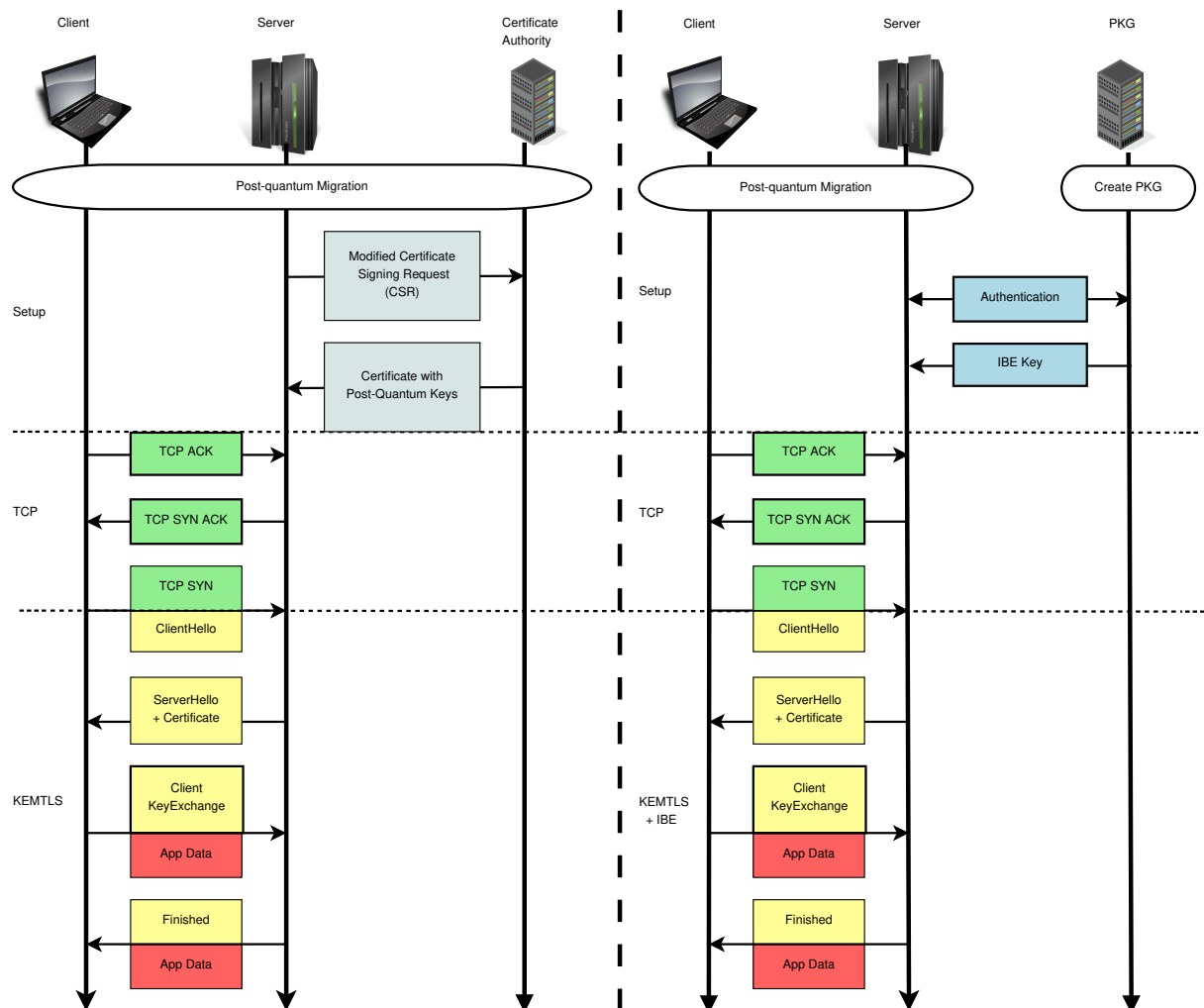


Figure 10 – Side-by-side comparison between KEMTLS (left) and our proposal (right). Our approach does not require changes in the certificate authority or new post-quantum certificates. If new keys are needed, we can use the same CSRs as in TLS 1.3, whereas typical KEMTLS would need to adapt.

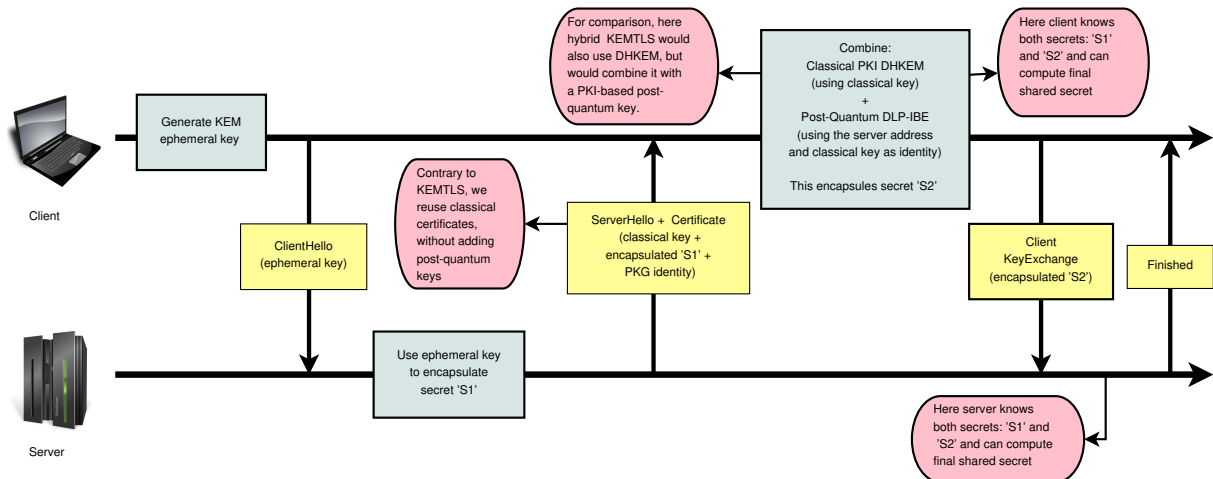


Figure 11 – Detailed view of our proposal mixing KEMTLS and IBE. The yellow boxes represent the messages sent between the client and server over the network. The gray boxes indicate computations performed locally by both the client and server. The red boxes provide additional comments on key aspects of the protocol and emphasize the differences between this proposal and a typical hybrid construction using KEMTLS.

To provide a practical illustration of how our proposal can be used during migration, we compare it with existing proposals like KEMTLS and PQ-TLS. We also discuss some potential use cases in future scenarios.

Migrating existing systems to post-quantum protocols, such as KEMTLS or PQ-TLS, requires updates to the TLS protocol on clients, servers, and CAs. These entities must support the new algorithms. Additionally, certificate authorities must adapt their CSRs, since the current proof of knowledge, where a server proves possession of the secret key associated with the public key, assumes that the certified key belongs to a signature scheme, not a KEM. For post-quantum KEMs, proof of knowledge would need a new interactive protocol or a non-interactive zero-knowledge proof (as proposed for lattice-based KEMs in (GÜNEYSU et al., 2022)).

In contrast, our proposal requires only that the client and server update their protocol; certificate authorities do not need to change, and the same protocols can be used for the Certificate Signing Request. Instead, the protocol now includes the presence of a PKG, which acts as a trustworthy central authority that distributes the secret keys associated with the post-quantum IBE used in our KEM. This distribution must be authenticated.

Additionally, while existing KEMTLS and PQ-TLS proposals require servers to generate new keys and re-certify post-quantum keys, our approach allows the continued use of existing classical keys and certificates.

Figure 10 illustrates the migration process, comparing a KEMTLS protocol migration (left) with our proposal (right). Figure 11 gives more details about how our modified KEMTLS works, stressing the differences with a more typical hybrid version of KEMTLS.

The specific details of the migration process depend on the scenario. Below are some example use cases:

- **IoT devices:** A manufacturer has devices in the market that use TLS to communicate with a server. Through software and firmware updates, the manufacturer can update all involved components, centralizing the PKG responsibility. Devices can receive the PKG's public key via updates, while the server receives its new secret key for the IBE. Since existing certificates need not be changed, the migration can proceed even if certificates are stored in ROM and cannot be updated;
- **Centralized service:** An operating system manufacturer, or a group of manufacturers, offers our protocol as a service that provides automatic post-quantum protection to servers. Server administrators need only subscribe to the service; no new keys need to be generated. Multiple PKGs may exist, but they would be limited in number. The PKG's public key can be identified by an associated number, which is signed with the server's classical key and included in the "ServerHello" message;
- **Decentralized PKGs:** If the migration is not centrally managed, multiple independent PKGs may emerge. Clients and servers can migrate via operating system updates or by installing specific applications. Two trust models are possible: (1) the client can trust the server's identification of the PKG via a signed message, allowing the server to certify the PKG for the client; or (2) the PKG must be certified by an authority trusted by both the client and the server. In this latter case, the certificate would need to be extended to include both the server's and the PKG's identification, which would negate the advantage of using existing certificates.

## 5.9 EXPERIMENTS

To evaluate the performance and feasibility of our proposed hybrid KEMTLS, we modified the KEMTLS implementation from (CELI et al., 2021) to incorporate the hybrid construction detailed in Section 5.8. For the identity-based post-quantum KEM, we adopted the DLP-IBE scheme from (DUCAS et al., 2014), using the implementation provided in (SCOTT, 2023). The classical DHKEM implementation was sourced from the Circl library (CLOUDFLARE, 2024). Additionally, we constructed a hybrid KEM combining Kyber (AVANZI et al., 2019) with DHKEM for ephemeral key encapsulation. The combiner logic and the necessary KEMTLS adaptations were implemented in the Go programming language (THE GO AUTHORS, 2024).

Our proof-of-concept implementation took the form of a modified version of the Go standard library. This modified library can be linked to any Go program that uses TLS, enabling it to use our proposed hybrid KEMTLS. However, the program must explicitly

specify that it wants to use the modified TLS instead of TLS 1.3, and it must also configure the connection with the relevant PKG information. Along with this modification of the TLS protocol, we developed tests and benchmarks integrated into the library's test suite, which were used to perform the performance measurements described below.

Our proposal offers three distinct deployment scenarios based on the level of information exchange between the server and client:

- **KEMTLS-IBE-PDK:** In environments where the client already possesses the necessary server keys, we use KEMTLS with pre-distributed keys (PDK), as described in (SCHWABE et al., 2021). This scenario applies when the client and server have previously established a connection, or in IoT cases where a device always connects to the same server. This configuration enables a single-round handshake without requiring certificate chains or the IBE master public key (*mpk*).
- **KEMTLS-IBE:** When the IBE *mpk* has been disseminated via out-of-band channels, we use the standard KEMTLS protocol.
- **KEMTLS-IBE-MPK:** For cases where the IBE *mpk* must be communicated during the handshake, we introduce an additional signed message following the certificate chain. This message is signed using the P256 key embedded in the leaf certificate via ECDSA.

Our initial experiment aimed to assess the performance impact of transitioning from TLS 1.3 to our proposed solution without altering the underlying TLS infrastructure. To establish a baseline, we constructed a certificate chain comprising a self-signed RSA-4096 root certificate (using SHA-384), an intermediate RSA-2048 certificate (hashed with SHA-256), and a leaf certificate with a 256-bit elliptic curve key (also hashed with SHA-256). This certificate chain configuration mirrors common internet practices, including Google's web pages, and was used for our experiments. For the KEMTLS-IBE-MPK configuration, which requires sending the PKG *mpk* during transmission, we sent it as an additional signed message along with the "ServerHello" message, as described in the last use case in Subsection 5.8.4.

To compare handshake sizes, we measured the total number of bytes exchanged between the client and server, encompassing all messages. Table 3 presents the results. Note that the total handshake size slightly exceeds the sum of individual message sizes due to the inclusion of the final "Finished" message. The same certificate chain, sized at 2050 bytes, was used for all protocols except KEMTLS-IBE-PDK, as protocols with pre-distributed keys do not require certificates.

To evaluate performance under simulated network conditions, we employed Linux Network Emulator (NetEM) (HEMMINGER et al., 2005) on a local machine running Linux (Ubuntu 22.04.4) with an Intel i5 3.10GHz 8-core processor. We measured

Table 3 – Handshake message sizes (in bytes).

| Protocol       | ClientHello | ServerHello | ClientKeyExchange | Total  |
|----------------|-------------|-------------|-------------------|--------|
| TLS 1.3        | 385         | 2,325       | -                 | 2,746  |
| KEMTLS-IBE-PDK | 5,291       | 974         | -                 | 6,301  |
| KEMTLS-IBE     | 1,179       | 2,979       | 4,109             | 8,303  |
| KEMTLS-IBE-MPK | 1,179       | 6,639       | 4,109             | 11,963 |

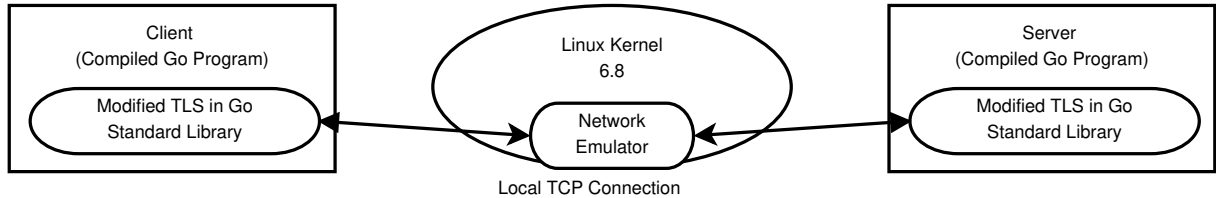


Figure 12 – Setup for our local tests in a simulated environment. Since our tests were integrated into the Go standard library, both the client and server were Go executables, each running a different test function. By modifying a configuration structure in the source code, we were able to use various protocols over the local TCP connection, including TLS 1.3, KEMTLS, and different variations of our proposed KEMTLS-IBE. To simulate a high-bandwidth channel with different latencies, we invoked the “tc qdisc” command with different parameters before running our tests, configuring the network emulator in the Linux kernel.

handshake time in milliseconds by timing the complete TLS handshake process. Network simulations imposed a 1000 Mbit/s bandwidth cap and varied packet delays between 0ms and 150ms. Table 4 summarizes the results, presenting mean and standard deviation values for handshake times across different protocols and latency settings. Additionally, the table includes measurements from a real-world network scenario for comparison. The setup for the tests in simulated environments is shown in Figure 12.

Directly comparing handshake times between KEMTLS and TLS 1.3 is challenging, as KEMTLS involves an additional round-trip. However, both protocols allow application data to be sent after the first round-trip. To address this, we use the “Time-to-send-app-data” (or “SendApp Time”) metric introduced in (SCHWABE et al., 2021), which measures the delay before the client can send application data. While TLS 1.3 and KEMTLS-IBE-PDK show similar times for both the full handshake and the SendApp Time, KEMTLS-IBE and KEMTLS-IBE-MPK demonstrate significantly shorter SendApp Times, as they allow application data to be sent before the handshake is fully completed.

Figure 14 provides a visual representation of the handshake and SendApp time data using violin plots when the protocol is run locally, without introducing delays with NetEM. Figures 15 and 16 show the same metrics with latencies of 1ms and 5ms respectively. As we increase the simulated network delay, Figures 17 and 18 demonstrate the protocol performance with 50ms and 150ms latencies.

To conduct a more realistic evaluation, we deployed a client in London, UK,

and a server in Dallas, USA, on virtual machines hosted by different providers, as shown in Figure 13. While the measured latency between these machines ranged from 40ms to 50ms, we acknowledge that internet connection dynamics may vary. We executed 1000 handshake iterations for each protocol, recording the full handshake time in milliseconds. The results from these real-world tests are included in the final row of Table 4 and illustrated in Figure 19.

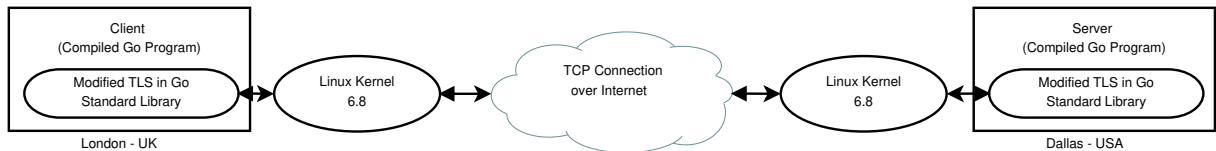


Figure 13 – Setup for remote experiments over a real network. We used the same client and server configuration as in the simulated environment and ran the same tests. However, the client and server were deployed on different rented machines across different continents. Instead of configuring the Linux network emulator, we connected them over a real internet connection using different TLS protocols.

Table 4 – Handshake timings in milliseconds.

| Delay | TLS 1.3        |         | KEMTLS-IBE-PDK |         | KEMTLS-IBE     |         |              |         | KEMTLS-IBE-MPK |         |              |         |
|-------|----------------|---------|----------------|---------|----------------|---------|--------------|---------|----------------|---------|--------------|---------|
|       | Handshake Time |         | Handshake Time |         | Handshake Time |         | SendApp Time |         | Handshake Time |         | SendApp Time |         |
|       | Mean           | St.Dev. | Mean           | St.Dev. | Mean           | St.Dev. | Mean         | St.Dev. | Mean           | St.Dev. | Mean         | St.Dev. |
| 0     | 1.32           | 0.39    | 4.99           | 0.23    | 4.70           | 0.80    | 4.25         | 0.74    | 4.84           | 0.81    | 4.40         | 0.75    |
| 1     | 3.36           | 0.37    | 7.19           | 0.29    | 9.21           | 1.05    | 6.68         | 0.96    | 9.35           | 1.08    | 6.83         | 0.99    |
| 5     | 12.10          | 0.98    | 15.00          | 0.24    | 27.77          | 2.00    | 16.87        | 1.83    | 27.85          | 2.07    | 17.00        | 1.89    |
| 50    | 104.08         | 1.29    | 105.16         | 0.28    | 212.92         | 0.97    | 111.47       | 0.90    | 213.20         | 1.01    | 111.74       | 0.94    |
| 150   | 304.53         | 1.32    | 305.38         | 0.26    | 614.37         | 0.75    | 312.63       | 0.73    | 614.53         | 0.57    | 312.85       | 0.55    |
| Real  | 97.74          | 0.84    | 99.23          | 0.70    | 197.28         | 1.69    | 100.49       | 1.30    | 197.51         | 1.66    | 100.70       | 1.05    |

The data show that using our protocol, clients would need to wait between 2.6% and 7.1% more time before being able to send application data when connecting to a remote server with latencies typical of an Internet connection ( $\geq 50$ ms). The difference is larger in low-latency environments, likely because the slower algorithms used by IBE have a greater impact in such scenarios. Additionally, the data indicate that in high-bandwidth scenarios, sending or not sending an additional master public key (*mpk*) had minimal impact on overall performance.

To further evaluate how our proposal compares with other hybrid KEMTLS constructions, we conducted a second set of tests simulating a future TLS infrastructure that is fully resistant to quantum attacks. In this scenario, secret sharing and authentication are exclusively handled by hybrid KEMTLS schemes, and certificate chains are updated to use hybrid signature algorithms. The leaf certificate may optionally include an IBE master public key (*mpk*) for KEMTLS-IBE-MPK, as described in the last use case in Subsection 5.8.4.

We selected several NIST Level 3 post-quantum KEMs for testing. This security level aligns with the DLP-IBE scheme, which requires a power-of-two value for the

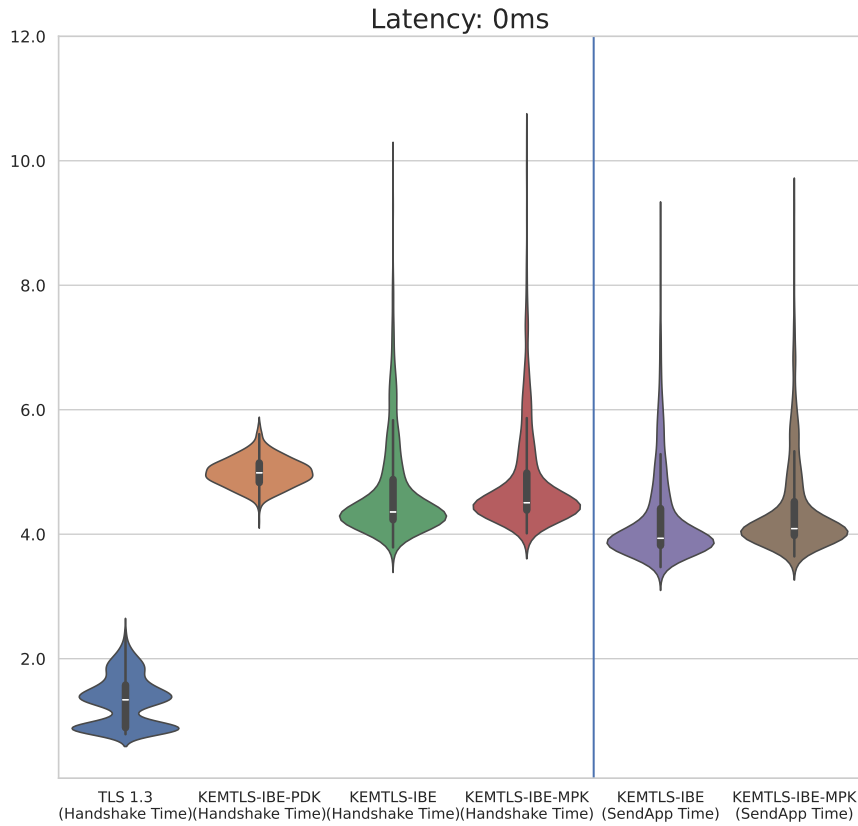


Figure 14 – Comparison of our proposal with TLS 1.3 at 0ms latency. All timings are in milliseconds. Running the protocols over a local connection without introducing any artificial latency using NetEM.

lattice dimension parameter  $N$  (DUCAS et al., 2014). A lattice dimension of 512 offers approximately 80-bit security (below NIST Level 1), while increasing  $N$  to 1024 yields around 192 bits of security, suitable for NIST Level 3. Although some studies, such as (BANERJEE; CHANDRAKASAN, 2020), explore modifications for 128-bit security, these changes have minimal impact on ciphertext size and do not affect the master public key length. Therefore, NIST Level 3 is ideal for leveraging this IBE construction.

To comprehensively assess our proposal’s performance in a fully post-quantum TLS environment, we conducted a second set of experiments. We combined various post-quantum KEMs with classical DHKEM to create different hybrid KEMTLS configurations. Simulating a future post-quantum ecosystem, we incorporated post-quantum certificate chains using a hybrid signature scheme combining ECDSA and the Dilithium-3 post-quantum signature (DUCAS et al., 2018). These certificate chains consist of a root (omitted during transmission), an intermediate, and a leaf certificate. Table 5 summarizes the cumulative handshake message sizes for these configurations. The experimental methodology follows the approach in (GIRON et al., 2023) for comparative

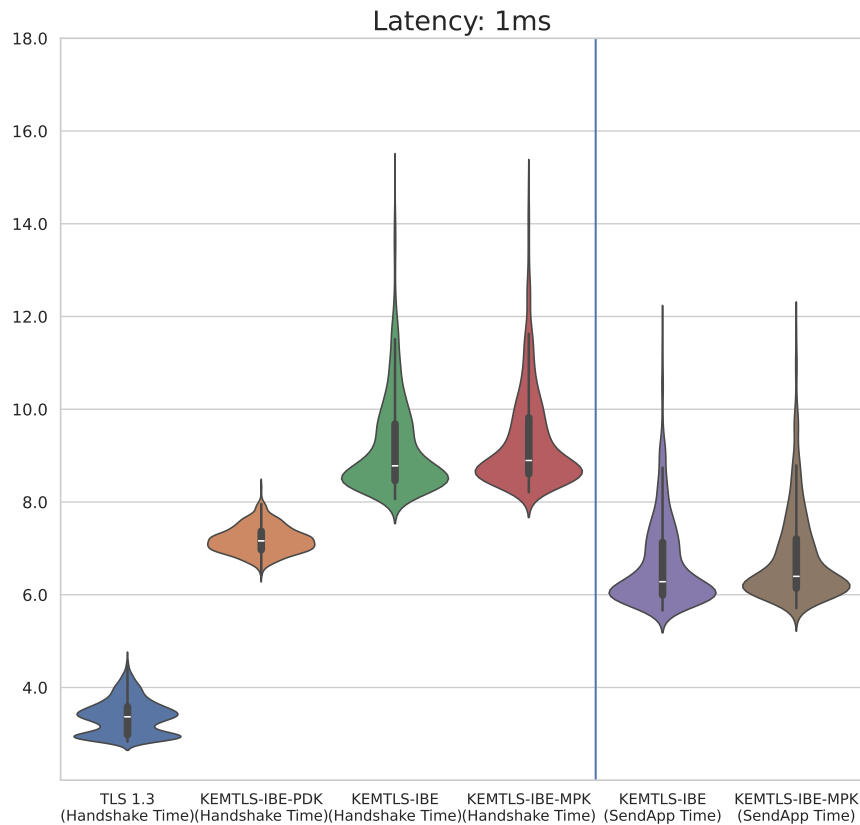


Figure 15 – Comparison of our proposal with TLS 1.3 at 1ms latency. All timings are in milliseconds. Introducing a latency of 1 ms with NetEM.

analysis.

Table 5 – Handshake size (in bytes) comparing our Hybrid KEMTLS construction with traditional Hybrid KEMTLS protocols. All scenarios utilize a hybrid combination of DHKEM and a post-quantum KEM.

| Hybrid Algorithms | ClientHello | ServerHello | ClientKeyExchange | Total  |
|-------------------|-------------|-------------|-------------------|--------|
| DHKEM + Saber     | 1,379       | 11,798      | 1,225             | 14,438 |
| DHKEM + Kyber     | 1,571       | 11,994      | 1,225             | 14,826 |
| DHKEM + NTRU HPS  | 1,317       | 11,581      | 1,067             | 14,001 |
| DHKEM + NTRU-HRSS | 1,525       | 11,996      | 1,239             | 14,832 |
| KEMTLS-IBE        | 1,595       | 10,781      | 4,109             | 16,596 |
| KEMTLS-IBE-MPK    | 1,595       | 14,384      | 4,109             | 20,124 |

Mirroring our previous evaluation, we conducted experiments on a local machine simulating networks with varying latencies and packet loss. Each scenario included 1000 handshake measurements, with mean and standard deviation calculated. Additionally, handshake timings were captured for a client and server connected over the Internet, geographically separated.

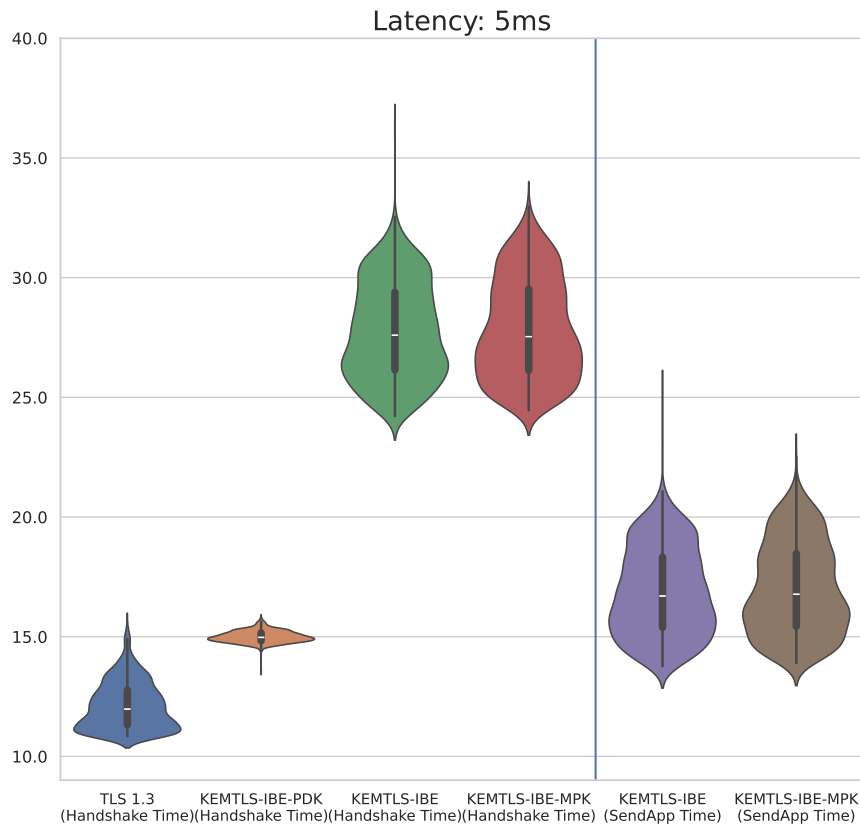


Figure 16 – Comparison of our proposal with TLS 1.3 at 5ms latency. All timings are in milliseconds. Latency of 5 ms introduced with NetEM.

Figures 20 and 21 present violin plots comparing our proposal’s handshake times with other hybrid KEMTLS algorithms under various latency conditions. Violin plots are particularly effective for visualizing the distribution of handshake times, as they combine the features of box plots with kernel density estimates, revealing both central tendencies and the complete distribution shape.

Each subplot corresponds to a specific latency setting. Figure 20 displays lower latency scenarios: the first subplot shows algorithms running locally without artificial delay, while the second and third subplots represent increasing latencies (1ms and 5ms) simulated with NetEm. Figure 21 presents higher latency scenarios (50ms and 150ms) and displays timings for client-server connections over a real Internet connection with geographically distant locations. Detailed numerical data for all configurations is available in Table 6.

A surprising outcome was the shorter handshake times for KEMTLS-IBE compared to other hybrid KEMTLS constructions, despite its larger message sizes. However, these performance differences were minimal, smaller even than variations observed in local, latency-free environments. This suggests that while KEMTLS-IBE may incur some

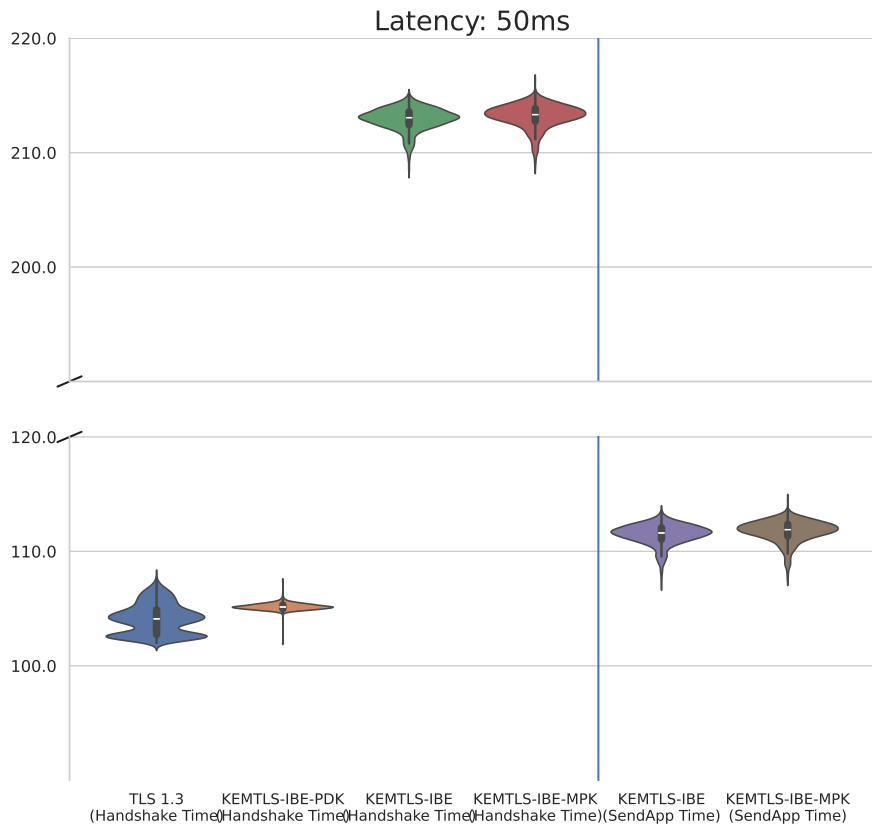


Figure 17 – Comparison of our proposal with TLS 1.3 at 50ms latency. All timings are in milliseconds. Latency of 50 ms introduced with NetEM.

Table 6 – Handshake time (in milliseconds) for various hybrid KEMTLS configurations.

| Latency | DHKEM+Saber |         | DHKEM+Kyber |         | DHKEM+NTRU-HPS |         | DHKEM+NTRU-HRSS |         | KEMTLS-IBE |         | KEMTLS-IBE-MPK |         |
|---------|-------------|---------|-------------|---------|----------------|---------|-----------------|---------|------------|---------|----------------|---------|
|         | Mean        | St.Dev. | Mean        | St.Dev. | Mean           | St.Dev. | Mean            | St.Dev. | Mean       | St.Dev. | Mean           | St.Dev. |
| 0       | 24.43       | 3.60    | 24.39       | 3.47    | 24.60          | 3.65    | 24.55           | 3.65    | 15.47      | 2.48    | 15.18          | 2.45    |
| 1       | 29.76       | 3.47    | 29.68       | 3.06    | 29.73          | 3.47    | 29.68           | 2.81    | 21.46      | 3.68    | 21.56          | 3.50    |
| 5       | 46.17       | 3.14    | 45.72       | 2.82    | 46.14          | 2.96    | 45.94           | 2.89    | 47.70      | 4.38    | 47.67          | 4.25    |
| 50      | 227.24      | 2.76    | 227.24      | 3.06    | 228.06         | 4.39    | 227.39          | 2.88    | 234.61     | 3.53    | 235.06         | 3.39    |
| 150     | 627.14      | 2.38    | 626.96      | 2.43    | 627.41         | 2.65    | 627.22          | 2.48    | 628.82     | 5.32    | 627.38         | 4.34    |
| Real    | 217.99      | 2.85    | 217.96      | 3.11    | 219.00         | 2.81    | 218.38          | 3.10    | 214.32     | 2.58    | 309.44         | 3.18    |

network overhead, its computational efficiency compensates, resulting in faster overall handshake times. Simulation data indicate that increased latency could potentially reverse this trend, favoring other hybrid algorithms.

Another notable result is that, while not observed in simulation, KEMTLS-IBE-MPK performed significantly worse in a real network. We speculate that adding the *mpk* to the certificate crossed a certain threshold, making the message too large to transfer at comparable speeds.

We also evaluated the performance of KEMTLS-IBE-PDK, a variant that uses pre-distributed keys, against other hybrid KEMTLS constructions with pre-distributed keys. Table 7 compares handshake message sizes for these configurations, while Table 8

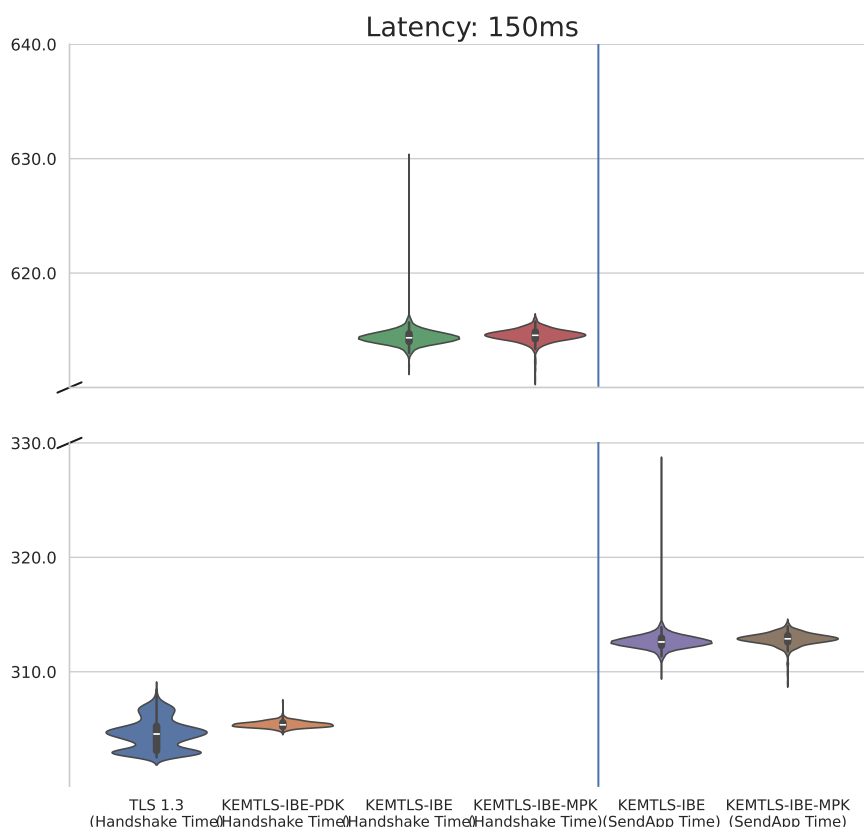


Figure 18 – Comparison of our proposal with TLS 1.3 at 150ms latency. All timings are in milliseconds. Latency of 150 ms introduced with NetEm.

and Figures 22 and 23 compare the timings.

In Figure 22, the first subplot represents algorithms running without artificial delays on the same machine, while the second and third subplots show performance with latencies of 1ms and 5ms simulated using NetEm. Figure 23 presents higher latency scenarios (50ms and 150ms) and displays timings for client-server connections over a real Internet connection with geographically distant locations.

Table 7 – Handshake size (in bytes) comparing our Hybrid KEMTLS construction with traditional Hybrid KEMTLS protocols using pre-distributed keys.

| Hybrid Algorithms       | ClientHello | ServerHello | Total |
|-------------------------|-------------|-------------|-------|
| DHKEM + Saber (PDK)     | 2,601       | 1,331       | 3,963 |
| DHKEM + Kyber (PDK)     | 2,793       | 1,326       | 4,185 |
| DHKEM + NTRU HPS (PDK)  | 2,381       | 1,168       | 3,585 |
| DHKEM + NTRU-HRSS (PDK) | 2,797       | 1,376       | 4,209 |
| KEMTLS-IBE-PDK          | 5,707       | 1,331       | 7,033 |

Despite the larger message sizes, our proposal demonstrated shorter handshake times under certain simulated and real-world network conditions. This unexpected re-

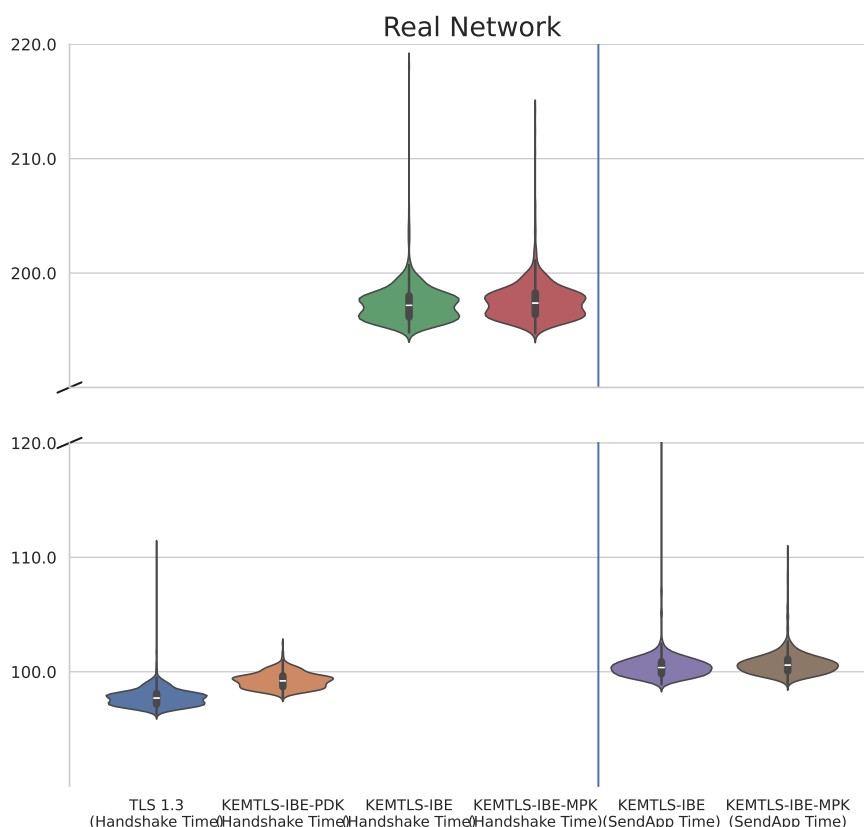


Figure 19 – Comparison of our proposal with TLS 1.3 over a real network. All timings are in milliseconds. Measuring timings over a real internet connection between London, UK and Dallas, USA.

Table 8 – Handshake time in milliseconds when server keys were pre-distributed.

| Latency | Saber (PDK) |         | Kyber (PDK) |         | NTRU-HPS (PDK) |         | NTRU-HRSS (PDK) |         | KEMTLS-IBE-PDK |         |
|---------|-------------|---------|-------------|---------|----------------|---------|-----------------|---------|----------------|---------|
|         | Mean        | St.Dev. | Mean        | St.Dev. | Mean           | St.Dev. | Mean            | St.Dev. | Mean           | St.Dev. |
| 0       | 22.55       | 3.18    | 22.13       | 2.91    | 22.38          | 3.08    | 22.60           | 3.20    | 15.26          | 2.47    |
| 1       | 25.26       | 3.13    | 25.67       | 3.14    | 25.48          | 3.09    | 25.72           | 3.25    | 22.66          | 4.88    |
| 5       | 34.82       | 2.92    | 35.19       | 2.87    | 35.20          | 2.95    | 35.44           | 2.94    | 40.26          | 4.66    |
| 50      | 128.64      | 4.03    | 128.55      | 4.21    | 128.87         | 4.25    | 129.37          | 4.23    | 127.19         | 4.79    |
| 150     | 326.82      | 2.63    | 326.40      | 2.26    | 326.95         | 2.35    | 327.06          | 2.56    | 323.71         | 1.79    |
| Real    | 121.29      | 3.15    | 121.14      | 2.96    | 120.93         | 2.86    | 121.10          | 2.94    | 118.27         | 2.84    |

sult is likely due to the minor impact of network latency compared to the computational efficiency of our implementation. In the pre-distributed key scenario, algorithmic optimizations appear to play a more significant role in determining overall performance. Further research is required to evaluate all algorithms under equivalent optimization levels to establish a more definitive comparison.

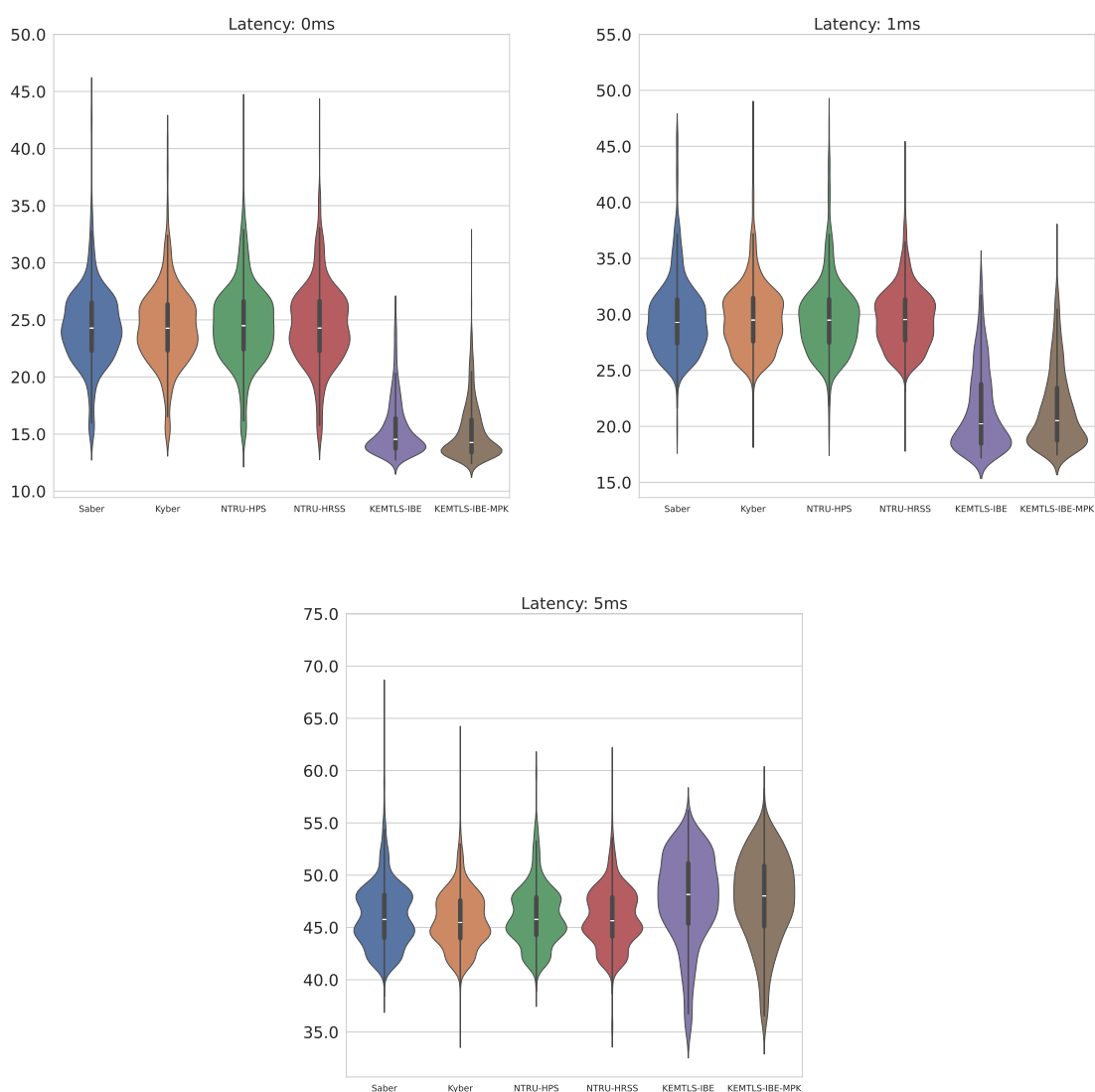


Figure 20 – Comparison of our proposal with other hybrid algorithms (low latency scenarios). All timings are in milliseconds. **(a)** Running the protocols over a local connection without introducing any artificial latency with NetEM. **(b)** Introducing a latency of 1 ms with NetEM. **(c)** Latency of 5 ms introduced with NetEM.

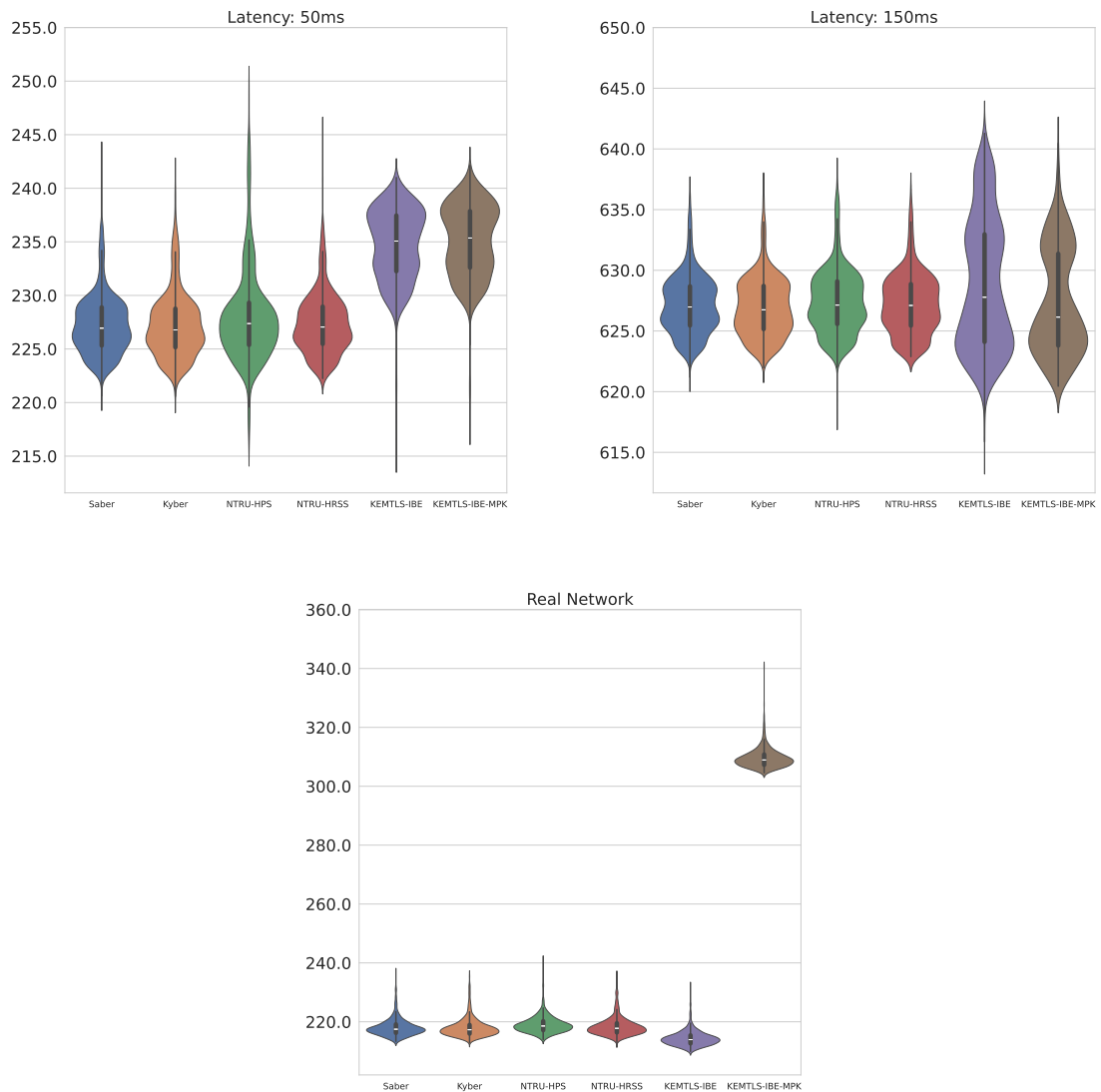


Figure 21 – Comparison of our proposal with other hybrid algorithms (high latency and real network scenarios). All timings are in milliseconds. (a) Latency of 50 ms introduced with NetEM. (b) Latency of 150 ms with NetEM. (c) Measurements over a real Internet connection between geographically distant locations.

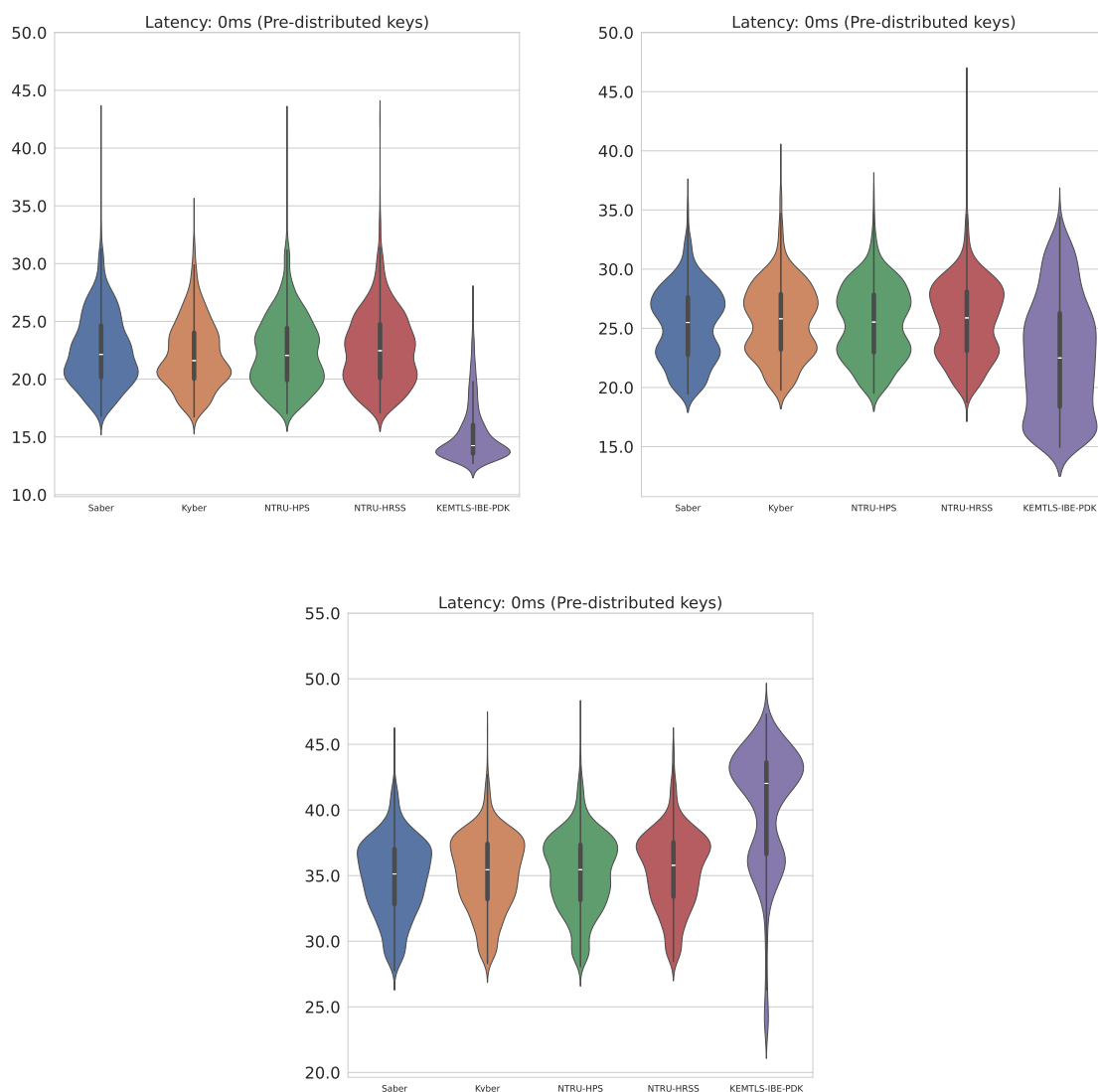


Figure 22 – Comparison of our proposal with other hybrid algorithms using pre-distributed keys (low latency scenarios). All timings are in milliseconds. (a) Running the protocols over a local connection without introducing any artificial latency using NetEM. (b) Introducing a latency of 1 ms with NetEM. (c) Latency of 5 ms introduced with NetEM.

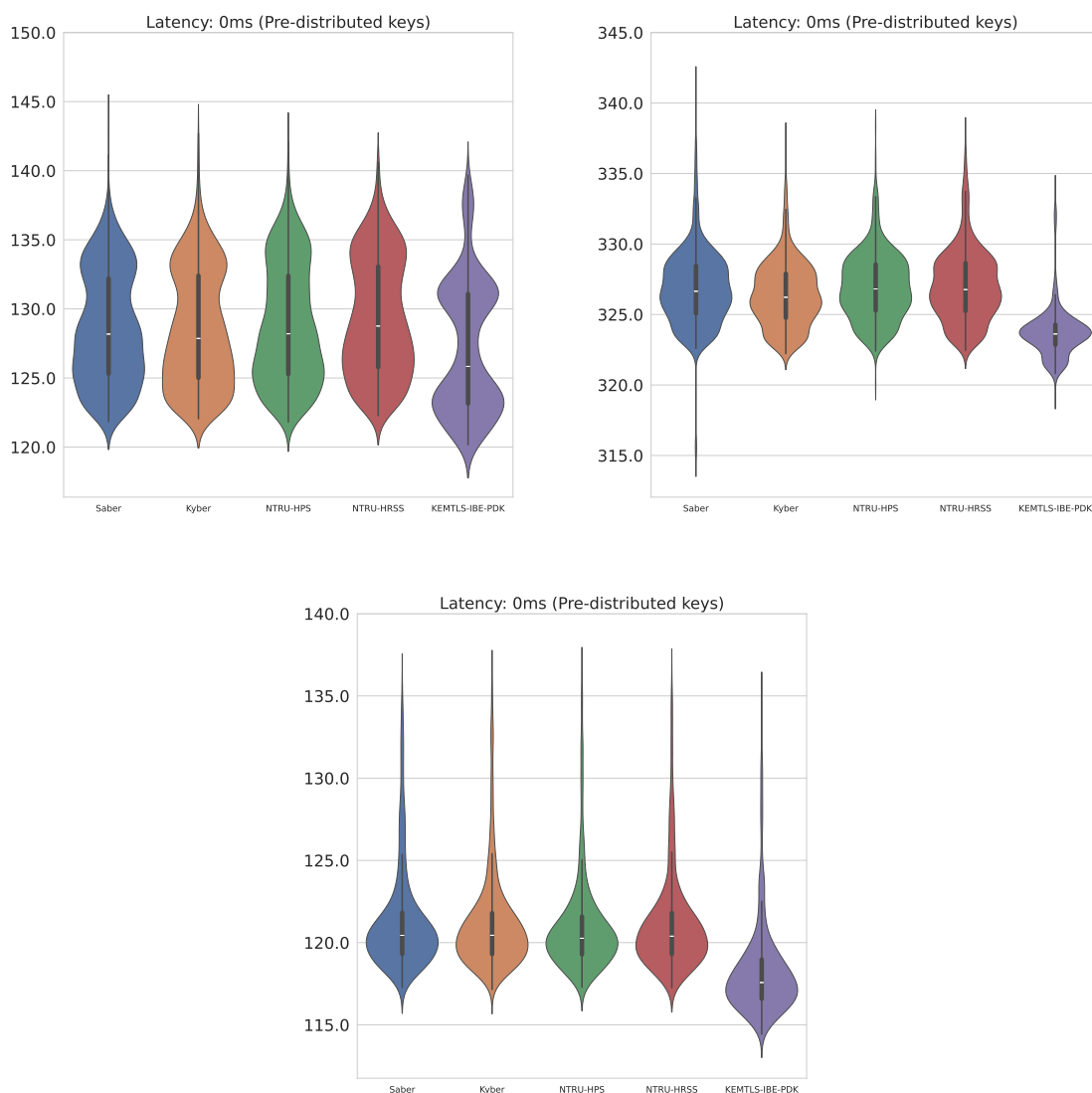


Figure 23 – Comparison of our proposal with other hybrid algorithms using pre-distributed keys (high latency and real network scenarios). All timings are in milliseconds. **(a)** Latency of 50 ms introduced with NetEM. **(b)** Latency of 150 ms with NetEM. **(x)** Measuring timings over a real Internet connection between geographically distant locations.

## 5.10 CONCLUSION

This chapter has presented a novel approach to post-quantum migration for the Transport Layer Security protocol that addresses one of the most significant challenges faced by the cryptographic community: transitioning critical internet infrastructure to quantum-resistant algorithms without disrupting existing deployments. Our proposed hybrid KEMTLS architecture with identity-based encryption offers a pragmatic solution to this complex transition problem, providing a bridge between current classical cryptography and future post-quantum security.

The experimental results demonstrate that our approach maintains reasonable performance characteristics compared to both traditional TLS 1.3 and other proposed post-quantum variants. In typical internet scenarios with latencies of 50ms or more, clients using our protocol experience only a marginal increase of 2.6-7.1% in waiting time before being able to send application data. This modest performance impact represents an acceptable trade-off given the significant migration advantages our solution provides. Notably, our approach's computational efficiency partially compensates for the larger message sizes, particularly in pre-distributed key scenarios where algorithmic optimizations play a more prominent role than network overhead.

A key innovation of our approach is the integration of identity-based encryption, which eliminates the need for post-quantum keys in TLS certificates. This design decision addresses the certificate chain regeneration challenge that has impeded previous migration strategies. By maintaining backward compatibility with existing certificates and PKI infrastructure, our solution allows for a more gradual, phased migration that can be implemented at the protocol level without requiring immediate changes to certificate authorities and their established procedures.

We have provided rigorous security proofs demonstrating that our hybrid construction maintains security even when the same key pair is used for both ECDSA signatures and DHKEM operations. This dual-purpose key usage represents a significant practical advantage, enabling servers to continue using their existing certified keys while gaining post-quantum protection. Furthermore, our XOR-then-MAC combiner ensures that the overall scheme remains secure as long as at least one of its component primitives remains uncompromised, providing defense-in-depth against both classical and quantum adversaries.

The proposed approach also effectively addresses the two fundamental challenges associated with identity-based encryption in TLS: the revocation problem and the key escrow problem. By incorporating the server's classical public key into the identity itself, our solution aligns IBE revocation with existing certificate revocation mechanisms. Similarly, by combining IBE with traditional public key cryptography through a secure KEM combiner, our approach ensures that even in a worst-case scenario where a PKG is compromised, the system would still maintain classical security levels.

Our implementation and evaluation of multiple deployment scenarios, KEMTLS-IBE-PDK for pre-distributed keys, KEMTLS-IBE for out-of-band master public key distribution, and KEMTLS-IBE-MPK for in-band transmission, provide various options that can be selected based on specific operational requirements and constraints. This flexibility makes our solution applicable across diverse environments, from IoT deployments with resource constraints to large-scale web services with frequently changing client connections.

While our experimental results show that message sizes in the proposed protocol are larger than those of other post-quantum KEM solutions being standardized by NIST, this represents a reasonable compromise given the significant migration advantages. Additionally, as the field of identity-based cryptography continues to evolve, more efficient constructions may emerge that could further reduce these overheads.

Looking forward, our approach provides a viable pathway for the initial phase of post-quantum migration, focusing on securing the key exchange aspect of TLS while deferring authentication components to later phases. This incremental strategy aligns with industry best practices for large-scale infrastructure transitions and complements other proposed solutions like the hash-based signature approach for root certificate updates. By enabling a smoother transition toward quantum resistance without disrupting existing systems, our proposal contributes to the broader effort of securing internet communications against emerging quantum threats.

In sum up, this chapter's contributions offer both theoretical advances in cryptographic protocol design and practical solutions to real-world deployment challenges. Our hybrid KEMTLS approach with identity-based encryption represents a significant step toward enabling widespread adoption of post-quantum cryptography in one of the internet's most critical security protocols.

## 6 DISCUSSION ABOUT THE CONTRIBUTIONS AND CONCLUSION

### Executive Summary

This chapter synthesizes the contributions of this thesis to post-quantum cryptographic adaptations for legacy systems and contextualizes their significance within the broader field. The research addresses two fundamental challenges in quantum-resistant protocol transitions: enabling controlled mutability in blockchain systems and facilitating TLS migration without disrupting existing infrastructure.

#### Key Contributions Synthesized:

- Our redactable blockchain protocol leverages the preimage property of chameleon hash functions to enable transparent, secure content modification while maintaining cryptographic integrity, demonstrating up to 24.5% performance improvement over existing approaches;
- Our hybrid TLS approach combines identity-based encryption with classical cryptography to achieve post-quantum security while maintaining compatibility with existing certificates, imposing only 2.6-7.1% additional latency in realistic network conditions;
- Both contributions provide practical migration pathways that balance security requirements with implementation realities, enabling incremental adoption of quantum-resistant technologies.

**Broader Implications:** This research establishes a new paradigm for post-quantum migration strategies that emphasizes backward compatibility, performance optimization, and practical deployment considerations. By reconciling seemingly contradictory requirements, such as immutability versus controlled mutability and quantum resistance versus infrastructure compatibility, our work demonstrates that careful cryptographic design can ease transition challenges without compromising security fundamentals.

### 6.1 INTRODUCTION

This chapter synthesizes the primary contributions of this thesis, contextualizes them within the broader field of post-quantum cryptography, and outlines directions for future research. Our work has addressed two critical challenges in the preparation for quantum-resistant cryptographic protocols: enabling controlled mutability in blockchain systems and simplifying the transition to post-quantum security in TLS communications. These contributions are united by a common theme, developing practical solutions that enhance post-quantum security while minimizing disruption to existing infrastructure and operational processes.

The methodological approach throughout this thesis has balanced theoretical rigor with practical implementation considerations. For each proposed solution, we

have provided formal security analyses, mathematical proofs of correctness and security properties, and empirical performance evaluations under realistic conditions. This comprehensive approach ensures that our contributions are not only theoretically sound but also viable for real-world deployment in diverse computing environments.

Quantum computing advancement continues at an accelerating pace, with recent developments suggesting that cryptographically relevant quantum computers may become reality sooner than previously anticipated. This increasing urgency underscores the importance of practical migration strategies for critical security protocols and infrastructure. Our work contributes to this goal by offering solutions that can be implemented incrementally, allowing systems to adopt quantum resistance without requiring immediate, wholesale changes to existing cryptographic infrastructure.

In the following sections, we discuss each major contribution in detail, examining their implications, limitations, and potential impact on the field. We begin with our innovations in redactable blockchain technology, which leverage specialized properties of chameleon hash functions to enable controlled mutability without compromising security. We then explore our novel approach to post-quantum TLS migration using identity-based encryption, highlighting how it addresses key challenges in certificate management during transition periods. Finally, we outline promising directions for future research that build upon the foundations established in this thesis.

## 6.2 CONTRIBUTIONS TO BLOCKCHAIN PROTOCOLS

This thesis introduces significant methodological innovations to enhance post-quantum blockchain protocols, with particular emphasis on addressing the fundamental tension between immutability and controlled mutability. Our primary contribution lies in developing a novel approach to block chaining that enables secure redaction without compromising cryptographic integrity or exposing vulnerabilities.

### 6.2.1 Novel Preimage-Based Redaction Mechanism

We developed a groundbreaking approach to blockchain redactability by leveraging the preimage property of specific chameleon hash constructions. This innovation fundamentally transforms how blocks can be modified:

- **Collision-Free Redactions:** Unlike previous approaches that leak collision information during redaction, thereby undermining security after multiple modifications, our mechanism enables redactions without exposing collisions, maintaining robustness even after multiple edits;
- **Rehabilitation of Traditional Constructions:** Our approach revitalizes several traditional chameleon hash functions previously considered impractical for blockchain

applications due to their weak collision-resistance. Notably, we identified two specific constructions that deliver superior performance while eliminating dependence on the non-standard generic group model;

- **Performance Advantages:** Our experimental evaluations demonstrated that these rehabilitated constructions achieve hash computation times just 13.5% and 20.5% slower than non-redactable blockchains, a substantial improvement over existing alternatives that incur penalties of 38% or more.

### 6.2.2 Transparency Through Structural Differentiation

A distinctive feature of our proposed redactable blockchain is what we term "structural differentiation", the explicit modification of block structure during redaction:

- **Transparent Modification History:** Redacted blocks adopt a differentiated format that includes additional data fields to maintain verifiability while explicitly signaling modification. This design creates permanent, visible "scars" in the blockchain structure;
- **Enhanced Accountability:** By making modifications structurally evident, our approach prevents surreptitious alterations and promotes accountability among trusted redactors. Users can unambiguously identify which blocks have been modified and when these changes occurred;
- **Trust Preservation:** This transparency directly addresses a critical limitation of previous redactable blockchain designs, which could undermine trust through invisible modifications.

### 6.2.3 Quantum-Resistant Security Architecture

Our redactable blockchain design represents the first construction based on chameleon hash functions that simultaneously achieves:

- **Post-Quantum Security:** By incorporating lattice-based chameleon hash constructions, our solution remains secure against quantum adversaries, addressing an increasingly critical requirement for long-term blockchain viability;
- **Minimal Cryptographic Dependencies:** Our architecture eliminates reliance on additional cryptographic primitives such as signature schemes, zero-knowledge proofs, or multiple trapdoors, significantly reducing implementation complexity and potential vulnerability surfaces;

- **Single-Key Simplicity:** Our design allows for efficient redaction with a single trapdoor key, simplifying key management while maintaining robust security properties.

This construction represents a significant advancement over previous approaches, which typically required either enhanced collision-resistance, necessitating complex mechanisms to prevent cascading modifications, or full collision-resistance, for which no quantum-resistant constructions have been demonstrated.

#### 6.2.4 Practical Impact and Extensions

Our work has immediate practical implications beyond theoretical advancement:

- **Universal Blockchain Applicability:** While our primary evaluation focused on the Bitcoin protocol, our methodology is applicable to any blockchain system requiring controlled mutability, including private chains for enterprise applications and consortium blockchains;
- **Content Moderation Solutions:** Our approach offers a technically sound solution to the pressing challenge of illegal content insertion in public blockchains, allowing for removal of prohibited material without compromising the overall system integrity;
- **Adaptability to Other Cryptographic Contexts:** The principles underlying our construction extend beyond blockchains to other applications requiring controlled modification capabilities, such as sanitizable signatures and mutable commitment schemes.

For example, our approach could be adapted to create efficient sanitizable signature schemes, particularly in scenarios requiring limited, predefined modification capabilities. In such implementations, a signer could generate distinct signatures for both original and potential sanitized document versions, with the sanitizable sections utilizing our preimage chameleon hash technique to ensure cryptographic consistency while explicitly tracking modifications.

The results of this research have been published in (ASTRIZI, T. L.; CUSTÓDIO, 2022).

### 6.3 CONTRIBUTIONS TO THE TLS 1.3 PROTOCOL

This thesis presents innovative approaches to the critical challenge of transitioning the Transport Layer Security (TLS) protocol to post-quantum security. Our contributions address the fundamental migration barriers that have hindered widespread adoption of quantum-resistant cryptography in this essential internet security protocol.

### 6.3.1 Novel Hybrid Architecture

We developed a groundbreaking hybrid KEMTLS protocol that fundamentally rethinks the approach to post-quantum migration:

- **Integration of Identity-Based Encryption:** Our protocol uniquely combines traditional public-key cryptography with post-quantum identity-based encryption, creating a hybrid system that leverages the strengths of both approaches while mitigating their respective weaknesses;
- **Certificate Infrastructure Preservation:** Unlike previous proposals that require modifications to certificate chains and formats, our approach enables quantum resistance while maintaining compatibility with existing X.509 certificate infrastructure;
- **Dual-Layer Defense Strategy:** By implementing both classical and post-quantum cryptographic primitives, we provide resilience against adversaries with access to both traditional and quantum computing capabilities;
- **KEMTLS Enhancement:** Our work extends the KEMTLS framework with identity-based mechanisms, addressing key limitations in the original proposal while preserving its performance advantages over traditional signature-based TLS approaches.

### 6.3.2 Implementation and Validation

We moved beyond theoretical proposals to demonstrate practical feasibility through implementation and rigorous testing:

- **Reference Implementation:** We developed a fully functional implementation of our hybrid protocol as an extension to the Go TLS library, demonstrating its viability in real-world software environments;
- **Cross-Continental Performance Evaluation:** Our experimental validation included testing across geographically distributed nodes in different continents, providing realistic performance measurements under actual internet conditions;
- **Comparative Analysis:** We conducted systematic comparisons against both classical TLS 1.3 and other proposed post-quantum TLS variants, establishing comprehensive performance benchmarks across different network conditions and latency scenarios;
- **Resource Utilization Assessment:** Our implementation was evaluated for computational overhead, memory requirements, and bandwidth consumption, demonstrating practical viability even in resource-constrained environments.

### 6.3.3 Cryptographic Foundations

Our work includes significant contributions to the theoretical underpinnings of post-quantum cryptography:

- **Security Model Innovation:** We developed a novel security model for hybrid key encapsulation mechanisms that integrates identity-based and traditional approaches, addressing the unique security considerations of such combined systems;
- **Rigorous Security Proofs:** We provided formal security proofs for our hybrid KEM construction, demonstrating security under the IND-CCA model even when one component is compromised;
- **Key Reuse Security Analysis:** We proved that using the same key pair for both ECDSA signatures and DHKEM does not compromise security, a critical result that enables certificate reuse without introducing vulnerabilities;
- **Combiner Construction:** Our XOR-then-MAC combiner construction for hybrid KEMs ensures that the overall security of the system is at least as strong as the security of its strongest component.

### 6.3.4 Migration Pathway Innovations

Our approach provides unique advantages for practical post-quantum migration:

- **Phased Transition Strategy:** We enable organizations to adopt post-quantum security in stages, beginning with the key exchange components while deferring authentication modifications to a later phase;
- **Certificate Authority Independence:** Our solution eliminates the need for certificate authorities to immediately support post-quantum algorithms, removing a major bottleneck in the transition process;
- **Deployment Flexibility:** The protocol supports multiple implementation scenarios, including pre-distributed keys for IoT environments, cached parameters for frequent connections, and full handshakes for new connections, accommodating diverse usage patterns;
- **Key Escrow Mitigation:** Our hybrid approach addresses the inherent key escrow concerns in identity-based systems through the maintenance of classical cryptographic protections.

### 6.3.5 Performance and Compatibility

Our experimental evaluations revealed significant practical advantages:

- **Comparable Latency Performance:** In realistic network scenarios with typical internet latencies ( $\geq 50\text{ms}$ ), our protocol exhibited only 2.6-7.1% additional delay compared to classical TLS 1.3 before application data transmission;
- **Competitive Handshake Efficiency:** Our protocol demonstrated handshake times comparable to or better than other hybrid post-quantum proposals despite differences in message sizes;
- **Bandwidth Optimization:** While our approach incurs bandwidth overhead in certain scenarios, we identified specific optimizations that provide favorable performance in high-latency environments typical of real-world internet connections;
- **Preservation of Early Data Capabilities:** Our protocol maintains TLS 1.3's ability to send application data early in the handshake process, preserving a key performance feature for latency-sensitive applications.

The results of this research were published in (ASTRIZI, T. L.; CUSTÓDIO, 2024) and offer a practical migration path for one of the internet's most critical security protocols. Importantly, while our implementation focused on TLS 1.3, the methodology is applicable to any protocol that relies on certificates and classical elliptic curve keys, including DTLS, QUIC, SSH, and various industry-specific secure communication protocols.

## 6.4 BROADER IMPLICATIONS FOR CRYPTOGRAPHY AND SECURE SYSTEMS

The methodologies and frameworks developed in this thesis extend beyond their immediate applications to blockchain redactability and TLS protocol evolution, offering insights and approaches that could influence multiple domains within cryptography and secure systems engineering.

### 6.4.1 Paradigm Shift in Cryptographic Migration Strategies

Our work challenges the conventional wisdom that post-quantum migration necessarily requires wholesale replacement of existing cryptographic infrastructure. The hybrid approach demonstrated in our TLS protocol adaptation suggests a more nuanced migration philosophy, one that emphasizes incremental adoption through strategic combination of classical and quantum-resistant primitives. This paradigm shift has significant implications for:

- **Critical Infrastructure Protection:** Organizations responsible for critical national infrastructure can begin hardening their systems against quantum threats without disrupting operational continuity, potentially accelerating widespread adoption of quantum-resistant technologies;
- **Standardization Processes:** Standards bodies might reconsider their approach to post-quantum standardization, embracing staged migration paths that maintain interoperability between classical and post-quantum systems during transitional periods;

#### 6.4.2 Reconciling Immutability with Practical Governance

The tension between perfect immutability and practical governance requirements pervades many distributed systems beyond blockchains. Our approach to controlled mutability through specialized cryptographic constructions offers a framework for addressing this fundamental tension in various contexts:

- **Distributed Data Systems:** Systems requiring both verifiable history and compliance with data protection regulations (such as the General Data Protection Regulation GDPR's "right to be forgotten") could adopt similar cryptographic mechanisms;
- **Smart Contract Platforms:** The ability to selectively modify blockchain content without compromising overall integrity could inform the design of smart contract upgrade mechanisms and governance frameworks;
- **Digital Archives:** Long-term archives requiring both provenance guarantees and redaction capabilities for sensitive content might leverage our approach to cryptographically secure redaction.

#### 6.4.3 Cryptographic Primitive Repurposing

Our work demonstrates how established cryptographic primitives can be repurposed and combined in novel ways to address emerging challenges. This approach to cryptographic innovation, finding new applications for existing, well-analyzed primitives rather than designing entirely new ones, offers several advantages:

- **Security Confidence:** By building on primitives with established security properties and extensive cryptanalysis, our constructions inherit a degree of confidence not immediately available to novel cryptographic schemes;
- **Implementation Efficiency:** Repurposing existing primitives allows leveraging optimized implementations and hardware acceleration already available for these constructions;

- **Accessibility:** This approach makes advanced cryptographic capabilities more accessible to mainstream developers by building on familiar primitives rather than requiring entirely new expertise.

#### 6.4.4 Bridging Theory and Practice in Cryptographic Engineering

This thesis exemplifies how rigorous theoretical analysis can be directly connected to practical implementation and real-world evaluation, a model that could influence cryptographic research methodology more broadly:

- **Realistic Adversary Modeling:** Our security analyses consider practical attack vectors and deployment constraints, rather than focusing exclusively on asymptotic properties;
- **Performance-Aware Security Design:** The methodical performance evaluations across diverse network conditions demonstrate how security mechanisms can be optimized for real-world deployment scenarios without compromising their fundamental guarantees;
- **Implementation Artifacts as Research Contributions:** By producing and evaluating working implementations, this work affirms the value of implementation artifacts as first-class research contributions in cryptography.

#### 6.4.5 Implications for Quantum-Era Security Architecture

Looking further ahead, the approaches developed in this thesis suggest architectural principles for security systems in the quantum computing era:

- **Defense in Depth Through Cryptographic Diversity:** The hybrid constructions demonstrated in this work suggest that cryptographic diversity, combining primitives based on different hardness assumptions, may become a standard architectural pattern for robust security systems;
- **Identity-Centric Security Models:** Our use of identity-based cryptography for post-quantum migration hints at a potential shift toward more identity-centric security architectures that reduce reliance on complex certificate hierarchies;
- **Controlled Mutability as a Design Principle:** The notion that perfect immutability might give way to cryptographically-controlled, transparent mutability could influence how we design secure logs, auditing systems, and other integrity-critical components.

These broader implications suggest that the specific technical contributions of this thesis may, in time, influence not just the protocols examined but the fundamental

approaches we take to designing, implementing, and evolving secure systems in an increasingly quantum-capable world.

## 6.5 FUTURE WORK

Building on this research, several avenues for future work can be explored to further improve post-quantum protocols:

### 6.5.1 Preimage Chameleon Hash Applications

- **Expanded Applications in Digital Signatures:** The preimage property of chameleon hash functions could be leveraged in signature schemes beyond sanitizable signatures, particularly for creating efficient one-time signatures with minimal verification overhead. Preliminary work in this direction by El Bansarkhani suggests potential for significant performance improvements in lattice-based signatures by incorporating preimage chameleon hash techniques;
- **Secure Multiparty Computation:** Explore using preimage chameleon hash functions to create more efficient commitment schemes in secure multiparty computation protocols, potentially reducing the communication complexity and improving privacy guarantees in distributed computing environments;
- **Dynamic Proofs of Storage:** Develop new mechanisms for cloud storage verification where preimage chameleon hash functions enable efficient updates to stored data while preserving verification capabilities. This would address a key limitation in current proof-of-storage systems that struggle with mutable data;
- **Formal Security Analysis Framework:** Create a comprehensive framework for analyzing the security properties of various chameleon hash constructions, particularly examining the relationship between preimage resistance and collision resistance in post-quantum settings.

### 6.5.2 IBE Optimization and Integration

- **Ciphertext Compression Techniques:** Investigate specialized compression algorithms for IBE ciphertexts that could significantly reduce the bandwidth overhead of our proposed protocol while maintaining security guarantees;
- **Hierarchical IBE for TLS:** Extend our approach to incorporate Hierarchical Identity-Based Encryption (HIBE) (HORWITZ; LYNN, 2002), which would enable more flexible key management and delegation capabilities while maintaining the advantages of our proposed system;

- **Hardware Acceleration:** Explore the potential for hardware acceleration of lattice-based IBE operations, particularly in constrained environments like IoT devices where computational resources are limited;
- **Quantum-Resistant IBE Variants:** Research novel constructions for IBE based on alternative post-quantum assumptions beyond lattices, such as code-based or multivariate cryptography, potentially offering different security-performance tradeoffs.

### 6.5.3 Post-Quantum Protocol Enhancements

- **Comprehensive Integration with NIST Standards:** Expand the hybrid KEMTLS protocol to fully support the finalized NIST post-quantum cryptography standards, ensuring compliance with emerging regulatory frameworks and industry best practices;
- **Migration Path Analysis:** Develop detailed migration frameworks for organizations transitioning to post-quantum TLS, including quantitative models for evaluating security-performance tradeoffs during different phases of deployment;
- **Client Authentication Mechanisms:** Extend the protocol to support client authentication using post-quantum identity-based signatures or other quantum-resistant mechanisms, providing comprehensive security for mutual authentication scenarios;
- **Privacy-Preserving Extensions:** Investigate the integration of privacy-enhancing technologies with our post-quantum TLS approach, particularly for scenarios where both confidentiality and anonymity are required.

### 6.5.4 Implementation and Deployment

- **Large-Scale Performance Evaluation:** Conduct extensive performance benchmarking in diverse network environments, including high-latency connections, congested networks, and mobile scenarios to identify optimization opportunities;
- **Hardware Security Module Integration:** Research approaches for integrating our protocol with hardware security modules, providing enhanced protection for cryptographic keys while maintaining the efficiency advantages of our approach;
- **Automated Testing Framework:** Develop comprehensive testing methodologies specifically designed to validate the security and performance of post-quantum TLS implementations across different platforms and environments;

- **Cloud Service Provider Integration:** Collaborate with major cloud service providers to implement and evaluate the protocol at scale, addressing practical deployment challenges in multi-tenant environments.

#### 6.5.5 Security Hardening and Verification

- **Formal Verification:** Apply formal verification techniques to prove the security properties of the hybrid KEMTLS protocol, particularly focusing on the security of the key derivation mechanisms and the hybrid combiner;
- **Quantum Side-Channel Analysis:** Investigate potential side-channel vulnerabilities specific to post-quantum algorithms, developing countermeasures to protect against timing attacks, power analysis, and other implementation-specific vulnerabilities;
- **Quantum Random Number Generation:** Explore the integration of quantum random number generators to enhance the security of key generation in post-quantum protocols, particularly for high-security applications;
- **Post-Compromise Security Models:** Develop new security models that address forward secrecy and post-compromise security in the context of post-quantum protocols, accounting for the unique characteristics of quantum adversaries.

#### 6.5.6 Standardization and Practical Adoption

- **Protocol Specification Development:** Create detailed technical specifications for standardization bodies, facilitating broader adoption and interoperability of our approach across different implementations;
- **Deployment Best Practices:** Develop comprehensive guidelines for organizations transitioning to post-quantum TLS, including configuration recommendations, monitoring strategies, and fallback mechanisms;
- **Compliance Framework:** Create a compliance framework for evaluating post-quantum TLS implementations against emerging regulatory requirements, particularly for industries with strict security mandates like healthcare and finance;
- **Educational Resources:** Develop educational materials and training programs to help organizations understand the implications of post-quantum cryptography and implement appropriate migration strategies.

These expanded future work directions build upon the foundation established in this thesis, addressing both theoretical and practical aspects of post-quantum cryptography implementation while maintaining alignment with current developments in the field.

## REFERENCES

- ALAGIC, Gorjan et al. Status report on the second round of the NIST post-quantum cryptography standardization process. **US Department of Commerce, NIST**, v. 2, p. 69, 2020.
- ALAGIC, Gorjan et al. Status report on the third round of the NIST post-quantum cryptography standardization process. US Department of Commerce, National Institute of Standards and Technology . . . , 2022.
- ALAMATI, Navid; MONTGOMERY, Hart; PATRANABIS, Sikhar; ROY, Arnab. Minicrypt Primitives with Algebraic Structure and Applications. In: ISHAI, Yuval; RIJMEN, Vincent (Eds.). **Advances in Cryptology – EUROCRYPT 2019**. Cham: Springer International Publishing, 2019. P. 55–82.
- ALNAHAWI, Nouri; MÜLLER, Johannes; OUPICKÝ, Jan; WIESMAIER, Alexander. A Comprehensive Survey on Post-Quantum TLS. **IACR Communications in Cryptology**, International Association for Cryptologic Research, v. 1, n. 2, 2024. DOI: 10.62056/ahee0iuc.
- ASTRIZI, Thiago; CUSTÓDIO, Ricardo; MOURA, Lucia. Post-quantum signature with preimage chameleon hashing. In: SBC. ANAIS do XX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. [S.l.: s.n.], 2020. P. 69–82.
- ASTRIZI, Thiago Leucz; CUSTÓDIO, Ricardo. Redacting Blockchain Without Exposing Chameleon Hash Collisions. In: SPRINGER. INTERNATIONAL Conference on Cryptology and Network Security. [S.l.: s.n.], 2022. P. 339–358.
- ASTRIZI, Thiago Leucz; CUSTÓDIO, Ricardo. Seamless Transition to Post-Quantum TLS 1.3: A Hybrid Approach Using Identity-Based Encryption. **Sensors**, MDPI, v. 24, n. 22, p. 7300, 2024.
- ATENIESE, Giuseppe; MAGRI, Bernardo; VENTURI, Daniele; ANDRADE, Ewerton. Redactable blockchain—or—rewriting history in bitcoin and friends. In: IEEE. 2017 IEEE European symposium on security and privacy (EuroS&P). [S.l.: s.n.], 2017. P. 111–126.
- ATENIESE, Giuseppe; MEDEIROS, Breno de. Identity-based chameleon hash and applications. In: SPRINGER. INTERNATIONAL Conference on Financial Cryptography. [S.l.: s.n.], 2004a. P. 164–180.

ATENIESE, Giuseppe; MEDEIROS, Breno de. On the key exposure problem in chameleon hashes. In: SPRINGER. INTERNATIONAL Conference on Security in Communication Networks. [S.l.: s.n.], 2004b. P. 165–179.

AVANZI, Roberto et al. CRYSTALS-Kyber algorithm specifications and supporting documentation. **NIST PQC Round**, v. 2, n. 4, p. 1–43, 2019. Available from: <https://www.pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf>.

BANERJEE, Utsav; CHANDRAKASAN, Anantha P. Efficient post-quantum TLS handshakes using identity-based key exchange from lattices. In: IEEE. ICC 2020-2020 IEEE International Conference on Communications (ICC). [S.l.: s.n.], 2020. P. 1–6.

BARKER, Elaine; CHEN, Lily; DAVIS, Richard, et al. Recommendation for key-derivation methods in key-establishment schemes. **NIST Special Publication**, v. 800, p. 56c, 2018.

BARNES, Richard; BHARGAVAN, Karthikeyan; LIPP, Benjamin; WOOD, Christopher A. **Hybrid Public Key Encryption**. [S.l.]: RFC Editor, 2022. RFC 9180. (Request for Comments, 9180). DOI: 10.17487/RFC9180. Available from: <https://www.rfc-editor.org/info/rfc9180>.

BELLARE, Mihir; BOLDYREVA, Alexandra; MICALI, Silvio. Public-key encryption in a multi-user setting: Security proofs and improvements. In: SPRINGER. ADVANCES in Cryptology—EUROCRYPT 2000: International Conference on the Theory and Application of Cryptographic Techniques Bruges, Belgium, May 14–18, 2000 Proceedings 19. [S.l.: s.n.], 2000. P. 259–274. DOI: 10.1007/3-540-45539-6\_18.

BELLARE, Mihir; RISTOV, Todor. A characterization of chameleon hash functions and new, efficient designs. **Journal of cryptology**, Springer, v. 27, n. 4, p. 799–823, 2014.

BINDEL, Nina; BRENDEN, Jacqueline; FISCHLIN, Marc; GONCALVES, Brian; STEBILA, Douglas. Hybrid key encapsulation mechanisms and authenticated key exchange. In: SPRINGER. POST-QUANTUM Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10. [S.l.: s.n.], 2019. P. 206–226. DOI: 10.1007/978-3-030-25510-7\_12.

BLAZY, Olivier; CHEVALIER, Céline. Generic Construction of UC-Secure Oblivious Transfer. In: MALKIN, Tal; KOLESNIKOV, Vladimir; LEWKO, Allison Bishop; POLYCHRONAKIS, Michalis (Eds.). **Applied Cryptography and Network Security**. Cham: Springer International Publishing, 2015. P. 65–86.

BLAZY, Olivier; CHEVALIER, Céline; GERMOUTY, Paul. Adaptive Oblivious Transfer and Generalization. In: CHEON, Jung Hee; TAKAGI, Tsuyoshi (Eds.). **Advances in Cryptology – ASIACRYPT 2016**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016. P. 217–247.

BLAZY, Olivier; CHEVALIER, Céline; VU, Quoc Huy. Post-Quantum UC-Secure Oblivious Transfer in the Standard Model with Adaptive Corruptions. In: PROCEEDINGS of the 14th International Conference on Availability, Reliability and Security. Canterbury, CA, United Kingdom: Association for Computing Machinery, 2019. (ARES '19). DOI: 10.1145/3339252.3339280. Available from: <https://doi.org/10.1145/3339252.3339280>.

BLAZY, Olivier; KAKVI, Saqib; KILTZ, Eike; PAN, Jiaxin. Tightly-secure signatures from chameleon hash functions. In: SPRINGER. IACR International Workshop on Public Key Cryptography. [S.l.: s.n.], 2015. P. 256–279.

BONEH, Dan; CORRIGAN-GIBBS, Henry. Bivariate Polynomials Modulo Composites and Their Applications. In: SARKAR, Palash; IWATA, Tetsu (Eds.). **Advances in Cryptology – ASIACRYPT 2014**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. P. 42–62.

BROWN, Daniel RL. Generic groups, collision resistance, and ECDSA. **Designs, Codes and Cryptography**, Springer, v. 35, n. 1, p. 119–152, 2005. DOI: 10.1007/s10623-003-6154-z.

CAMENISCH, Jan; DERLER, David; KRENN, Stephan; PÖHLS, Henrich C.; SAMELIN, Kai; SLAMANIG, Daniel. Chameleon-Hashes with Ephemeral Trapdoors. In: FEHR, Serge (Ed.). **Public-Key Cryptography – PKC 2017**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. P. 152–182.

CASH, David; HOFHEINZ, Dennis; KILTZ, Eike; PEIKERT, Chris. Bonsai trees, or how to delegate a lattice basis. In: SPRINGER. ANNUAL international conference on the theory and applications of cryptographic techniques. [S.l.: s.n.], 2010. P. 523–552.

CASTRYCK, Wouter; DECRU, Thomas. An efficient key recovery attack on SIDH. In: SPRINGER. **ADVANCES in Cryptology – EUROCRYPT 2023**. [S.l.: s.n.], 2023. P. 423–447. DOI: 10.1007/978-3-031-30589-4\_15.

CELI, Sofía; FAZ-HERNÁNDEZ, Armando; SULLIVAN, Nick; TAMVADA, Goutam; VALENTA, Luke; WIGGERS, Thom; WESTERBAAN, Bas; WOOD, Christopher A. Implementing and measuring KEMTLS. In: SPRINGER. **PROGRESS in**

Cryptology–LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6–8, 2021, Proceedings. [S.l.: s.n.], 2021. P. 88–107. DOI: 10.1007/978-3-030-88238-9\_5.

CHEN, Xiaofeng; ZHANG, Fangguo; KIM, Kwangjo. Chameleon hashing without key exposure. In: SPRINGER. INTERNATIONAL Conference on Information Security. [S.l.: s.n.], 2004. P. 87–98.

CHEN, Zhihong; LIU, Baolei; ZHENG, Yuliang. **The SM2 Cryptographic Algorithm**. [S.l.], 2013. Available from: <http://www.gmbz.org.cn/main/bz1b.html>.

CHEVALLIER-MAMES, Benoît; JOYE, Marc. A Practical and Tightly Secure Signature Scheme Without Hash Function. In: ABE, Masayuki (Ed.). **Topics in Cryptology – CT-RSA 2007**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. P. 339–356.

CLOUDFLARE. **CIRCL: Cloudflare Interoperable Reusable Cryptographic Library**. [S.l.: s.n.], 2024. <https://github.com/cloudflare/circl>. Accessed: 2024-10-01.

DERLER, David; SAMELIN, Kai; SLAMANIG, Daniel. Bringing Order to Chaos: The Case of Collision-Resistant Chameleon-Hashes. In: PUBLIC Key Cryptography (1). [S.l.: s.n.], 2020. P. 462–492.

DÖTTLING, Nico; GARG, Sanjam. From Selective IBE to Full IBE and Selective HIBE. In: KALAI, Yael; REYZIN, Leonid (Eds.). **Theory of Cryptography**. Cham: Springer International Publishing, 2017. P. 372–408.

DÖTTLING, Nico; GARG, Sanjam; ISHAI, Yuval; MALAVOLTA, Giulio; MOUR, Tamer; OSTROVSKY, Rafail. Trapdoor Hash Functions and Their Applications. In: BOLDYREVA, Alexandra; MICCIANCIO, Daniele (Eds.). **Advances in Cryptology – CRYPTO 2019**. Cham: Springer International Publishing, 2019. P. 3–32.

DUCAS, Léo; KILTZ, Eike; LEPOINT, Tancrede; LYUBASHEVSKY, Vadim; SCHWABE, Peter; SEILER, Gregor; STEHLÉ, Damien. CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme. **IACR Transactions on Cryptographic Hardware and Embedded Systems**, v. 2018, n. 1, p. 238–268, Feb. 2018. DOI: 10.13154/tches.v2018.i1.238-268. Available from: <https://tches.iacr.org/index.php/TCHES/article/view/839>.

DUCAS, Léo; LYUBASHEVSKY, Vadim; PREST, Thomas. Efficient identity-based encryption over NTRU lattices. In: SPRINGER. INTERNATIONAL Conference on the

Theory and Application of Cryptology and Information Security. [S.l.: s.n.], 2014. P. 22–41. DOI: 10.1007/978-3-662-45608-8\_2.

DUCAS, Léo; MICCIANCIO, Daniele. Improved short lattice signatures in the standard model. In: SPRINGER. ANNUAL Cryptology Conference. [S.l.: s.n.], 2014. P. 335–352.

EL BANSARKHANI, Rachid; BUCHMANN, Johannes. Improvement and Efficient Implementation of a Lattice-Based Signature Scheme. In: SELECTED Areas in Cryptography – SAC 2013, LNCS 8282. Berlin, Heidelberg: Springer, 2014. P. 48–67.

FERSCH, Manuel; KILTZ, Eike; POETTERING, Bertram. On the one-per-message unforgeability of (EC) DSA and its variants. In: SPRINGER. THEORY of Cryptography: 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II 15. [S.l.: s.n.], 2017. P. 519–534. DOI: 10.1007/978-3-319-70503-3\_17.

FERSCH, Manuel; KILTZ, Eike; POETTERING, Bertram. On the provable security of (EC) DSA signatures. In: PROCEEDINGS of the 2016 ACM SIGSAC Conference on Computer and Communications Security. [S.l.: s.n.], 2016. P. 1651–1662. DOI: 10.1145/2976749.2978413.

GARAY, Juan; KIAYIAS, Aggelos; LEONARDOS, Nikos. The Bitcoin Backbone Protocol: Analysis and Applications. In: OSWALD, Elisabeth; FISCHLIN, Marc (Eds.). **Advances in Cryptology - EUROCRYPT 2015**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. P. 281–310.

GENTRY, Craig; PEIKERT, Chris; VAIKUNTANATHAN. Trapdoors for hard lattices and new cryptographic constructions. In: PROCEEDINGS of the fortieth annual ACM symposium on Theory of computing. [S.l.: s.n.], 2008. P. 197–206.

GIACON, Federico; HEUER, Felix; POETTERING, Bertram. KEM combiners. In: SPRINGER. PUBLIC-KEY Cryptography–PKC 2018: 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part I 21. [S.l.: s.n.], 2018. P. 190–218. DOI: 10.1007/978-3-319-76578-5\_7.

GIRON, Alexandre Augusto; NASCIMENTO, João Pedro Adami do; CUSTÓDIO, Ricardo; PERIN, Lucas Pandolfo; MATEU, Víctor. Post-quantum hybrid KEMTLS performance in simulated and real network environments. In: SPRINGER. INTERNATIONAL Conference on Cryptology and Information Security in Latin America. [S.l.: s.n.], 2023. P. 293–312. DOI: 10.1007/978-3-031-44469-2\_15.

GOLDWASSER, Shafi; MICALI, Silvio; RIVEST, Ronald L. A digital signature scheme secure against adaptive chosen-message attacks. **SIAM Journal on computing**, SIAM, v. 17, n. 2, p. 281–308, 1988.

GONZALEZ, Ruben; WIGGERS, Thom. KEMTLS vs. post-quantum TLS: Performance on embedded systems. In: SPRINGER. INTERNATIONAL Conference on Security, Privacy, and Applied Cryptography Engineering. [S.l.: s.n.], 2022. P. 99–117. DOI: 10.1007/978-3-031-22829-2\_6.

GROTH, Jens; SHOUP, Victor. On the security of ECDSA with additive key derivation and presignatures. In: SPRINGER. ANNUAL International Conference on the Theory and Applications of Cryptographic Techniques. [S.l.: s.n.], 2022. P. 365–396. DOI: 10.1007/978-3-031-06944-4\_13.

GÜNEYSU, Tim; HODGES, Philip; LAND, Georg; OUNSWORTH, Mike; STEBILA, Douglas; ZAVERUCHA, Greg. Proof-of-possession for KEM certificates using verifiable generation. In: PROCEEDINGS of the 2022 ACM SIGSAC Conference on Computer and Communications Security. [S.l.: s.n.], 2022. P. 1337–1351. DOI: 10.1145/3548606.3560560.

GÜNEYSU, Tim; ODER, Tobias. Towards lightweight identity-based encryption for the post-quantum-secure Internet of Things. In: IEEE. 2017 18th International Symposium on Quality Electronic Design (ISQED). [S.l.: s.n.], 2017. P. 319–324. DOI: 10.1109/ISQED.2017.7918335.

HANSER, Christian; SLAMANIG, Daniel. Warrant-Hiding Delegation-by-Certificate Proxy Signature Schemes. In: PAUL, Goutam; VAUDENAY, Serge (Eds.). **Progress in Cryptology – INDOCRYPT 2013**. Cham: Springer International Publishing, 2013. P. 60–77.

HARTMANN, Dominik; KILTZ, Eike. Limits in the provable security of ECDSA signatures. In: SPRINGER. THEORY of Cryptography Conference. [S.l.: s.n.], 2023. P. 279–309. DOI: 10.1007/978-3-031-48624-1\_11.

HEMMINGER, Stephen et al. Network emulation with NetEm. In: LINUX conf au. [S.l.: s.n.], 2005. P. 8.

HOFHEINZ, Dennis. All-But-Many Lossy Trapdoor Functions. In: POINTCHEVAL, David; JOHANSSON, Thomas (Eds.). **Advances in Cryptology – EUROCRYPT 2012**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. P. 209–227.

HORWITZ, Jeremy; LYNN, Ben. Toward hierarchical identity-based encryption. In: SPRINGER. INTERNATIONAL conference on the theory and applications of cryptographic techniques. [S.l.: s.n.], 2002. P. 466–481.

HOU, Hongxia. Certificateless Based Chameleon Hash Scheme. In: 2012 International Conference on Computer Science and Service System. [S.l.: s.n.], 2012. P. 1126–1129. DOI: 10.1109/CSSS.2012.285.

ISHIZAKA, Masahito; MATSUURA, Kanta. Strongly Unforgeable Signature Resilient to Polynomially Hard-to-Invert Leakage Under Standard Assumptions. In: CHEN, Liqun; MANULIS, Mark; SCHNEIDER, Steve (Eds.). **Information Security**. Cham: Springer International Publishing, 2018. P. 422–441.

JAO, David et al. **SIKE - Supersingular Isogeny Key Encapsulation**. [S.l.: s.n.], 2021. <https://sike.org/>. Accessed: 19/06/2024.

JIA, Dingding; LIBERT, Benoît. SO-CCA secure PKE from pairing based all-but-many lossy trapdoor functions. **Designs, Codes and Cryptography**, Springer, v. 89, n. 5, p. 895–923, 2021.

KITCHENHAM, Barbara A; BUDGEN, David; BRERETON, O Pearl. 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.

KOMANO, Yuichi; OHTA, Kazuo; MIYAKE, Hideyuki; SHIMBO, Atsushi. Algorithmic Tamper Proof (ATP) Counter Units for Authentication Devices Using PIN. In: ABDALLA, Michel; POINTCHEVAL, David; FOUQUE, Pierre-Alain; VERGNAUD, Damien (Eds.). **Applied Cryptography and Network Security**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. P. 306–323.

KRAWCZYK, Hugo; RABIN, Tal. Chameleon Hashing and Signatures. **IACR Cryptol. ePrint Arch.**, v. 1998, p. 10, 1998.

LAI, Junzuo; DENG, Robert H.; LIU, Shengli. Chameleon All-But-One TDFs and Their Application to Chosen-Ciphertext Security. In: CATALANO, Dario; FAZIO, Nelly; GENNARO, Rosario; NICOLOSI, Antonio (Eds.). **Public Key Cryptography – PKC 2011**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. P. 228–245.

LIBERT, Benoît; QIAN, Chen. Lossy Algebraic Filters with Short Tags. In: LIN, Dongdai; SAKO, Kazue (Eds.). **Public-Key Cryptography – PKC 2019**. Cham: Springer International Publishing, 2019. P. 34–65.

LIU, Shengli; LAI, Junzuo; DENG, Robert H. General Construction of Chameleon All-But-One Trapdoor Functions. In: BOYEN, Xavier; CHEN, Xiaofeng (Eds.). **Provable Security**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. P. 257–265.

LU, Xingye; AU, Man Ho; ZHANG, Zhenfei. Raptor: A Practical Lattice-Based (Linkable) Ring Signature. In: DENG, Robert H.; GAUTHIER-UMAÑA, Valérie; OCHOA, Martín; YUNG, Moti (Eds.). **Applied Cryptography and Network Security**. Cham: Springer International Publishing, 2019a. P. 110–130.

LU, Xingye; AU, Man Ho; ZHANG, Zhenfei. Raptor: a practical lattice-based (linkable) ring signature. In: SPRINGER. INTERNATIONAL Conference on Applied Cryptography and Network Security. [S.l.: s.n.], 2019b. P. 110–130.

MATZUTT, Roman; HENZE, Martin; ZIEGELDORF, Jan Henrik; HILLER, Jens; WEHRLE, Klaus. Thwarting Unwanted Blockchain Content Insertion. In: 2018 IEEE International Conference on Cloud Engineering (IC2E). [S.l.: s.n.], 2018. P. 364–370. DOI: 10.1109/IC2E.2018.00070.

MICCIANCIO, Daniele; PEIKERT, Chris. Trapdoors for lattices: Simpler, tighter, faster, smaller. In: SPRINGER. ANNUAL International Conference on the Theory and Applications of Cryptographic Techniques. [S.l.: s.n.], 2012. P. 700–718.

MOHASSEL, Payman. One-time signatures and chameleon hash functions. In: SPRINGER. INTERNATIONAL Workshop on Selected Areas in Cryptography. [S.l.: s.n.], 2010. P. 302–319.

MOSCA, Michele; PIANI, Marco. 2021 quantum threat timeline report. **Global Risk Institute**, 2022.

NAKAMOTO, Satoshi. Bitcoin: A peer-to-peer electronic cash system. **Decentralized Business Review**, p. 21260, 2008.

NISHIMAKI, Ryo. Equipping Public-Key Cryptographic Primitives with Watermarking (or: A Hole Is to Watermark). In: PASS, Rafael; PIETRZAK, Krzysztof (Eds.). **Theory of Cryptography**. Cham: Springer International Publishing, 2020. P. 179–209.

OPEN QUANTUM SAFE PROJECT. **Repository with liboqs library**. [S.l.: s.n.], 2024. <https://github.com/open-quantum-safe/liboqs>.

PAQUIN, Christian; STEBILA, Douglas; TAMVADA, Goutam. Benchmarking post-quantum cryptography in TLS. In: SPRINGER. POST-QUANTUM Cryptography:

11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings 11. [S.l.: s.n.], 2020. P. 72–91.

PAUL, Sebastian; KUZOVKOVA, Yulia; LAHR, Norman; NIEDERHAGEN, Ruben. Mixed certificate chains for the transition to post-quantum authentication in TLS 1.3. In: PROCEEDINGS of the 2022 ACM on Asia Conference on Computer and Communications Security. [S.l.: s.n.], 2022. P. 727–740. DOI: 10.1145/3488932.3497755.

PENG, Changyan; ZHANG, Quan; TANG, Chaojing. Improved TLS Handshake Protocols Using Identity-Based Cryptography. In: 2009 International Symposium on Information Engineering and Electronic Commerce. [S.l.: s.n.], 2009. P. 135–139. DOI: 10.1109/IEEC.2009.33.

PETERSEN, Kai; VAKKALANKA, Sairam; KUZNIARZ, Ludwik. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and software technology**, Elsevier, v. 64, p. 1–18, 2015.

QIN, Baodong; LIU, Shengli. Leakage-Flexible CCA-secure Public-Key Encryption: Simple Construction and Free of Pairing. In: KRAWCZYK, Hugo (Ed.). **Public-Key Cryptography – PKC 2014**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. P. 19–36.

QIN, Baodong; LIU, Shengli. Leakage-Resilient Chosen-Ciphertext Secure Public-Key Encryption from Hash Proof System and One-Time Lossy Filter. In: SAKO, Kazue; SARKAR, Palash (Eds.). **Advances in Cryptology - ASIACRYPT 2013**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. P. 381–400.

RAY, Marsh; DISPENSA, Steve. **Renegotiating tls**. [S.l.]: November, 2009. Available from: [https://svn.dd-wrt.com/export/55387/src/router/matrixssl/doc/Renegotiating\\_TLS.pdf](https://svn.dd-wrt.com/export/55387/src/router/matrixssl/doc/Renegotiating_TLS.pdf).

RESCORLA, Eric. **The Transport Layer Security (TLS) Protocol Version 1.3**. [S.l.]: RFC Editor, 2018. RFC 8446. (Request for Comments, 8446). DOI: 10.17487/RFC8446. Available from: <https://www.rfc-editor.org/info/rfc8446>.

ROHLOFF, Kurt; COUSINS, Dave; POLYAKOV, Yuriy. **PALISADE Lattice Cryptography Library (release 1.9.2)**. [S.l.: s.n.], Apr. 2020. <https://palisade-crypto.org/>.

SCHMIDT-SAMOA, Katja; TAKAGI, Tsuyoshi. Paillier's Cryptosystem Modulo  $p^2q$  and Its Applications to Trapdoor Commitment Schemes. In: DAWSON, Ed; VAUDENAY, Serge (Eds.). **Progress in Cryptology – Mycrypt 2005**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. P. 296–313.

SCHWABE, Peter; STEBILA, Douglas; WIGGERS, Thom. More efficient post-quantum KEMTLS with pre-distributed public keys. In: SPRINGER. COMPUTER Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26. [S.l.: s.n.], 2021. P. 3–22. DOI: 10.1007/978-3-030-88418-5\_1.

SCHWABE, Peter; STEBILA, Douglas; WIGGERS, Thom. Post-quantum TLS without handshake signatures. In: PROCEEDINGS of the 2020 ACM SIGSAC Conference on Computer and Communications Security. [S.l.: s.n.], 2020. P. 1461–1480. DOI: 10.1145/3372297.3423350.

SCOTT, Michael. **On TLS for the Internet of Things, in a Post Quantum world**. [S.l.: s.n.], 2023. Cryptology ePrint Archive, Paper 2023/095. <https://eprint.iacr.org/2023/095>. Available from: <https://eprint.iacr.org/2023/095>.

SHAMIR, Adi; TAUMAN, Yael. Improved Online/Offline Signature Schemes. In: KILIAN, Joe (Ed.). **Advances in Cryptology — CRYPTO 2001**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. P. 355–367.

SHANKS, D. Five number-theoretic algorithms. **Proceedings of the Second Manitoba Conference on Numerical Mathematics (Winnipeg), 1973**, 1973. Available from: <https://cir.nii.ac.jp/crid/1572261550443913728>.

SHOR, Peter W. Algorithms for quantum computation: discrete logarithms and factoring. In: IEEE. PROCEEDINGS 35th annual symposium on foundations of computer science. [S.l.: s.n.], 1994. P. 124–134.

SHOR, Peter W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. **SIAM review**, SIAM, v. 41, n. 2, p. 303–332, 1999.

SHOUP, Victor. **A Proposal for an ISO Standard for Public Key Encryption**. [S.l.: s.n.], 2001. Cryptology ePrint Archive, Paper 2001/112. Available from: <https://eprint.iacr.org/2001/112>.

SHOUP, Victor. Sequences of games: a tool for taming complexity in security proofs. **cryptology eprint archive**, 2004.

SIKERIDIS, Dimitrios; KAMPANAKIS, Panos; DEVETSIKIOTIS, Michael. 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. [S.l.: s.n.], 2020. P. 149–156. DOI: 10.1145/3386367.3431305.

SMYTH, Ben; PIRONTI, Alfredo. Truncating {TLS} Connections to Violate Beliefs in Web Applications. In: 7TH USENIX Workshop on Offensive Technologies (WOOT 13). [S.l.: s.n.], 2013.

STEBILA, Douglas; FLUHRER, Scott; GUERON, Shay. Hybrid key exchange in TLS 1.3. **IETF draft**, 2020. Available from:  
<https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/>.

SUN, Yanbin; GU, Lize; QING, Sihan; ZHENG, Shihui; SUN, Bin; YANG, Yixian; SUN, Yan. Timeliness optimistic fair exchange protocol based on key-exposure-free chameleon hashing scheme. In: 2010 The 12th International Conference on Advanced Communication Technology (ICACT). [S.l.: s.n.], 2010. P. 1560–1564.

TECHNICAL REGULATING, Russian Federal Agency on; METROLOGY. **Information technology - Cryptographic data security - Signature and verification processes**. [S.l.: s.n.], 2012. GOST R 34.10-2012. Available from:  
<https://tc26.ru/standards/rs.html>.

THANALAKSHMI, P; ANITHA, R; ANBAZHAGAN, N; CHO, Woong; JOSHI, Gyanendra Prasad; YANG, Eunmok. A Hash-Based Quantum-Resistant Chameleon Signature Scheme. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 21, n. 24, p. 8417, 2021.

THE GO AUTHORS. **The Go Programming Language**. [S.l.: s.n.], 2024.  
<https://go.dev/>.

WANG, Yuyu; TANAKA, Keisuke. Generic Transformation to Strongly Existentially Unforgeable Signature Schemes with Continuous Leakage Resiliency. In: FOO, Ernest; STEBILA, Douglas (Eds.). **Information Security and Privacy**. Cham: Springer International Publishing, 2015. P. 213–229.

WANG, Zhiwei; YIU, Siu Ming. CCA Secure PKE with Auxiliary Input Security and Leakage Resiliency. In: LOPEZ, Javier; MITCHELL, Chris J. (Eds.). **Information Security**. Cham: Springer International Publishing, 2015. P. 319–335.

XU, Jie; XUE, Kaiping; TIAN, Hangyu; HONG, Jianan; WEI, David S. L.; HONG, Peilin. An Identity Management and Authentication Scheme Based on Redactable Blockchain for Mobile Networks. **IEEE Transactions on Vehicular Technology**, v. 69, n. 6, p. 6688–6698, 2020. DOI: 10.1109/TVT.2020.2986041.

YANG, Xuan; YU, Zhaoping; KANG, Bin. Chameleon-Based Optimistic Fair Exchange Protocol. In: 2008 International Conference on Embedded Software and Systems. [S.l.: s.n.], 2008. P. 298–302. DOI: 10.1109/ICISS.2008.26.

YAO, Jing; ZHENG, Yifeng; WANG, Cong; GUI, Xiaolin. Enabling Search Over Encrypted Cloud Data With Concealed Search Pattern. **IEEE Access**, v. 6, p. 11112–11122, 2018. DOI: 10.1109/ACCESS.2018.2810297.

YOSHIDA, Yusuke; KITAGAWA, Fuyuki; TANAKA, Keisuke. Non-Committing Encryption with Quasi-Optimal Ciphertext-Rate Based on the DDH Problem. In: GALBRAITH, Steven D.; MORIAI, Shiho (Eds.). **Advances in Cryptology – ASIACRYPT 2019**. Cham: Springer International Publishing, 2019. P. 128–158.

ZHANG, Di; LE, Junqing; LEI, Xinyu; XIANG, Tao; LIAO, Xiaofeng. Exploring the redaction mechanisms of mutable blockchains: A comprehensive survey. **International Journal of Intelligent Systems**, Wiley Online Library, v. 36, n. 9, p. 5051–5084, 2021.

ZHANG, Yilian; CHEN, Zhide; GUO, Fuchun. Online/Offline Verification of Short Signatures. In: LAI, Xuejia; YUNG, Moti; LIN, Dongdai (Eds.). **Information Security and Cryptology**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. P. 350–358.

# **Appendix**

## APPENDIX A – PROOF OF THEOREM 7

**Proof:** The proof consists of two steps. First, we assume that the classical Key Encapsulation Mechanism (KEM) is stronger than the identity-based KEM (IDKEM). In the second step, we assume the IDKEM is stronger. From these two cases, we conclude that in both scenarios, the security of the hybrid KEM is equivalent to the security of the strongest component. This means that even if one of the primitives is broken, the hybrid construction remains secure due to the strength of the other.

In both steps, the proof follows the standard hybrid argument, commonly used in indistinguishability proofs. We demonstrate that an adversary cannot distinguish between two different scenarios by gradually transforming one scenario into another through a series of intermediate "hybrid" scenarios. We show that an adversary can detect these gradual changes only if they are capable of breaking one of the security assumptions (either of the KEM or IDKEM).

### Step 1: Assuming that KEM is safer than IDKEM

In this step, we assume that the classical KEM is more secure than IDKEM. To prove that the security of our hybrid KEM remains as strong as the KEM in this case, we introduce hybrid games that transition between the two scenarios. We gradually replace real keys with random ones, demonstrating that an adversary cannot distinguish between these scenarios unless they can break the underlying KEM.

**Hybrid Game 2 and 3:** This game is described below and using Algorithm 25. It lies between hybrid KEM Game 0 and hybrid KEM Game 1. In Game 2, we replace the real key produced by  $\text{KEM.Encaps}$  with a randomly chosen uniform key, while keeping the ciphertext  $c_1$  intact. However, the key produced by  $\text{IDKEM.Encaps}$  remains unchanged. To ensure consistent behavior, the decapsulation query will return the same random value whenever it decapsulates  $c_1$ .

---

#### Hybrid Attack Game 2 and 3

---

```

1:  $id^* \leftarrow \emptyset$ 
2:  $Dict \leftarrow \text{EmptyDictionary}()$ 
3:  $(msk, mpk) \leftarrow \text{HKEM.MasterKeyGen}(1^\lambda)$ 
4:  $(id^*, st) \leftarrow \mathcal{A}_1^{\text{Decaps}(msk, \cdot), \text{Compromise}(msk, \cdot), \text{PublicKey}(\cdot)}(mpk)$ 
5:  $pk^* \leftarrow \text{PublicKey}(id^*)$ 
6:  $(c_1, k_{1a} \parallel k_{1b}) \leftarrow \text{KEM.Encaps}(pk^*)$ 
7:  $(c_2, k_{2a} \parallel k_{2b}) \leftarrow \text{IDKEM.Encaps}(mpk, id^*)$ 
8:  $k_{1a} \parallel k_{1b} \xleftarrow{\$} K$  ▷ Randomly chosen key from key space  $K$ 
9:  $k_{kem} \leftarrow k_{1a} \oplus k_{2a}$ 
10:  $k_{mac} \leftarrow k_{1b} \parallel k_{2b}$ 
11:  $c \leftarrow (c_1, c_2)$ 
12:  $t \leftarrow \text{MAC}_{k_{mac}}(c)$ 
13:  $b \leftarrow \mathcal{A}_2^{\text{Decaps}(msk, \cdot), \text{Compromise}(msk, \cdot), \text{PublicKey}(\cdot)}(st, (c, t), k_{kem})$ 
14: return  $b$ 

```

---

**Algorithm 31** Decapsulation Query for Hybrid Attack Games 2 and 3

---

```

1: Decaps( $msk, id, c$ ):
2: if  $Dict[id] \neq \emptyset$  then
3:    $(pk, sk) \leftarrow Dict[id]$ 
4: else
5:    $(pk, sk) \leftarrow HKEM.LocalKeyGen(1^\lambda)$ 
6:    $Dict[id] \leftarrow (pk, sk)$ 
7: end if
8:  $sk_{id} \leftarrow HKEM.Extract(msk, id)$ 
9:  $(c'_1, c'_2) \leftarrow c$ 
10:  $k'_{1a} \parallel k'_{1b} \leftarrow KEM.Decaps(sk, c'_1)$ 
11: if  $c'_1 = c_1$  then ▷ Only in Hybrid Game 2
12:    $k'_{1a} \parallel k'_{1b} \leftarrow k_{1a} \parallel k_{1b}$  ▷ Only in Hybrid Game 2
13: end if
14:  $k_{2a} \parallel k_{2b} \leftarrow IDKEM.Decaps(sk_{id}, c'_2)$ 
15:  $k_{kem} \leftarrow k'_{1a} \oplus k_{2a}$ 
16:  $k_{mac} \leftarrow k'_{1b} \parallel k_{2b}$ 
17: if  $t \neq MAC_{k_{mac}}(c)$  then
18:   return  $\perp$ 
19: else
20:   return  $k_{kem}$ 
21: end if

```

---

The primary difference between Games 2 and 3 and Hybrid KEM Game 0 for  $HKEM_{xtM}$  lies in the challenge. In Hybrid Game 2, we maintain consistency in the decapsulation query by checking if the ciphertext matches  $c_1$ , and if it does, we return the same random value every time. In Hybrid Game 3, this consistency is not maintained; each decapsulation of  $c_1$  will return a freshly generated random key.

Clearly, distinguishing between Hybrid KEM Game 2 and Hybrid KEM Game 0 is as hard as distinguishing between KEM Game 0 and KEM Game 1: if an adversary  $\mathcal{B}$  could distinguish between both of them, it could be used to produce an adversary  $\mathcal{A}_1$  capable of breaking the KEM security as described in Section 2.2.4. To show this, we build adversary  $\mathcal{A}_1$  as follows:

- **Initialization:** Adversary  $\mathcal{A}_1$  sets up the environment, initializing counters and variables, and selects one of the identities for the key encapsulation process.
  - 1: **procedure**  $\mathcal{A}_1(1^\lambda, pk, c^*, k^*)$
  - 2:  $Dict \leftarrow EmptyDictionary()$
  - 3:  $k \xleftarrow{\$} \{1, \dots, p\}$
  - 4:  $count \leftarrow 0$
  - 5:  $(mpk, msk) \leftarrow HKEM.MasterKeyGen(1^\lambda)$
  - 6:  $\mathcal{B}(mpk)$  ▷  $\mathcal{A}_1$  initializes adversary  $\mathcal{B}$  by providing the  $mpk$ .
  - 7: **end procedure**

- **Decapsulation Query:** Whenever  $\mathcal{B}$  asks to decapsulate a ciphertext for a specific identity,  $\mathcal{A}_1$  either performs the decapsulation using the KEM or checks for the secret key to perform decapsulation for IDKEM:

- If no key is found for the queried identity, a new key is generated and stored;
- If the current query matches the chosen identity  $k$ ,  $\mathcal{A}_1$  performs the decapsulation using its own decapsulation challenge from the KEM.

```

1: procedure  $\mathcal{A}_1(msk, id_i, c_i)$ 
2:    $(pk_i, sk_i) \leftarrow Dict.get(id_i)$ 
3:   if no key was found in  $Dict$  then
4:      $count \leftarrow count + 1$ 
5:     if  $count = k$  then
6:        $Dict.store(id_i, (pk, \emptyset))$ 
7:        $k_i \leftarrow Decaps(c_i[0])$   $\triangleright \mathcal{A}_1$  runs its own decapsulation query.
8:     else
9:        $(pk_i, sk_i) \leftarrow HKEM.KeyGen(1^\lambda)$ 
10:       $Dict.store(id_i, (pk_i, sk_i))$ 
11:    end if
12:  end if
13:  Use either  $msk$  or the decapsulation of  $c_i$  to compute the  $IDKEM$  decapsulation.
14:  Provide a valid response to  $\mathcal{B}$ .
15: end procedure

```

- **Compromise Query:** If  $\mathcal{B}$  asks to compromise an identity,  $\mathcal{A}_1$  reveals the corresponding secret key unless the query matches the target identity  $k$ , in which case  $\mathcal{A}_1$  halts. If the compromised identity is the chosen target,  $\mathcal{A}_1$  halts because it cannot proceed with the game.

```

1: procedure  $\mathcal{A}_1(id_i)$ 
2:   if no key was found in  $Dict$  then
3:      $count \leftarrow count + 1$ 
4:     if  $count = k$  then
5:        $\mathcal{A}_1$  cannot succeed and halts.
6:     else
7:        $(pk_i, sk_i) \leftarrow HKEM.KeyGen(1^\lambda)$ 
8:        $Dict.store(id_i, (pk_i, sk_i))$ 
9:     end if
10:  end if
11:  if  $sk_i = \emptyset$  then

```

```

12:      $\mathcal{A}_1$  cannot succeed and halts.
13: end if
14:      $sk_{id_i} \leftarrow HKEM.Extract(msk, id_i)$ 
15:     return  $(sk_i, sk_{id_i})$ 
16: end procedure

```

- **Public Key Query:** If  $\mathcal{B}$  requests the public key of an identity,  $\mathcal{A}_1$  checks the dictionary and returns the corresponding public key or creates a new one. If the queried identity is the target  $k$ ,  $\mathcal{A}_1$  returns the challenge's public key  $pk$ .

```

1: procedure  $\mathcal{A}_1(id_i)$ 
2:      $(pk_i, sk_i) \leftarrow Dict.get(id_i)$ 
3:     if no key was found in  $Dict$  then
4:          $count \leftarrow count + 1$ 
5:         if  $count = k$  then
6:              $Dict.store(id_i, (pk, \emptyset))$ 
7:             return  $pk$ 
8:         else
9:              $(pk_i, sk_i) \leftarrow HKEM.KeyGen(1^\lambda)$ 
10:             $Dict.store(id_i, (pk_i, sk_i))$ 
11:        end if
12:    end if
13:    return  $pk_i$ 
14: end procedure

```

- **Challenge:** When  $\mathcal{B}$  selects the target identity for its attack,  $\mathcal{A}_1$  gets the identity public key from the dictionary. Then, it submits the challenge ciphertext  $c^*$  and encapsulated key  $k^*$  to  $\mathcal{B}$ , which can be produced if the recovered public key is the same than the one that  $\mathcal{A}$  got during initialization.

```

1: procedure  $\mathcal{A}_1(id_i)$ 
2:      $(pk_i, sk_i) \leftarrow Dict.get(id_i)$ 
3:     if  $pk_i \neq pk$  then
4:          $\mathcal{A}_1$  cannot succeed and halts.
5:     end if
6:      $\mathcal{A}_1$  gets the challenge  $(c^*, k^*)$  from its challenger
7:     return  $(c^*, k^*)$ 
8: end procedure

```

- **Finalization:** When adversary  $\mathcal{B}$  finalizes returning some bit  $b$ , adversary  $\mathcal{A}_1$  also finalizes sending this same value  $b$  to its own  $KEM$  challenger.

Note that when the adversary  $\mathcal{A}_1$  interacts with KEM Game 0, it produces responses for  $\mathcal{B}$  that are consistent with Hybrid KEM Game 0. Similarly, when  $\mathcal{A}_1$  interacts with KEM Game 1, its responses are consistent with Hybrid KEM Game 2. The adversary  $\mathcal{A}_1$  will produce a coherent output without halting only if it correctly guesses the value of  $k$  during initialization; otherwise, it will halt during the challenge phase or in response to a decapsulation query. The probability of guessing correctly is  $1/p$ . Assuming this, the probability of  $\mathcal{A}_1$  succeeding is exactly the same as the probability of  $\mathcal{B}$  succeeding. Therefore, if  $W_0$  denotes the event where adversary  $\mathcal{B}$  outputs 1 in Hybrid KEM Game 0, and  $W_2$  denotes the event in Hybrid KEM Game 2, we have the same probability of success for the corresponding adversary  $\mathcal{A}_1$ :

$$\text{Adv}_{\text{KEM}}^{\text{IND-CCA}}(\mathcal{A}_1) \geq \frac{1}{p} |Pr[W_2] - Pr[W_0]|$$

This result provides a negligible upper bound for distinguishing between the two games:

$$|Pr[W_2] - Pr[W_0]| \leq p \cdot \text{Adv}_{\text{KEM}}^{\text{IND-CCA}}(\mathcal{A}_1)$$

**Hybrid Game 3:** This hybrid game is very similar to Hybrid Game 2. The key difference is that we no longer maintain consistent responses for decapsulation queries when the adversary attempts to decapsulate the value  $c_1$ . Instead, if such a query is made, we return the actual decapsulation result. In other words, we omit lines 11 and 12 in the Decapsulation procedure described in Algorithm 31. Other than this change, Hybrid Game 3 is defined exactly as Hybrid Game 2.

Notice that in Hybrid Game 3, the only way for the adversary to distinguish a difference in queries compared to Hybrid Game 2 is by making a decapsulation query where the second element matches the one in the challenged ciphertext, while the first element differs. Additionally, the adversary must produce a valid tag that is accepted by the MAC; otherwise, in both Hybrid games, the decapsulation query would return  $\perp$ .

Therefore, if there existed an adversary  $\mathcal{B}$  capable of distinguishing between Hybrid Game 3 and Hybrid Game 2, we could construct an adversary  $\mathcal{A}_2$  that successfully wins the MAC Attack Game (Subsection 2.2.6). We build  $\mathcal{A}_2$  as follows:

- **Initialization:** Adversary  $\mathcal{A}_2$  aims to break the MAC security by controlling the rightmost bits of the key, which are generated by IDKEM. These bits cannot be treated as independent, random, and uniform in this game, so it is crucial to control them in the MAC key. The steps are as follows:
  1. Use the KEM, IDKEM, and MAC algorithms to initialize adversary  $\mathcal{B}$  with key  $mpk$ , providing valid responses for all queries made by  $\mathcal{B}$ .  $\mathcal{A}_2$  interacts with  $\mathcal{B}$  in the same way as a legitimate challenger would, until  $\mathcal{B}$  issues its

challenge by selecting  $id$  as the target identity. Let  $pk$  be the corresponding public key associated with this identity;

2. Use the IDKEM and KEM algorithms to generate an encapsulation  $c^*$  for the identity  $id$ , while also obtaining the secret  $k_2$  produced by IDKEM and the encapsulation  $c_2$ ;

3. **Return**  $(id, 1, k_2)$

- **Finalization:**  $\mathcal{A}_2$  continues interacting with  $\mathcal{B}$  to attempt to produce a forgery. It follows the steps below:

1. Sends  $((c^*, tag^*), k^*)$  as challenge to  $\mathcal{B}$ . Here,  $c^*$  is the encapsulation previously generated for  $id$ ,  $tag^*$  is the tag returned by the challenger, and  $k^*$  is a random and uniformly chosen value;
2. Continue using the KEM, IDKEM and MAC algorithms to generate valid and consistent responses for all the queries made by  $\mathcal{B}$ , provided that no decapsulation query involving  $c_1$  is sent;
3. If a decapsulation query involving  $c_1$  and a given tag is received, use the *Verification* procedure to check the validity of the tag. If the tag is rejected, return  $\perp$  as response. Otherwise,  $\mathcal{A}_2$  halts and submits the tag as a successful forgery.

As discussed, the probability of  $\mathcal{A}_2$  succeeding is equivalent to the probability of  $\mathcal{B}$  detecting a difference between Hybrid Game 2 and Hybrid Game 3, which corresponds to the probability that it forges a valid tag for the MAC. Therefore, we obtain the following upper bound:

$$|Pr[W_3] - Pr[W_2]| \leq Adv_{MAC}^{OT-sEUF}(\mathcal{A}_2)$$

Finally, there is no difference between Hybrid Game 3 and KEM Game 1. In both games, the true decapsulation result is returned for all queries, without attempting to maintain consistency with the encapsulation from the challenge. In Hybrid Game 3, the secret is computed using a XOR operation between a random, uniform value and some other value  $k_{2a}$  (as in Hybrid Attack Games 2 and 3, line 9). The result of this XOR operation is also a random and uniform value. Therefore, there is no difference between the two games:

$$|Pr[W_1] - Pr[W_3]| = 0$$

By combining all these results, we conclude that for any adversary  $\mathcal{A}_4$ , there exist adversaries  $\mathcal{A}_1$  and  $\mathcal{A}_3$  such that:

$$Adv_{HKEM_{xtM}}^{IND-CCA}(\mathcal{A}_4) \leq p \cdot Adv_{KEM}^{IND-CCA}(\mathcal{A}_1) + Adv_{MAC}^{OT-sEUF}(\mathcal{A}_3)$$

Therefore, if KEM is more secure than IDKEM, we can use this upper bound while ignoring the security of IDKEM.

**Step 2: Assuming IDKEM is more secure than KEM**

In the previous step, we demonstrated that it is possible to ignore the security of IDKEM and state the security of HKEM purely in terms of the security of KEM and MAC. Thus, even if IDKEM is completely insecure, this would not necessarily compromise the security of HKEM.

In this step, we will show that it is also possible to express the security of HKEM solely in terms of IDKEM and MAC security, ignoring KEM. Therefore, even if KEM is completely insecure, HKEM would still remain secure based on the other primitives.

We now introduce the new hybrid games: Game 2' and Game 3'. Since they are very similar to Games 2 and 3, we will describe them briefly and explain how they help complete the proof.

**Hybrid Game 2'**: The primary difference here is that instead of replacing the KEM secret (as done in line 8 of Hybrid Attack Game 2 and 3) with a random, uniform value, we replace IDKEM secret, i.e., " $k_{2a} || k_{2b} \xleftarrow{\$} K$ ". We then show that if an adversary can distinguish between Hybrid Game 2' and Hybrid KEM Game 0, this adversary can be used to construct an adversary  $\mathcal{A}_3$  that breaks the IND-CCA security for IDKEM. The remainder of the proof is analogous, except that adversary  $\mathcal{A}_3$  does not need to guess which identity will be attacked. Since both KEMs are in the multi-user setting,  $\mathcal{A}_3$  can make queries to respond to all of  $\mathcal{B}$ 's queries. Adversary  $\mathcal{A}_3$  chooses which identity to attack only after the adversary  $\mathcal{B}$  makes its own choice. This give us the following upper bound:

$$|Pr[W'_2] - Pr[W_0]| \leq Adv_{IDKEM}^{IND-CCA}(\mathcal{A}_3)$$

**Hybrid Game 3'**: Similar to Hybrid Game 3, we modify the Decapsulation query in Hybrid Game 2' to always return the correct decapsulation, without maintaining consistency with the random value that replaced the decapsulation of  $c_2$  in Hybrid Game 2'. As before, distinguishing between Game 2' and Game 3' is possible only if an adversary can forge a valid MAC tag. In this case,  $\mathcal{A}_3$  works as before, but instead controlling the rightmost bits of the MAC key, it controls the leftmost bits which must be consistent with the secret  $k_1$  produced by KEM. This modification does not affect the results; we obtain the same upper bound as before.

By combining both cases, we conclude that for any adversary  $\mathcal{A}_4$ , there exist adversaries  $\mathcal{A}_1$ ,  $\mathcal{A}_2$  and  $\mathcal{A}_3$  such that:

$$Adv_{HKEM_{XIM}}^{IND-CCA}(\mathcal{A}_4) \leq \min \left\{ p \cdot Adv_{KEM}^{IND-CCA}(\mathcal{A}_1), Adv_{IDKEM}^{IND-CCA}(\mathcal{A}_2) \right\} + Adv_{MAC}^{OT-sEUF}(\mathcal{A}_3)$$

This upper bound concludes our security proof, demonstrating that under our assumptions, the advantage of any attacker in compromising the security of the scheme is negligible.  $\square$

## APPENDIX B – PROOF OF THEOREM 8

**Proof:** As discussed in the text, we use a modified KEM Attack Game where the adversary attempts to distinguish between Game 0 and Game 1 or forge an ECDSA signature in either game. This augmented KEM Attack Game allows the adversary to query ECDSA signatures and assumes that the signature and the KEM (a DHKEM) share the same key.

Following the proof in (FERSCH et al., 2016), the operation on line 5 of the ECDSA signature algorithm (Subsection 2.2.7.1) is modeled as a composition of functions  $\psi \circ \Pi \circ \varphi$  that maps a group element to an integer modulo  $q$ . The function  $\varphi$  maps an elliptic curve point to a string encoding the x-coordinate of the point,  $\Pi$  is a bijective random oracle, and  $\psi$  performs the modulo  $q$  operation, where  $q$  is the elliptic curve group order.

First, we create a Hybrid Attack Game 2 and demonstrate that if an adversary  $\mathcal{B}$  can forge signatures or distinguish between Attack Game 0 and Game 1, then we can solve the computational Diffie-Hellman (CDH) problem using an adversary  $\mathcal{A}$ . Adversary  $\mathcal{A}$  does not precisely simulate Game 0 or Game 1 from the KEM Attack Game, but instead models a new Hybrid Game 2, which is a combination of the two. Adversary  $\mathcal{A}$  operates as follows:

- **Initialization:** Adversary  $\mathcal{A}$  is initialized with the security parameter  $1^\lambda$  and a computational Diffie-Hellman instance  $(G, g, q, y_a = g^a, y_b = g^b)$ , where its objective is to compute  $g^{ab}$ .

To simulate the bijective random oracle  $\Pi : A \rightarrow B$ ,  $\mathcal{A}$  initializes several lists:

- $\Pi^\rightarrow$ : stores tuples  $(a, b)$  with  $a \in A$  and  $b \in B$  for forward queries;
- $\Pi^\leftarrow$ : stores tuples  $(a, b, \cdot)$  for backward queries;
- *DecapsList*: stores values for consistent decapsulation responses;
- *OracleList*: stores oracle query responses;
- *SignList*: stores ECDSA signatures.

Additionally,  $\mathcal{A}$  samples  $Q$  distinct values  $\beta_1, \dots, \beta_Q$  from set  $B$  (expected number of queries to the random oracle) and initializes a counter  $t = 0$  to track queries.

Finally,  $\mathcal{A}$  initializes adversary  $\mathcal{B}$ , providing it with randomness  $\rho$ , the security parameter  $1^\lambda$ , and the public key  $pk = y_a$ .

- **Bijective Random Oracle Query:** When adversary  $\mathcal{B}$  queries  $\Pi(a_i)$ ,  $\mathcal{A}$  first checks if  $(a_i, b_i) \in \Pi^\rightarrow$  or if  $(a_i, b_i, \cdot) \in \Pi^\leftarrow$ . If such a tuple exists, it returns  $b_i$ . Otherwise, it increments the counter  $t$ , stores  $(a_i, \beta_t)$  in  $\Pi^\rightarrow$ , and responds with  $\beta_t$ ;

- **Backward Bijective Random Oracle Query:** When adversary  $\mathcal{B}$  queries  $\Pi^{-1}(b_i)$ ,  $\mathcal{A}$  checks if  $(a_i, b_i)$  exists in either  $\Pi^{\rightarrow}$  or  $\Pi^{\leftarrow}$ . If not, it selects a random  $a_i \in A$ , ensuring  $a_i$  is not already in  $\Pi^{\leftarrow}$  or  $\Pi^{\rightarrow}$ . If  $a_i$  lies within the range of  $\varphi$ ,  $\mathcal{A}$  samples a random  $\alpha \in \mathbb{Z}_q$  and sets  $a_i = \varphi(g^\alpha)$ . It then stores  $(a_i, b_i, \alpha)$  in  $\Pi^{\leftarrow}$  and responds with  $a_i$ .
- **KEM Random Oracle Query:** When adversary  $\mathcal{B}$  sends a query requesting the KEM random oracle output for message  $m_i$ , we proceed as follows:
  1. If  $(m_i, k_i) \in \text{OracleList}$ , return  $k_i$  as a response;
  2. Attempt to parse  $m_i$  as three elliptic curve points from the group  $G$ . If parsing fails, sample a random value  $k_i$  uniformly from the oracle range, store  $(m_i, k_i)$  in OracleList, and return  $k_i$  as the response;
  3. Let  $(y_1, y_2, y_3)$  be the three elliptic curve points parsed from  $m_i$ . If  $y_3 \neq y_a$  or if  $(y_1, y_2, y_3)$  is not a valid Diffie-Hellman tuple, sample a random value  $k_i$  uniformly from the oracle range, store  $(m_i, k_i)$  in OracleList, and return  $k_i$  as the response;
  4. If  $(y_1, y_2, y_3)$  is a valid Diffie-Hellman tuple and  $y_3 = y_a$ , check if  $y_2 = y_b$ . If this holds, then  $y_1 = y^{ab}$ , and we have found a solution to the computational Diffie-Hellman problem. If  $y_2 \neq y_b$ , check whether there is a tuple  $(y_2, k_i)$  in DecapsList. If such a tuple exists, return  $k_i$  as the response. If no response has been produced yet, sample a random value  $k_i$  uniformly from the oracle range, store  $(m_i, k_i)$  in OracleList, and also store  $(y_2, k_i)$  in DecapsList to remember that  $k_i$  is the correct decapsulation value for  $y_2$  when using the public key  $y_a$ . Finally, return  $k_i$  as the response.
- **Signing Query:** When adversary  $\mathcal{B}$  sends a signing query with message  $msg_i$ , adversary  $\mathcal{A}$  can simulate a valid signature, even without knowledge of the secret key, as follows. First, it selects a value  $b_i$  uniformly at random from the bijective random oracle range  $B$ . If  $(\cdot, b_i) \in \Pi^O$  (which occurs with negligible probability),  $\mathcal{A}$  is unable to produce a consistent signature and halts. Otherwise,  $\mathcal{A}$  samples a random value  $s_i \leftarrow \mathbb{Z}_q$  and computes:

$$y' \leftarrow g^{\text{Hash}(msg)s_i^{-1}} y_a^{\varphi(b_i)s_i^{-1}}$$

Next, it computes  $a_i \leftarrow \phi(y')$  and checks whether either  $a_i$  or  $b_i$  already appears in some tuple in  $\Pi^O$ . If either does,  $\mathcal{A}$  fails to simulate a consistent bijective random oracle and halts. Otherwise, it stores  $(a_i, b_i)$  in  $\Pi^{\leftarrow}$  and returns the signature  $(\varphi(b_i), s_i)$  in response to the query.

If an invalid value is generated when creating the signature (e.g.,  $r_i = \varphi(b_i) = 0$  or  $s_i = 0$ ),  $\mathcal{A}$  can attempt the process again, as done in standard signature

generation. All messages and signatures generated through this query are stored in the list *SignList*;

- **Challenge:** To generate the challenge, adversary  $\mathcal{A}$  constructs the KEM challenge by selecting  $y_b = g^b$  as the ciphertext  $c$ . The correct secret should be the output of the random oracle for the input  $g^{ab}||g^b||g^a$ . However, since  $\mathcal{A}$  does not know the value of  $g^{ab}$ , it instead selects a value  $k$  uniformly at random from the oracle range and sends  $(g^b, k)$  as the challenge.

Since the secret  $k$  is chosen randomly and independently of the ciphertext, this behavior is indistinguishable from that of Game 1, where the secret is also chosen at random.

- **Decapsulation Query:** When adversary  $\mathcal{B}$  queries the decapsulation of some ciphertext  $c_i \neq y_b$ , adversary  $\mathcal{A}$  checks if a corresponding tuple  $(c_i, k_i)$  exists in the *DecapsList*. If such a tuple is found, it returns the value  $k_i$  as the response. Otherwise,  $\mathcal{A}$  selects a value  $k$  uniformly at random from the range of the random oracle, stores  $(c_i, k_i)$  in the *DecapsList*, and returns  $k_i$  as the response.
- **Finalization:** After all queries have been processed, we assume that adversary  $\mathcal{B}$  produces a bit as output, attempting to distinguish whether it was in Game 0 or Game 1, as well as a potential forgery for the ECDSA signature.

As mentioned earlier, the game that adversary  $\mathcal{A}$  simulates is indistinguishable from Game 1 because the secret  $k$  sent during the challenge was chosen uniformly at random and independently of the ciphertext  $c = g^b$ . However, given the nature of the random oracle, the only way to determine that the secret  $k$  is random and not the decapsulation of  $g^b$  is to identify that the oracle output for  $g^{ab}||g^b||g^a$  (the real encapsulation of  $g^b$ ) is different from  $k$ . Adversary  $\mathcal{A}$  would only discover this if it queried the oracle with  $g^{ab}||g^b||g^a$ . In doing so, it would reveal the solution  $g^{ab}$  to the computational Diffie-Hellman (CDH) problem. Thus, even when using the keys to generate signatures, breaking the DH-KEM is as hard as solving the CDH problem.

Now, suppose a valid forgery  $(msg^*, (r^*, s^*))$  is produced for the ECDSA signature. From these values, adversary  $\mathcal{A}$  can compute:

$$y^* \leftarrow g^{\text{Hash}(msg^*)/s^*} y_a^{r^*/s^*}.$$

Let  $k$  be the discrete logarithm of  $y^*$ . We know from the exponents that:

$$ks^* = \text{Hash}(msg^*) + ar^*,$$

even without knowledge of the values  $k$  and  $a$ . At this point, it is possible to compute  $a^* \leftarrow \varphi(y^*)$ , and we can assume, without loss of generality, that a pair in  $\Pi^\rightarrow$  or  $\Pi^\leftarrow$  contains  $a^*$  as the first element. This could occur either because adversary  $\mathcal{A}$  queried it from the bijective random oracle or because this value was produced during a signing query.

**Case 1:** if  $(a^*, b^*) \in \Pi^\rightarrow$  was produced during a signing query,  $\mathcal{A}$  must have also generated a past signature in the form  $(msg', (r', s'))$  such that:

$$y' = g^{\text{Hash}(msg')/s'} y_a^{r'/s'} \quad \text{and} \quad y' = g^{\pm k*}.$$

By searching in *SignList*, adversary  $\mathcal{A}$  can recover these values. This yields the equation:

$$\text{Hash}(msg^*)/s^* + ar^*/s^* = \pm(\text{Hash}(msg')/s'^{-1} + ar'/s'),$$

from which adversary  $\mathcal{A}$  can compute  $a$  and subsequently derive  $y_b^a = g^{ab}$ , solving the computational Diffie-Hellman problem.

**Case 2:** If  $(a^*, b^*, \alpha) \in \Pi^\leftarrow$ , the tuple was produced by a backward bijective random oracle query. In this case, we know the value  $\alpha$  such that  $g^{\pm\alpha} = y^*$ . From this, and the definition of  $y^*$ , adversary  $\mathcal{B}$  obtains the equation:

$$\pm\alpha = \text{Hash}(msg^*)/s^* + ar^*/s^*,$$

which allows it to isolate the discrete logarithm  $a$  and solve the computational Diffie-Hellman problem.

**Case 3:** If  $(a^*, b^*) \in \Pi^\rightarrow$  was produced by a bijective random oracle query, then  $b^* = \beta_j$  for some  $0 < j \leq Q$ .

In this case, adversary  $\mathcal{A}$  can generate a new sequence of distinct values from  $B : \beta'_j, \beta'_{j+1}, \dots, \beta'_Q$ , and run adversary  $\mathcal{B}$  again using fresh lists but the same randomness source  $\rho$  and input as before. The only difference is that when the counter  $t$  increments to some value  $l \geq j$ ,  $\mathcal{A}$  uses the new value  $\beta'_l$  instead of the previous one. By the Forking Lemma, if  $\mathcal{A}$  produces a forgery of this type with non-negligible probability, then with non-negligible probability, two forgeries can be obtained from both executions: one with the value  $\beta_j$  in the signature, and the second with  $\beta'_j$ . If not,  $\mathcal{A}$  fails and aborts.

Let  $(msg^{**}, (r^{**}, s^{**}))$  be the second forgery. Combining the signatures from both executions, adversary  $\mathcal{A}$  can derive the equation:

$$\pm(\text{Hash}(msg^*)/s^* + ar^*/s^*) = \text{Hash}(msg^{**})/s^{**} + ar^{**}/s^{**},$$

from which it can compute the discrete logarithm  $a$ , thereby solving the computational Diffie-Hellman problem.

The two additional possibilities not previously discussed are:

1. Adversary  $\mathcal{B}$  produces a forgery by finding a collision in the hash function;
2. In the final equation, an adversary could produce values such that  $\text{hash}(msg^{**})/r^{**} = \text{Hash}(msg^*)/r^*$ .

In either of these cases, adversary  $\mathcal{A}$  would fail to compute the discrete logarithm. However, these events occur with negligible probability if  $G$  is collision-resistant and has high division entropy resistance, as required for the security of the ECDSA signature scheme (as discussed in (FERSCH et al., 2016)).

Thus, except with negligible probability, adversary  $\mathcal{A}$  will successfully simulate a bijective random oracle (we omit the calculation of these probabilities since they are identical to those found in the ECDSA proof in (FERSCH et al., 2016)). If adversary  $\mathcal{B}$  forges a signature, adversary  $\mathcal{A}$  can compute the solution to the computational Diffie-Hellman problem. Furthermore, adversary  $\mathcal{A}$  can also compute a solution with non-negligible probability if adversary  $\mathcal{B}$  breaks the IND-CCA2 security of the DH-KEM with non-negligible probability.  $\square$