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

Lucas Mayr de Athayde

On the Improvements to Digital Document Signature Through PKI Transparency

Florianópolis
2024

Lucas Mayr de Athayde

On the Improvements to Digital Document Signature Through PKI Transparency

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de Mestre em Ciência da Computação.
Orientador: Prof. Ricardo Felipe Custódio, Dr.

Florianópolis
2024

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

Athayde, Lucas Mayr de
On the Improvements to Digital Document Signature Through PKI Transparency / Lucas Mayr de Athayde ; orientador, Ricardo Felipe Custódio, 2024.
75 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2024.

Inclui referências.

1. Ciência da Computação. 2. Infraestrutura de chaves públicas. 3. Documentos digitais. 5. Certificado digital. I. Custódio, Ricardo Felipe. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Lucas Mayr de Athayde

On the Improvements to Digital Document Signature Through PKI Transparency

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

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

Prof. Luiz Carlos Zancanella, Dr.
Universidade Federal do Rio Grande do Sul

Prof. Claudio Ribeiro Lima, Dr.
University of Kent

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

ICP-Edu
Documento assinado digitalmente
Marcio Bastos Castro
Data: 12/08/2024 18:50:33-0300
CPF: ***.707.720-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Márcio Bastos Castro, Dr.
Coordenador do Programa

ICP-Edu
Documento assinado digitalmente
Ricardo Felipe Custodio
Data: 12/08/2024 15:55:16-0300
CPF: ***.981.609-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Ricardo Felipe Custódio, Dr.
Orientador

Florianópolis, 2024.

This thesis is dedicated to my dear mother, Adriana.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Prof. Dr. Ricardo Felipe Custódio, for the many discussions about security, privacy, and usability that led to this theme and for all the hours spent debating the many ways in which we could potentially improve public key infrastructure. Similarly, many thanks to all my peers at LabSEC for putting up with my shenanigans for so long. Last but certainly not least, I would like to express my deepest thanks to my family, especially my mother, Adriana, for all the support they have given me throughout my academic studies, without which I would not have reached this far.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

"And if that only inflames your curiosity, I say to you,
a writer without curiosity is a bird without feathers."
(SALYARDS, 2012)

RESUMO

Documentos eletrônicos são assinados usando-se chaves privadas e verificados com o auxílio do certificado digital correspondente através do conhecido modelo de infraestrutura de chaves públicas. Uma das maiores ameaças de se usar esta infraestrutura é o comprometimento de chaves privadas. Portanto, tais chaves devem ser guardadas em dispositivos especificamente construídos para tal propósito para que possam ser reutilizadas. Sendo assim, o gerenciamento destas chaves é de essencial importância para o qual não existe solução infalível. Consequentemente, dada a existência da possibilidade de comprometimento, faz-se necessário a existência de mecanismos que possam revogar futuras assinaturas vindas de chaves comprometidas. Listas de certificados revogados e o protocolo *online certificate status* são os exemplos mais comuns usados para suprir tal necessidade. Entretanto, antes que tais mecanismos possam ser aplicados, faz-se necessária a detecção de um comprometimento. O protocolo *Certificate Transparency* é o principal exemplo de estratégia para o monitoramento de emissão errônea de certificados. Sistemas que utilizam da infraestrutura de chaves públicas são obrigados, inevitavelmente, a empregar soluções de revogação e monitoramento que levam ao aumento de complexidade de suas aplicações. Ademais, a maioria destes mecanismos foram desenvolvidos com protocolos como *TLS* e operadores qualificados em mente; levando a um cenário onde a transparência do usuário final é depreciada. Argumenta-se que sistemas tradicionais não dão o foco necessário para usuários leigos e suas necessidades em relação a assinatura de documentos digitais, prejudicando uma maior participação desta parte da população. Este trabalho contribui com esta problema em duas maneiras: (i) propõe-se um novo modelo de gerenciamento de chaves privadas construído sobre certificados irrevogáveis de longa duração, cada qual ligado a apenas um documento. O modelo emite um certificado digital único para cada assinatura a ser feita. Demonstra-se que chaves privadas associadas a estes certificados podem ser destruídas após cada assinatura, eliminando-se a necessidade de dispositivos para sua guarda. Mostra-se que tais certificados não requerem mecanismos de revogação nem de marcas temporais para serem válidos por longos períodos de tempo. Além disso, analisa-se as consequências sobre a performance do modelo, seu impacto na privacidade e segurança, e o compara com o modelo tradicional, e (ii) modifica-se a estrutura do protocolo *Certificate Transparency* para suportar o monitoramento de documentos assinador por usuário sem prejudicar a transparência inerente ao sistema através do registro de certificados digitais atrelados a um resumo criptográfico, simultaneamente protegendo documentos contra ataques de pré-datação e habilitando o uso seguro de esquemas de assinatura de uso único. Demonstra-se a viabilidade da proposta através de duas implementações, uma usando a plataforma *Hyperledger* e a outra usando *Ethereum*, analisando-se os custos de implantação.

Palavras-chave: Certificado digital. Infraestrutura de chaves públicas. Certificate Transparency.

RESUMO EXPANDIDO

INTRODUÇÃO

Graças aos avanços de computadores pessoais e da alta disponibilidade da internet, documentos em papel estão lentamente sendo substituídos por sua contrapartida digital. Ao contrário de documentos em papel, onde a garantia de integridade é dada pela ausência de rasuras no documento original, um documento digital requer de meios eletrônicos para prover tal garantia. Por exemplo, da mesma maneira que documentos em papel podem ser assinados por uma caneta, documentos digitais podem ser assinados com o auxílio de computadores, mantendo propriedades jurídicas similares às da assinatura de próprio punho. Esta assinatura digital é criada através da adição de informações que simultaneamente identificam o assinante e os associam com o conteúdo do documento.

Tradicionalmente, um esquema de assinatura digital é composto pelo cálculo do resumo criptográfico do documento e pelo subsequente uso de esquemas de criptografia assimétrica parametrizados pela chave privada do assinante, o que representa a sua assinatura digital. O uso adequado e seguro dos algoritmos e das chaves privadas é de grande importância e podem ser complexos. Documentos podem conter várias assinaturas e enviados a múltiplas outras entidades que devem ser capazes de reconhecer os artefatos criptográficos presentes para que consigam confiar, ou não, no documento assinado. Portanto a criação de ferramentas e ambientes propícios para tais funções se faz necessário. Um modelo popular que provém esses serviços é a infraestrutura de chaves públicas (ICP).

A confiança no gerenciamento de chaves privadas é um dos maior desafios do modelo de ICP. Chaves privadas são protegidas por contâiners criptográficos que contém mecanismos robustos de controle de acesso. Tais preocupações de gerenciamento de chaves são exacerbadas quando usuários leigos fazer parte do sistema. Modelos tradicionais requerem que seus usuários tenham algum conhecimento sobre a responsabilidade de se manter um chave privada segura e sobre o que fazer quando um possível comprometimento ocorrer. Resulta-se que usuários finais geralmente utilizam dispositivos mais simples como *smartcards* ou *tokens* criptográficos. Adicionalmente, uma alternativa popular é o gerenciamento de chaves em nuvem, entretanto existem preocupações quanto a privacidade garantida por tais sistemas (MUSHTAQ et al., 2017).

Outro desafio recorrente em ICPs é a revogação de certificados digitais. Revogação se faz necessária por conta de certificados com atributos de identidade desatualizados e em virtude da possibilidade do comprometimento de chaves privadas. Não existem soluções infalíveis para a revogação de certificados (ÅRNES, 2000). Devido a estes desafios, argumenta-se que a ICP tradicional não é suficientemente inclusiva, tanto por uma perspectiva de conhecimento necessário, quanto por uma perspectiva de usabilidade. Requer-se do usuário conhecimento sobre a funcionamento interno da ICP, importância de gerenciamento correto de chaves privadas, e dos processos de revogação. Para usuários leigos, são requerimentos excessivos; em uma ICP ideal, todos os usuários devem ser capazes de assinar documentos de forma transparente.

Mesmo com todas as precauções tomadas por Autoridades Certificadoras (AC), elas ainda estão suscetíveis a ataques cibernéticos. Por exemplo, o incidente DigiNotar mostrou que erros de processos podem levar a emissão errôneas de certificados (COMODO, 2011; MEULEN, 2013) e que a sua rápida detecção é de extrema importância. Enquanto ACs raizes e intermediárias são consideradas seguras o suficiente para que tal monitoramento não seja crítico, ACs finais, aquelas responsáveis por emitir certificados para usuários finais, são mais facilmente comprometidas (MEULEN, 2013). O protocolo chamado *Certificate Transparency* (LAURIE et al., 2021) foi desenvolvido para esse monitoramento. Outros sistemas de monitoramento são geralmente baseados neste protocolo.

Entretanto, mesmo com o monitoramento de certificados por ACs sendo tratado de alguma forma, usuários finais ainda devem lidar com o mesmo problema de monitoramento e detecção que ameaçavam ACs. Isto é, usuários ainda tem o mesmo problema de detecção rápida caso sua chave privada seja comprometida. A detecção de documentos assinados erroneamente é tão imprescindível quanto a emissão equívoca de certificados por ACs. Adicionalmente, o problema de detecção se torna ainda mais difícil ao considerar-se usuários finais leigos. É comum que a responsabilidade seja transferida inteiramente ao usuário final por meio de documentos legais, ondo o usuário se responsabiliza pela segurança da chave e que ela jamais será compartilhada. Entretanto, é comum tais chaves e contêineres serem compartilhados com contadores e secretários.

Um agravante para os problemas citados anteriormente é a inevitável existência de computadores quânticos. Tais computadores demandam outros métodos de assinatura digital chamados de esquemas pós-quânticos. Geralmente, algoritmos pós-quânticos trazem consigo consequências performáticas. Algoritmos pós-quânticos geralmente pecam significativamente em tempo de execução das suas operações ou no tamanho de seus artefatos. Esquemas de assinatura uso único conseguem atingir resultados satisfatórios em tempo e tamanho quando comparados a maioria dos algoritmos pós-quânticos, entretanto, só conseguem fazer uma única assinatura por par de chave sem comprometer a segurança do esquema.

Assinaturas baseadas em resumo dependem exclusivamente na resistência de pré-imagem de seu componente de resumo criptográfico e provém um sistema para a construção de esquemas de assinatura de uso único (BUCHMANN et al., 2011). A sua característica de uso único de chave privada é uma restrição significativa para sistemas de ICP tradicional, entretanto, a destruição do par de chaves livra o usuário de qualquer gerenciamento posterior, desde que seja garantida a sua deleção. Argumenta-se que um sistema que consiga utilizar de forma segura esquemas de assinaturas de uso único, consegue melhorar a usabilidade intrínseca de sua ICP.

Contribuição

Propõe-se duas modificações ao modelo tradicional de ICP que podem ser usadas em conjunto ou por si só a fim de melhorar certas características de sistemas de assinatura de documentos digitais baseados em certificados. A primeira proposta, chamada de OTSCHAIN (MAYR et al., 2023) consiste no registro e monitoramento de uma tupla contendo o certificado de usuário final e o resumo criptográfico do documento associado ao documento assinado através de um *distributed ledger*. Este registro é feito exclusivamente por ACs a fim de não prejudicar a transparência do sistema para o usuário final. Uma consequência direta deste modelo é o requerimento de uma ligação entre emissão de certificado de usuário final e documento a ser assinado, ou seja, cada certificado é atrelado unicamente a apenas um documento. Analisa-se o modelo usando um esquema de assinaturas digitais de uso único baseados em resumo criptográfico.

Através deste registro, consegue-se habilitar o monitoramento de emissão de certificado, como é feito em modelos baseados no *Certificate Transparency*, mas também é possível monitorar assinaturas feitas por usuários finais com variados níveis de granularidade. Habilita-se então o monitoramento de comprometimento de credenciais e chaves privadas de usuários finais ao requerer-se que as tuplas correspondentes estejam registradas antes do arquivo assinado ser aceito. Adicionalmente, nossa proposta é capaz de prevenir o múltiplo uso de chaves privadas de uso único, recusando-se a registrar chaves públicas previamente registradas através de contratos inteligentes. Portanto, habilita-se o uso seguro de esquemas de assinatura de uso único. O Capítulo 4 discute as implicações de privacidade e segurança, implementações em sistemas de *blockchain* públicos e privados, e analisa requerimentos de performance e armazenamento considerando ambientes realistas de ICP.

Ademais, constrói-se sobre a ideia de ligação única entre certificado e documento e propõem-se um novo tipo de certificado digital onde a ligação feita através da tupla registrada na *blockchain* na proposta anterior é realizada dentro do próprio certificado. Este modelo de certificação digital resulta em um modelo de gerenciamento de chaves privadas que fornece uma ICP mais amigável e inclusiva ao usuário final para o propósito de assinatura de documentos digitais. Foca-se na emissão e revogação de certificados digitais e na criação e validação de documentos assinados digitalmente. Propõe-se a criação de certificados digitais irrevogáveis únicos chamados de *Certificados de Assinatura Única*. O Capítulo 3 mostra que estes certificados resolvem o problema de gerenciamento de chave, revogação, e usabilidade em nível de usuário final. Compara-se esta proposta contra o modelo tradicional de ICP, avaliando-se performance e segurança.

Escopo

Este trabalho foca no ambiente de ICP e nos processos necessários para a assinatura de documentos digitais com foco em transparência e usabilidade. Isto é, nosso objetivo é a criação de um modelo de ICP altamente difundida e inclusiva onde usuário conseguem assinar documentos da forma mais direta possível. Enquanto nossas contribuições podem ter outros casos de uso, estes não serão endereçados fora da seção de trabalhos futuros da conclusão. Adicionalmente, nota-se que este trabalho se limita ao uso de criptografia assimétrica para assinatura digital apenas, logo que a destruição da chave privada impossibilita o seu uso para decifrar mensagens cifradas usando a sua chave pública correspondente.

OBJETIVOS

Nosso principal objetivo é a simplificação da ICP para que muitos dos desafios comuns sejam ou aliviados ou resolvidos; particularmente gerenciamento de chaves privadas, revogação de certificados digitais, e monitoramento de emissão errônea de certificados e assinaturas digitais. Para isto, foi conduzido um estudo detalhado de propostas existentes que lidam com tais problemas. Os objetivos específicos estão detalhados abaixo:

- Diminuir os requerimentos de gerenciamento de chave privada em nível de usuário final;
- Diminuir os requerimentos de revogação em nível de usuário final;
- Permitir o monitoramento de assinatura de documentos de forma transparente;
- Aumentar a transparência da ICP;
- Discutir os impactos das propostas nos quesitos de performance e privacidade;
- Medir e comparar os impactos das propostas com o sistema tradicional.

METODOLOGIA

Alcança-se os objetivos acima através de uma revisão sistemática da literatura e de uma análise prévia de características indesejáveis pertencentes a ICP. As consultas utilizadas são derivadas da pergunta de pesquisa sobre o aumento de usabilidade e simplificação dos processos inerentes a uma infraestrutura de chave pública. Os resultados são então filtrados e expendidos dependendo da relevância quanto ao tema. Posteriormente, as propostas são analisadas quanto a sua segurança e sua viabilidade prática quando comparadas com o modelo tradicional.

RESULTADOS E DISCUSSÃO

Mantém-se simultaneamente os requerimentos usuais de políticas de assinatura para documentos assinados usando certificados de assinatura única e melhora-se a performance de processos de validação de documentos assinados digitalmente. A troca entre usabilidade e performance durante o processo de assinatura pode ser aliviado através de escolhas inteligentes de algoritmos de assinatura e estratégias de geração de par de chaves. Estes recursos permitem a criação de uma ICP simples, eficiente, e inclusiva, criada para a rápida e abrangente adoção de assinatura de documentos digitais por pessoas leigas, mantendo-se recursos de usabilidade e segurança robustos.

A existência da possibilidade de comprometimento de chave ou de credenciais de usuários finais faz-se necessária a adoção de mecanismos de monitoramento de tais certificados e documentos. Atende-se a essas necessidades de forma célere através da proposta de

monitoramento em *distributed ledger* OTSCHAIN. Define-se uma estrutura que expande o protocolo *Certificate Transparency* através de registros em *blockchain*, mantendo todas suas funções nativas e habilitando o monitoramento granular de assinaturas feitas por usuário finais, permitindo a detecção de comprometimentos de sua chave privada ou credenciais. Este processo é transparente ao usuário pois a responsabilidade de registro é movida para a AC. Habilita-se o uso seguro de algoritmos de assinatura única através de verificações semânticas contidas nos contratos inteligentes das plataformas. Discute-se os impactos em segurança e privacidade destes registros e mostra-se que sua viabilidade é compatível com implementações atuais de ICP.

CONSIDERAÇÕES FINAIS

Gerenciamento de chaves privadas, revogação de certificados digitais e a detecção de comprometimentos são fontes de complexidade em ICPs desde seu nascimento. Nesta dissertação, analisa-se alguns dos problemas usuais com o medolo tradicional de ICP para a assinatura de documentos digitais. Além disso introduz-se um novo modelo de gerência de chaves que é capaz de resolver estes problemas em nível de usuário final através da ligação única entre usuário, certificado digital e documento. Mostra-se que vários artefatos responsáveis pelo aumento de complexidade da ICP não são mais necessários e que desafios de segurança associados com chaves privadas e atributos de identidade desatualizados são prevenidos através do nosso modelo.

Palavras-chave: Certificado digital. Infraestrutura de chaves públicas. Transparência de certificados.

ABSTRACT

Electronic documents are signed using private keys and verified using the corresponding digital certificates through the well-known public key infrastructure model. One of the major dangers of using such an infrastructure is the possible compromise of the private key, which in turn makes key management a critical component of public key infrastructures. Thus, several services and mechanisms were created to aid in guaranteeing that private keys are safe, from keeping the key itself secure in a container to mechanisms that enable the revocation of certificates. Alas, there is no all-encompassing failproof solution to the issues a compromised private key can bring. Furthermore, most of these services and mechanisms were developed either with trained personnel or computer systems in mind. Consequently, we argue that traditional systems do not focus enough on laypeople and day-to-day operations, harming the large-scale adoptability of digital signatures of electronic documents by laypeople. We tackle the issue of revocation, private key management, and monitoring while keeping usability in the forefront through two independent but synergistic contributions: (i) We propose a new cryptographic key management model built with long-term, irrevocable digital certificates, each bound to a single document via the issuance of a unique digital certificate for each new document to be signed. We demonstrate that private keys associated with these certificates do not need to be generated nor stored in secure environments. Furthermore, we show that these certificates do not require any revocation mechanism or timestamps to guarantee long-term verification. Lastly, we analyze the performance implications of our model, provide a security overview, and compare it to the traditional model; and (ii) we modify the CT framework to handle signed documents via the logging of certificates in the blockchain to enable the secure and user-friendly monitoring of one-time signatures, backdating protection, and effective CA misbehavior detection. Moreover, to demonstrate the feasibility of our proposal, we present distinct deployment scenarios and analyze the storage, performance, and monetary costs. Furthermore, under this framework, we prevent reused one-time signature scheme private keys from being accepted, thus enabling the secure monitoring of hash-based signature schemes.

Keywords: Digital certificate. Public key infrastructure. Certificate Transparency.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Since the adoption of paper as a way of recording communications, people have drawn their names or symbols representing their agreement with the contents of the document. Any interested stakeholder who recognizes the graphical inscriptions of the signatories will understand their commitment to those contents. Security properties such as integrity and authenticity of paper documents are achieved through physical elements, such as the paper itself, the pen ink, or the stamp mark of the signature. If required, new signatures are added from trusted third parties, such as notaries, to certify that previous signatures are genuine.

Thanks to the advent of digital computers, paper documents are slowly being replaced by their electronic counterparts. The integrity of the content of paper documents is guaranteed by the absence of erasures in what was supposedly recorded on paper. Moreover, in the same way that paper can be signed with a pen, digital documents may also be signed with computer systems, maintaining similar properties. A digital signature is created by adding information that simultaneously identifies the signers and associates them with the contents of the document.

The most straightforward digital signature scheme consists of determining the cryptographic hash of the document and then using an asymmetric cryptosystem parameterized by the private key of the signer to produce the information representing the signature. Although seemingly simple, the proper and secure use of keys and cryptographic algorithms is anything but simple. Documents may have multiple signatures and be forwarded to other parties, who must recognize the cryptographic artifacts to verify and acknowledge the signatures. Establishing a framework to organize and standardize such services is necessary. The model that provides these services is called *public key infrastructure* (PKI).

Trusting cryptographic key management is one of the main challenges of the PKI model. Private keys are protected by cryptographic hardware or software containers, which have several access control mechanisms. Security concerns are amplified when dealing with laypeople. Traditional models require users to understand the responsibilities of possessing a private key and the expected course of action if it is compromised. As a result, signers use simpler, relatively low-cost cryptographic devices such as smartcards or tokens. In addition, cloud-based key management services eliminate the need for users to carry a device, but using such providers raises privacy concerns (MUSHTAQ et al., 2017).

Another recurring challenge in PKIs is the revocation of digital certificates. Revocation is necessary to account for outdated certificate attributes and compromised private keys. There is no failproof solution to certificate revocation (ÅRNES, 2000). Due to these complications, we argue that traditional PKIs are not inclusive, both from a usability and knowledge point of view. On the usability side, it requires users to understand how the PKI model works, its inherent complexities, the importance of good key management, and revocation processes. For laypeople, this is an unreasonable requirement; all citizens should be able to sign electronic documents at their convenience.

Even with all these precautions, CAs are still susceptible to cyberattacks. For example, the DigiNotar incident showed that a process failure led to the fraudulent issuance of certificates (COMODO, 2011; MEULEN, 2013). Such incidents showed the pressing need for a system that could monitor certificate issuance even from trusted CAs. Typically, root and intermediate certification authorities (CAs) are assumed to be secure enough such that this monitoring is unnecessary. Final CAs, those that issue end-user certificates, on the other hand, are known to be more easily compromised (MEULEN, 2013). Hence, there exists a need to monitor certificate issuance by final CAs. Systems that deal with the monitoring of issued certificates are usually modeled after the well-known Certificate Transparency (CT) protocol (LAURIE et al., 2021).

However, even if the misissuance of certificates issued by final CAs is somewhat addressed, end-users still face the same problems faced by these CAs; before a private key compromise can be handled, it is imperative that it is detected. Nevertheless, the issue of private key misuse detection becomes considerably harder when considering that most end users are laypeople in a widespread PKI environment. Typically, this hurdle is negligently left in the hands of the user; responsibility is usually transferred solely to the end user through legal documents which state the user is responsible for the protection of their private key. In practice, however, private keys are shared between users and their accountants, secretaries, and with each other.

Moreover, these issues are aggravated by the approaching quantum computer. Post-quantum algorithms negatively impact many aspects of digital signatures and, by extension, digital certification and PKI. The most common consequences among the issues brought by the adoption of post-quantum algorithms pertain to performance; either keys or signatures are significantly larger, or the algorithms take considerably longer to compute. Conversely, one-time signature schemes such as hash-based signatures seem to break from this mold with another downside; they can only be used to generate a single signature.

Hash-based signatures rely exclusively on the pre-image resistance of its component cryptographic hash function and provide a framework to construct one-time signature schemes (OTSs) (BUCHMANN et al., 2011). It is strongly recommended that key pairs from OTS schemes be used to generate a single signature since they are derived directly from a private key. In the context of a PKI framework, this implies that a private key need not be stored after being used once. Thus, we argue that OTS schemes are an important asset in solving private key management issues, relieving the end user from holding sensitive information and consequently improving usability.

1.1 CONTRIBUTION

We propose two distinct modifications to the traditional PKI model that can be used either by themselves or in tandem to improve the characteristics of certificate-based digitally signed documents. The first, named OTSCHAIN (MAYR et al., 2023), consists of logging end-user certificates and hashes associated with digitally signed documents in a distributed ledger; this logging is made by CAs to relieve laypeople of such responsibilities, increasing the usability of end-user certificate holders. A direct consequence of this model is the requirement of a binding between certificate issuance and document, i.e., a digital certificate issued uniquely to each digitally signed file. This is done via the creation of new key pairs for each signed document or simply through the use of OTS schemes. Such bindings are then registered in the blockchain.

Moreover, we achieve effective monitoring of digital certificate issuance and creation of digitally signed documents, detecting possible malfeasance from CAs and/or compromise of user credentials and preventing backdating of digital signatures. Additionally, our strategy is quantum-resistant, considering a suitable choice of signature schemes throughout CAs, end-users, and distributed ledgers. The distributed ledger technique prevents effective private key reuse, solving the largest practical restriction of an OTS scheme. Chapter 4 discusses the privacy and security implications of this proposal, presents public and private blockchain models as alternative implementations, and analyzes storage and performance requirements considering a realistic PKI scenario.

Furthermore, we build upon the binding requirements introduced by OTSCHAIN and propose a new type of digital certificate where the binding from the blockchain is moved to the certificate itself. These certificates result in a simpler key management model that yields a more inclusive and user-friendly PKI for the purposes of digitally signing documents. We focus on the issuance and revocation of certificates and the creation and validation of digitally signed documents. We propose irrevocable, unique certificates called One-Time Certificates (OTCs) (MAYR et al., 2024). We show that OTCs solve the problem of key management, revocation, and usability at the signer level. We compare our proposal with the traditional public key infrastructure model, evaluating performance and security.

1.2 COLLABORATIONS

Our team was approached by many different agencies with needs corresponding with the goals of our research; the creation of a widespread environment for the signing of digital documents. The traditional model was unfit for their needs; either it required too much knowledge about the processes that happen inside a PKI, would have prohibitive costs when faced with the sheer number of participants it would have to answer to, or usually both. From these agencies, we signed cooperation agreements with the three that most aligned with our objectives: (i) *Registro Civil do Brasil de Pessoas Naturais*, the agency responsible for the Brazilian civil registry of natural persons, which required the PKI to be transparent when asking citizens to sign digital documents; the traditional model was too burdensome for most citizens; (ii) *Rede Nacional de Ensino e Pesquisa*, responsible for many of the services used by federal and state universities in Brazil; it required a widespread solution that would be able to provide digital signatures for the public universities of Brazil in a seamless fashion while being integrated to their existing systems; and (iii) *Instituto Nacional de Tecnologias de Informação e Comunicação* from Mozambique; responsible for the regulation, supervision, and auditing of the communication and information technology sector in Mozambique; which required a simple and broad digital signature solution to be deployed nation-wide.

We are glad to have been able to collaborate with such important agencies and happy to see the fruits of our work bear fruit; *Registro Civil do Brasil de Pessoas Naturais* was able to create their own PKI environment and accompanying legislation, *Registro Civil do Brasil de Pessoas Naturais* is soon to move from the pilot deployment to a production environment, ready to supply Brazilian universities with a free digital signature system, and Mozambique launched their own state PKI environment containing both a traditional certificate chain and an OTC one.

1.3 SCOPE

This work focuses on the PKI environment and the processes necessary to digitally sign and verify digital documents while aiming to maximize transparency and reach. i.e., our goal is to create a widespread and inclusive PKI model where end users are able to sign documents in the most seamless way possible. While our contributions may have other uses outside of digitally signed documents by laypeople, we do not explore these possibilities outside of the future works section in our conclusions. Furthermore, we only concern ourselves with digital signatures; we remark that asymmetric cryptography for the purposes of ciphering data using the public and deciphering it with its private counterpart is made impossible when deleting the private key.

1.4 OBJECTIVE

We aim to simplify PKI so that many usual challenges can be either eased or resolved, particularly private key management, certificate revocation, and certificate misissuance detection. To support this goal, we wish to conduct a detailed study of existing proposals that deal with these issues. Furthermore, the specific objectives are detailed below:

- Relax key management requirements for the user;
- Relax revocation requirements for the end user;
- Enable user document signature monitoring;
- Increase overall PKI transparency;
- Discuss the impacts on security and privacy of our proposals;
- Measure the impact of our proposal against traditional models.

1.5 METHODOLOGY

In this section, we describe the research strategy applied to produce this work and briefly discuss some articles that touch upon PKI and its issues; the related work section is expanded in each contribution section individually for clarity. Furthermore, we recall that the object of this study is PKI modifications that simplify its management, deployment, and user requirements.

We inspect formal works such as journals, conference articles and monographs existing in the scientific literature. We employed the Google Scholar web tool, a meta-indexer of scientific literature that allows advanced queries to help narrow the scope of our searches and used the following query: *((pki or public key infrastructure) or (digital certificate)) and (revocation or (key management) or irrevocable or simple)*

A large amount of results did not modify PKI directly and were instead simply using it. The mention of PKI made that our query would match them with our theme. In reality, few papers dealt with changes to the infrastructure itself. Papers were filtered based on title and abstract, and those selected were read in full.

1.5.1 Related Work

Most results cite revocation as one of the biggest challenges for PKIs. Notably, the original document detailing digital certificates warned that it did not behave well when the compromised key was concerned (KOHNFELDER, 1978). In 1998, Rivest would suggest that the best way to deal with revocation was to bypass it somehow (RIVEST, 1998).

Gutmann states that revocation can not work in the way designed by the original proposal (GUTMANN, 2002). Certificates would be in an unknown state between CRL periods, their validity only truly verifiable after the CRL has been published. They also point out that the other most common method of revocation, the Online Certificate Status Protocol, also has its glaring flaws (GUTMANN, 2002; WOHLMACHER, 2000). There have been many different proposals to solve the revocation problem. However, there are no fail-proof solutions (ÅRNES, 2000; CERENIUS et al., 2024).

Irrevocable certificates can be seen in short-lived certificates (TOPALOVIC et al., 2012). These certificates have a shorter lifespan, similar to the period of a CRL. If the private key is compromised, the certificate authority simply does not issue new certificates for the same key pair, eliminating the need for any revocation method (TOPALOVIC et al., 2012). The Simple Distributed Security Infrastructure (SDSI) operates in a similar way (RIVEST; LAMPSON, 1996). An unwanted side effect of this type of certificate is that signatures need to be reconfirmed periodically.

Key management papers were rather scarce when compared to ones dealing with revocation. It is usually assumed that the user keeps their private key safe (BOEYEN et al., 2008; HELMICH, 2000). Gutmann cites some alternatives to key management that forgo PKI as a whole (GUTMANN, 2004) and claims that the adoption of those systems is due to the ease of use for the end user. Key management as a Service (KMaaS) is an alternative method of storing and using key pairs that have become popular (LERMAN; MARKOWITCH; JR, 2012; KUZMINYKH; GHITA; SHIAELES, 2020; CHONG; KIM; CHOI, 2021) for allowing users to forgo hardware solutions like smart cards and tokens. Questions about the security of KMaaS were raised, citing the inherent lack of confidentiality of storing anything in the cloud (DAMGåRD et al., 2013; PADILHA; PEDONE, 2015; KUMAR; BHATIA, 2020).

Concerns were raised on the future of PKI when post-quantum algorithms were considered. (YUNAKOVSKY et al., 2021) discusses the impacts of quantum computers on existing PKI, comparing algorithms and making suggestions on the transitioning process. These post-quantum algorithms bring new problems to PKI and digital certificates (KAMPANAKIS et al., 2018). Proposals that aim to change PKI should also consider how they would fare in a quantum environment.

As previously mentioned, private key management and revocation account for great complexity in the PKI model. It raises costs and is therefore considered the main cause that prevents the widespread use of PKI in communication networks (Ponemon Institute, 2021) and in the most diverse applications that could benefit from its services. This has motivated the search for ways to simplify, if not all, then at least parts of the PKI. We briefly discuss works related to simplifying PKI mechanisms and how they affect usability and technology adoption.

Rivest and Topalovic et al. introduced short-lived digital certificates (RIVEST, 1998; TOPALOVIC et al., 2012) to remove the need for revocation at the signer level, issuing them with a very short validity period. They argue that if the certificate validity period were less than the latency of traditional revocation systems, then there would be no reason to revoke certificates. One of the main criticisms of this proposal is the increase in complexity on the side of the certificate holder. Furthermore, *certificate authorities* (CAs) should always be available to issue new certificates on demand.

Simple Public Key Infrastructure (RIVEST, 1998) (SPKI) and Simple Distributed Security Infrastructure (RIVEST; LAMPSON, 1996) (SDSI) are other instances of revocation-less PKIs. These models associate a certificate with a recency level. Thus, an entity may request a certificate with a particular "freshness", i.e., a certificate guaranteed to be issued after a certain moment. While the models do not allow certificate revocation, long-term certificates still have the same problems as in a traditional PKI (ÅRNES, 2000).

The Automatic Certificate Management Environment protocol (BARNES et al., 2019) is responsible for the increase in the adoption rate of the TLS protocol (AAS et al., 2019) through simplification and automation of the issuance of HTTPS certificates. By 2020, approximately 81% of HTTPS websites were using Let's Encrypt certificates (AAS; GRAN, 2020). Its success can also be attributed to the free issuance of the certificates. This effort is mentioned to exemplify how high-security standards can be rapidly embraced due to changes in a previously convoluted process.

The "certificateless" proposal (AL-RIYAMI; PATERSON, 2003) radically changes how PKIs are built; certificates are no longer used to transport keys and attributes. Other derived models (GUTMANN, 2002; GUTMANN, 2004; SCHEIBELHOFER, 2005; TOPALOVIC et al., 2012) introduce several trade-offs in exchange for features such as the absence of revocation. Particularly, there is an increase in the complexity of cryptographic key management in these models, among numerous other challenges. Furthermore, these proposals are not interoperable and backward-compatible with existing computer systems.

We note that there is a particular focus on solving the issue of revocation and that the private key is assumed to be operated by a knowledgeable holder. Furthermore, such issues are primarily approached from the perspective of the TLS protocol. We argue that there is a distinct lack of research from the perspective of the general public when signing and verifying digitally signed documents. We aim to solve this issue via the proposal of the OTC model, which enables secure, long-term, and revocation-less digital certificates. In our model, the underlying PKI is made completely transparent to the end user.

While the issue of certificate misissuance appeared in our previous queries, it was usually just in passing statements. Thus, further queries were made to narrow down solutions to the particular issue of certificate transparency and its interactions with post-quantum algorithms, more specifically, one-time signature schemes. Therefore, we included the following queries in our research: (i) *("blockchain" OR "distributed ledger") AND ("one-time signature" OR "winternitz" OR "xmss")*; (ii) *("certificate transparency") AND ("one-time signature" OR "winternitz" OR "xmss")*; and (iii) *("certificate transparency") AND ("blockchain" OR "distributed ledger")*. We note that many implementations of certificate transparency have been proposed, but the overall procedures remain roughly the same.

Blockchain implementations of the CT protocol have been proposed with some improvements. Madala et al. (MADALA; JHANWAR; CHATTOPADHYAY, 2018) introduce an implementation of CT where CAs must receive consent from the domain owner before a certificate can be issued. Their implementation is able to revoke certificates in the blockchain. In a similar manner, BB-PKI (GARBA et al., 2020) requires that registration authorities (RAs) reach a consensus on the issuance of a certificate, which is then signed by multiple CAs through an out-of-band channel to lower the chances of successful impersonation attacks against RAs.

Blockchain-based PKIs have been proposed as an alternative to the traditional model. In these proposals, CT is assumed, as most PKI artifacts are committed to the blockchain as part of the process of issuing certificates. However, most proposals simply replicate the same procedures of a traditional PKI, opting to simply log its operations without much disruption to usual procedures. However, some proposals create conditions for the issuance of digital certificates. Kubilay et al. (KUBILAY; KIRAZ; MANTAR, 2021) require that a consensus be reached through a dynamic threshold signature scheme before a certificate is issued. Another PKI implementation that aims at blocking certificate misissuance before it happens through the blockchain is the Pistis (LI et al., 2022) framework. In this framework, certificate issuance is done through the blockchain only when domain ownership is verified.

Some proposals include misissuance prevention as a feature of signature verification applications. In other words, verifiers can set conditions for when a signature can be deemed valid. For instance, Han and Hwang (HAN; HWANG, 2020) introduce a trust model dubbed "conditional trust". Certificates with many trusted signatures are seen as more trustworthy than those with fewer signatures. This proposal utilizes the blockchain to communicate among many CAs and RAs. However, an important observation of models that embed many signatures in a certificate is that it may grow indefinitely with time.

The intersection of blockchain and OTS limits itself to using such schemes to improve aspects of blockchain as a replacement for classical digital signature algorithms. Most works set hash-based algorithms as the de facto OTS scheme. Vives (VIVES, 2017) improves efficiency by replacing ECDSA signatures with a single hash-based signature to enable post-quantum authentication in the blockchain. Hyla and Pejaś (HYLA; PEJAś, 2020) show how blockchains can create a beneficial environment for the archival of digital documents by acting as a timestamp. They propose the usage of XMSS to increase blockchain longevity in the post-quantum scenario.

The Key Compromise Resilient Signature (XU et al., 2019) (KCRS) system includes the end-user signature in the list of PKI artifacts it logs in the blockchain. In this scheme, users have multiple key pairs, a master key pair certified by a digital certificate, and multiple "signing key pairs". The user derives the signing key pairs from the master key to sign symmetric-key encrypted messages sent through an out-of-band secure channel by the verifier and log the resulting signature in the blockchain. Thus, only the verifier and the signer know that the end-user has generated a signature. This scheme loses the monitoring qualities of CT by weakening the link between the user and the asymmetric key pair.

To the best of our knowledge, there are no published works that deal with private key reuse of OTS schemes by limiting the number of signed documents that can be validated. We note that most proposals draw heavily from the CT protocol and consequently focus on the perspective of TLS certificates. Research on the impact of post-quantum algorithms, certificate logging and monitoring, and usability for the signing of digital documents by laypeople is severely lacking.

Hypothesis. It follows from the analysis above that to the best of our knowledge, no works in the literature managed to either create long-term irrevocable certificates, safely remove revocation without drastically shortening certificate lifespan, or improve certificate transparency in meaningful ways to handle a post-quantum environment. Furthermore, there is little to no focus on the subject of digital signatures for laypeople. This is partially due to the challenge in detecting such breaches and the limitless nature of signature generation and digital certificates; a user can sign an unlimited number of documents until their certificate expires. These characteristics lead to strict private key management requirements and an increase in the associated dangers of its compromise, which leads either to the drastic shortening of the certificate lifespan or to some kind of revocation method. That is, the need for key management and revocation is linked directly to the dangers a compromised private key poses to the system. A PKI model that can lessen the impact of compromised private keys could relax key management and revocation constraints to the point of removal.

1.6 STRUCTURE

In Chapter 2, we present the necessary background for the comprehension of this work and discuss criticisms of the classic PKI model. Chapter 4 presents OTSCHAIN and discusses the benefits it brings to certificate misissuance detection and its impacts in regard to user privacy and blockchain size requirements. In Chapter 3, we present One-Time Certificates, compare them to the traditional model, measure their efficiency, and compare them to the traditional model. Lastly, Chapter 5 presents our conclusions and future works.

2 BACKGROUND

In this section, we present the cryptographic primitives, artifacts, and services needed to understand our proposals. We assume the reader to be comfortable with the general aspects of digital signature schemes and PKI.

2.1 CRYPTOGRAPHIC HASH FUNCTIONS

Such functions are used to detect changes to given inputs, for instance, due to a noisy channel or malicious alterations in transit. We note that the hash of a message may also be called a *digest* or *fingerprint*.

Definition 1 ((MENEZES; OORSCHOT; VANSTONE, 1996, Sec. 9))**.** A function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ is a *cryptographic hash function* if it respects the following set of properties:

(i) *Performance*: it is easy to compute $\mathcal{H}(x)$.

(ii) *One-wayness*: it is computationally hard to find any x such that $\mathcal{H}(x) = y$, for some y picked at random from the range of $\mathcal{H}$.

(iii) *Pre-image resistance*: it should be computationally hard to find a original message m given its digest y, i.e., to find any m such that $\mathcal{H}(m) = y$ where y is known.

(iv) *Second pre-image resistance*: it should be computationally hard to, given a message m, find another distinct message m' that generates the same digest y; i.e., to find m' such that $\mathcal{H}(m) = \mathcal{H}(m')$ where m is known.

(v) *Collision resistance*: it should be computationally hard to find any two distinct messages that output the same digest, i.e., to find any m and m' such that $\mathcal{H}(m) = \mathcal{H}(m')$ and $m \neq m'$.

2.2 DIGITAL SIGNATURE SCHEMES

Digital signatures are classic examples of how public-key cryptography can be employed. Such algorithms provide three main properties: authentication, which is the validity of the identity information related to an entity; integrity, which is the guarantee that the signed information has not been modified in transit; and non-repudiation, which is the assurance that the signer cannot deny having signed a given message. Oftentimes, the actual message being signed is the digest of the original message. A formal definition is given below.

Definition 2 ((GOLDREICH, 2004))**.** Consider a security parameter $\lambda \in \mathbb{N}$ and any message $m \in \{0,1\}^*$. A *digital signature scheme* is a triple of probabilistic polynomial-time algorithms: (i) the *key generation* $\textsc{Gen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk}) \in (\{0,1\}^* \times \{0,1\}^*)$, which outputs private and public keys, bound by a predefined property; (ii) the *signature generation* $\textsc{Sig}(\mathsf{sk}, m) \to \sigma \in$

$\{0,1\}^*$; and (iii) the *signature verification* $\textsc{Ver}(\mathsf{pk}, m, \sigma) \to \{0,1\}$, all of which must satisfy $\Pr[\textsc{Ver}(\mathsf{pk}, m, \textsc{Sig}(\mathsf{sk}, m)) = 1] = 1$ for any $(\mathsf{sk}, \mathsf{pk})$.

We remark that while the above definition describes signature schemes as a whole, they can further be divided into two categories related to their resistance to attacks made by a quantum adversary: the subset of *classic schemes* which are heavily impacted by Shor's (SHOR, 1999) and Grover's (GROVER, 1996) algorithms and *post-quantum schemes*, which rely on mathematical constructions that are either mildly affected or outright unaffected by these algorithms.

2.3 WINTERNITZ SIGNATURE SCHEME

Winternitz is an OTS scheme whose signatures are obtained from multiple applications of a cryptographic hash function $\mathcal{H}$ over the private key. We give our definition of the scheme based on (DODS; SMART; STAM, 2005) for readability, but we observe that there exist variants with fewer security requirements (HüLSING; KUDINOV, 2022).

Consider $\lambda \in \mathbb{N}$ as the security parameter, $n \in \mathbb{N}$ to be the message length, a cryptographic hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$, any message $m \in \{0,1\}^n$, and $w \geq 1$ to be the Winternitz parameter, usually a power of two, which represents the number of bits to be signed in parallel. Let $t_1 = \lceil \frac{n}{\log_2 w} \rceil$, $t_2 = \lfloor \frac{\log_2 t_1 (w-1)}{\log_2 w} \rfloor + 1$ and $t = t_1 + t_2$. We define $f^i(x)$ to be $f(f(\ldots f(x) \ldots))$ applied i times, with $f^0(x) = x$. We set $\lambda = n$ for practical purposes. We present the algorithms that form the signature scheme below.

$\textsc{Gen}(1^\lambda)$. Generate $\lambda \cdot t$ random bits and set the private key to be t random words of equal size, i.e., $\mathsf{sk} = (x_{t-1}, \ldots, x_0)$. The public key is $\mathsf{pk} = (\mathcal{H}^{w-1}(x_{t-1}), \ldots, \mathcal{H}^{w-1}(x_0))$, or the result of repeated applications of the cryptographic hash function to each element of the private key.

$\textsc{Sig}(\mathsf{sk}, m)$. Take m and slice it into a tuple with t_1 elements of w bits $B_1 = (b_{t-1}, \ldots, b_{t-t_1})$. (Without loss of generality, we assume $w \mid n$, such that no padding is required.) Calculate $c = \sum_{i=1}^{t_1} (w - 1 - b_{t-i})$ and slice it into an analogous tuple with t_2 elements of w bits $B_2 = (b_{t_2-1}, \ldots, b_0)$. Finally, let $\sigma_i = \mathcal{H}^{b_i}(x_i)$ for $0 \leq i \leq t-1$, and the signature is $\sigma = (\sigma_{t-1}, \ldots, \sigma_0)$.

$\textsc{Ver}(\mathsf{pk}, m, \sigma)$. Obtain B_1 and B_2 from m as per the procedure above. Then, the signature is valid if $\mathsf{pk} = (\mathcal{H}^{w-1-b_{t-1}}(\sigma_{t-1}), \ldots, \mathcal{H}^{w-1-b_0}(\sigma_0))$, and invalid otherwise.

Sizes of public keys and signatures are $n \cdot t$. For public keys, the size can be reduced to n by hashing the concatenation of all public key elements in a predefined order.

2.4 PUBLIC KEY INFRASTRUCTURE

A *Public Key Infrastructure* (PKI) is a set of policies, special hardware, software, and rigid procedures that work together to provide cryptographic services to entities. The main elements of a PKI are the digital certificates, the *certificate authorities* (CAs), and the *registration authorities* (RAs). CAs are responsible for issuing and revoking certificates and for maintaining the accuracy and validity of the attributes. Other responsibilities may include authentication of the certificate holders and key pair generation support. Identity management procedures and the communication between holders and CAs are usually delegated to RAs.

In general, the certificate hierarchy is modeled after a tree. At the highest level is the *root* CA, and the leaves are end-user certificates. In-between, there may exist *intermediate* and *final* CAs. Root CAs are the entities from which trust is derived. Root and intermediate CAs issue certificates to subordinate CAs, and rarely to end users, while final CAs only issue end-user certificates.

Additionally, we observe that, with the migration of several PKI services to cloud-based solutions, the role of an RA closely resembles that of an *identity provider* (IdP). These providers manage user identity attributes and forward them to requesting clients. These properties allow the replacement of RAs with IdPs in the PKI model when there is a need for automation or when there is such a high load of certificate issuance requests that the identity management process cannot be overseen by humans.

Traditionally, the steps required to receive a digital certificate from a final CA are: (i) the user generates a key pair; (ii) the user generates a Certificate Signing Request (CSR); (iii) the user authenticates with the CA; (iv) the CA validates the user information; (v) the CA issues the end user certificate; (vi) the user receives the certificate. RAs or IdPs can act as middlemen and support the user and the CA in steps (i), (ii), (iii), (iv), and (vi).

Furthermore, CAs are required to issue their own revocation artifacts, typically through Certificate Revocation Lists (CRLs) (BOEYEN et al., 2008) or responding to Online Certificate Status Protocol (OCSP) queries (SANTESSON et al., 2013). CRLs contain the serial numbers of revoked certificates and are digitally signed by the CA that issued the original certificates or by some entity delegated by the CA. Each CRL is valid for a certain period of time, and a new CRL is issued as soon as the previous one expires. OCSP is an online service that responds to queries about the status of a given certificate. If revocation information about a certificate is needed, an OCSP request is sent to the OCSP server. The server checks the status of a certificate and returns a signed response with these contents.

Digital certificate

A digital certificate is a signed document that contains attributes that identify an entity and a public key corresponding to the private key held by that entity. Certificates may be signed by other holders: this is referred to as certificate *issuance*. This process creates a trust link between two holders. Commonly, a digital certificate contains metadata describing its type, intended use, validity period, and cryptographic algorithms. Documents are signed using a private key and verified using the corresponding public-key certificate; certificates cannot be used to verify any signatures after their expiration.

A certificate is issued at time T_i and has a validity period $\Delta\mathsf{T}_v$ that begins at T_b and ends at T_a. A certificate that has been revoked has its validity period reduced to $\Delta\mathsf{T}_c$, which is equal to $\mathsf{T}_r - \mathsf{T}_b$, where T_r is the time of revocation. Such reference times are summarized in Figure 1. We observe that, for every certificate, $\mathsf{T}_i \leq \mathsf{T}_b \leq \mathsf{T}_r \leq \mathsf{T}_a$; however, certificates typically follow $\mathsf{T}_i \leq \mathsf{T}_b < \mathsf{T}_r < \mathsf{T}_a$, as issuing an expired or revoked certificate has little to no practical use.

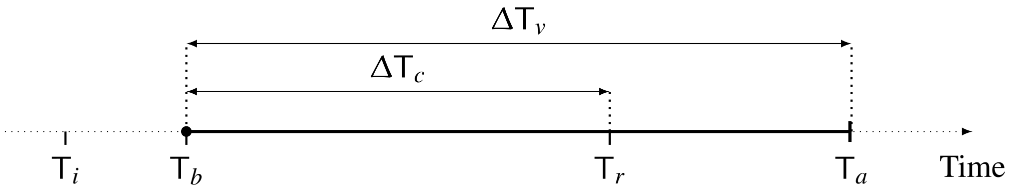

Figure 1 – The various time references for a digital certificate.

Signature policy

A policy is a set of technical and orderly criteria by which signatures are created and verified, each according to a particular set of requirements indicating which certificate extensions and artifacts must be included in the final signature for it to be correctly verified. Thus, before generating a signature, the user implicitly or explicitly selects the appropriate policy in relation to its intended use. For example, if a user wants to sign a short-lived non-critical resource for their company, they can choose a basic policy so that fewer artifacts will be needed to create a complete signature. However, suppose a document needs to be archived for future use. In that case, a policy must be chosen that foresees the incorporation of all the artifacts that the recipient will need to validate the signature.

Each policy establishes attributes that must be included in the signature. Some attributes must be signed by the end user, while others can be added by any party afterward, such as timestamps. Policies can be either implicitly or explicitly defined and are usually specified for several forms of electronic document representation. The most used formats are ASN.1, PDF, XML, and JSON. These are semantically equivalent, and most changes occur in the representation of the electronic document in the underlying language (ETSI, 2021b).

Advanced Electronic Signatures

In regulatory terms, electronic signatures that follow signature policies are called Advanced Electronic Signatures (AdES) (KUTYłOWSKI; BłAśKIEWICZ, 2023). Hereafter, we consider advanced signatures as defined in ETSI EN 319 102-1 (ETSI, 2021b). We argue that this set of European standards is far-reaching and robust enough to allow for mature comparisons with our proposal. The standards cover various use cases due to the needs of several countries and allow for the customization and interoperability of various signature formats. We compare our proposal against the CAdES (ETSI, 2021a) format, as the features of the ASN.1 language allow for straightforward parallels with other formats without loss of generality.

To generate an advanced electronic signature in the PKI model, it is not enough to simply create an individual signature σ. A certificate holder may include references to its certificate and those of its chain, as well as revocation artifacts and, optionally, timestamps. Furthermore, other end users can also include their own signed artifacts. Therefore, the signature verification process must be extended to include every such artifact in the final document. That is, a signed document with a timestamp must successfully validate both signatures for the document to be valid, i.e., it must verify the document, the timestamp, and both certification chains and corresponding revocation data. The total cost of verifying an advanced electronic signature is calculated below.

Let $t \in \mathbb{N}$ be the number of timestamps; for $1 \leq i \leq t+1$, let $c_i \in \mathbb{N}$ be the depth of the certificate chain of a signing certificate, i.e., the number of certificates from the end user to the root CA certificate, and $r_i \in \mathbb{N}$ the number of artifacts with revocation data. Then, the total number of signatures $s \in \mathbb{N}$ to be verified in any signed document is $s = \sum_{i=1}^{t+1}(1 + c_i + r_i)$.

Typically, certificate chains are three or four certificates deep. Note that every certificate in the chain is a CA, except for the end-user certificate. Therefore, the number of CAs in a chain is $c-1$. Every CA must also issue its revocation artifact such that $r = c-1$. We hereafter assume the worst-case scenario for depth-4 and depth-3 certificate chains, i.e., every chain in the signature has the same length. Thus, the aforementioned equation is simplified to $s = 2c(t+1)$.

2.5 BLOCKCHAIN

Blockchain is a distributed database maintained by a decentralized network of nodes. Commonly, in blockchain implementations, every node in the network keeps a copy of the blockchain. To maintain the copies cohesive, nodes must agree on the state of the blockchain by executing a consensus protocol. From a data structure perspective, we assume a blockchain is a linked list of blocks, with every block comprising a header and a list of transactions. Headers maintain metadata about the block and a hash pointer to the previous block, while the list of transactions keeps the data shared by nodes in the network.

Bitcoin (NAKAMOTO, 2008) and Ethereum (WOOD, 2022) are public blockchain networks, meaning virtually anyone can integrate the network. Also, both are permissionless networks since no authorization mechanism regulates who can write and read data in the blockchain. In other words, any node in these networks can propose new transactions and read any data stored in the chronological list of blocks. However, we note that public and permissionless networks may only work for some blockchain use cases. For instance, a use case may exist where only authorized nodes should be part of the network, and just a subset of them should have writing permissions.

For such use cases, one can count on private and permissioned blockchains. Hyperledger Fabric is an outstanding example of such a network. It is part of the Hyperledger initiative (Hyperledger Foundation, 2018) maintained by the Linux Foundation. In contrast to Bitcoin and Ethereum, Hyperledger Fabric does not have a cryptocurrency or a public network. Instead, one can deploy a private network where digital certificates identify nodes, and the policies regulate which nodes have the right to write and read data.

2.6 PRIVATE KEY CONTAINERS

These are specialized structures designed to store, generate, and process private keys safely. Protection levels, authentication methods offered, and the underlying processing power vary wildly. Such features directly reflect on the price of the equipment. Entities that require high levels of security use devices such as *hardware security modules* (HSMs). They offer the most robust physical and logical security levels, often having multiple tamper-resistant features and highly secure cryptographic algorithm implementations. Trusted platform modules (TPMs) are similar hardware-based devices that hold cryptographic secrets, shipped in consumer-grade computer processors.

Commonly, end users hold *smartcards* and/or *cryptographic tokens*, cheaper hardware alternatives with limited processing power and storage. These devices come in the form of plastic cards or USB sticks, featuring a microprocessor with embedded memory and specialized cryptographic functions. They are widely used in the traditional PKI model (LONGO; STAPLETON, 2002). *Key management as a service* (KMaaS) is an alternative method of managing key pairs via cloud-based solutions that have become popular (LERMAN; MARKOWITCH; JR, 2012; KUZMINYKH; GHITA; SHIAELES, 2020) particularly because it does not require users to handle any hardware directly (Ponemon Institute, 2021).

2.7 PUBLIC KEY INFRASTRUCTURE CRITICISMS

Revocation and key management are unavoidable requirements of PKIs due to how public key cryptography is designed. The lifetime of the private key is set by the validity period of the certificate. However, if there is any security issue with the private key or its container, or if any of the attributes of the respective certificate become outdated, it is necessary to revoke the certificate. Therefore, it is clear that effective, rapid, and straightforward revocation is not only desirable but necessary. Furthermore, any issues regarding certificate misissuance must first be discovered before they can be addressed, such that certificate issuance monitoring and CA accountability are essential to PKI environments. Hence, before presenting our proposal, we briefly discuss these major factors that contribute to the high complexity of traditional PKIs.

Private key management

The choice of a private key container dictates many characteristics of a PKI and how users interact with it. Hardware devices struggle with affordability and interoperability; laypeople are commonly unable to obtain or interface directly with HSMs, and although TPMs may be certified by CAs, in practice, they are commonly associated with *devices* instead of end users. In the case of smartcards and tokens, devices may only work with a specific reader or operating system version. Such devices may also be stolen or broken. As they hold private keys, the owner must always notify the CA of any complications so that it can revoke the corresponding digital certificate (BOEYEN et al., 2008). Moreover, while relatively cheap when compared to HSMs, the acquisition of these devices for each user in a widespread PKI environment may hinder its deployment, especially in countries with limited resources.

Conversely, cloud-based solutions have been essential to the growth of digital signatures for the general public (CHONG; KIM; CHOI, 2021) and show that digital signature adoption can be increased by relieving the user from the burden of carrying a physical device. However, several issues have already been raised in the literature (DAMGåRD et al., 2013; PADILHA; PEDONE, 2015; KUMAR; BHATIA, 2020). Signers no longer have unique possession of their keys, as they are now stored and possibly replicated in cloud servers. Furthermore, the inherent lack of confidentiality of cloud storage diminishes the assurance that a user has that its private key has not been used by any other entity. Supporters of cloud-based solutions claim that there are KMaaS tools and procedures that allow users to audit and verify the authorized usage of their cryptographic keys.

Alas, these requirements often neglect that a widespread PKI model would have laypeople as its main users and, therefore, cannot expect them to perform these technical operations. We observe that the issues brought forward by the need for a secure key management model are inherent to the combined existence of the private key and a layperson end user. Therefore, any model with these characteristics is subject to some form of a key management challenge. We argue that, to address these issues while designing the model, one must (i) restrict the type of user allowed to participate or (ii) remove the need for a private key container entirely. We choose the latter, as we aim to define an inclusive, user-friendly PKI.

Certificate revocation

Revocation is probably the most complex and vulnerable service for CAs, registration authorities (RAs), and users of a traditional PKI. In the literature, criticisms about this mechanism are regular (ZHENG, 2003; KOCHER, 1998; RAYA et al., 2006; WANG; GAO; CHEN, 2020; NAOR; NISSIM, 1998; BOUR et al., 2020), and there exist several alternative methods to facilitate, speed up and guarantee the verification of the status of a digital certificate. However, Årnes (ÅRNES, 2000) surveys most of these revocation mechanisms, and their conclusion is that they often present several shortcomings and narrow suitability, with no definitive solution yet known. A common denominator is that private key compromises must be preemptively detected before they can ever be addressed.

We present a non-exhaustive list of issues with the CRL model, as follows: (i) lists must be issued even if no certificates are revoked; (ii) compromised private keys can be used until the newly published list contains the respective revoked certificate; (iii) lists can grow to large file sizes, which can congest network channels and quickly fill non-volatile storage media; (iv) network load spikes happen when a list expires (COOPER, 1999); (v) lists may become inaccessible due to server outages; and (vi) human error may revoke certificates incorrectly.

OCSP also has its shortcomings (GUTMANN, 2002; WOHLMACHER, 2000; SANTESSON et al., 2013). Queries are online and thus increase the load on communication networks. Furthermore, servers are subject to denial of service attacks. OCSP servers may provide fresher revocation information when compared to CRLs. However, many OCSP services do not have direct access to CA databases due to security concerns and use CRLs as a primary source of revocation data, thus inheriting the latency issues of CRLs. Finally, querying revocation information may indirectly reveal sensitive data about the petitioner.

As previously mentioned, revocation is essential to maintaining trust in the PKI because private keys are always subject to compromises, and attributes in certificates become outdated. We argue that a model that minimizes such threats is able to reduce the consequences of maintaining and providing revocation information. End users comprise most certificate holders in a PKI and are commonly liable for having certificates revoked. Any model that aims to reduce the burden of revocation must target such users. Therefore, we propose a PKI framework that provides irrevocable certificates to end users.

Certificate misissuance detection

As previously mentioned, certificate revocation mechanisms are used to handle cases where a private key has been deemed unfit to sign further messages. However, before a certificate can be revoked, the compromise must be detected. Typically, due to the increased security of root and intermediate certification authorities (CAs), they are assumed to be secure enough such that this monitoring is unnecessary. Final CAs, however, are known to be more easily compromised (MEULEN, 2013). Hence, there exists a need to monitor certificate issuance by final CAs.

Systems that deal with the monitoring of issued certificates are usually named after the well-known Certificate Transparency (CT) protocol (LAURIE et al., 2021). In this protocol, CAs submit “pre-certificates” to CT loggers, which respond with signed CT timestamps, acting as a promise that these certificates will be logged. Typically, logs are built as a Merkle tree and reconstructed with new certificates after some specified period; CAs typically embed these timestamps in the issued certificates as X.509 extensions. Systems that require CT timestamps only verify their respective signatures, while monitors and auditors check log integrity. Still, the addition of extra signatures can lead to certificate bloat, harming the efficiency of TLS protocols, depending on the signature algorithms chosen by loggers.

Furthermore, the addition of multiple extra signatures on the certificate must be addressed when future-proofing any CT-like protocol. For instance, with the advent of quantum computing, classic signature algorithms such as RSA and ECDSA are fated to be replaced. Active research is being conducted on quantum-resistant cryptographic algorithms. However, the consequences of quantum resistance are an increase in key and signature sizes (KAMPANAKIS et al., 2018), which greatly affect CT artifacts as they might contain multiple signatures.

Moreover, while unusual, we can extend the reasoning for certificate misissuance detection to the end user in the form of *document "miss-signature"* detection. Just as CAs are susceptible to misissue certificates, either maliciously or by mistake, end users can also have their credentials stolen or their private key containers broken into (MAHANTA; AZAD; KHAN, 2015). Thus, the challenge of detecting certificate misissuance now becomes a question of detecting a "miss-signature". Additionally, the presence of laypeople instead of specialized individuals increases the overall difficulty of this problem when trying to maintain or improve usability and transparency.

While CT manages to monitor the issuance of TLS certificates with some degree of success, simply adapting it to track end-user documents would require extensive changes in document signature generation processes, requiring end users to log every created signature by themselves. A widespread PKI environment is expected to have laypeople as the largest portion of its user base and cannot rely on the assumption that non-technical certificate holders would follow the required steps to register signed documents. Imposing extra steps in the creation of digitally signed documents ultimately hurts usability and restrains adoption. Therefore, we argue that any proposal based on the CT framework must be transparent to the end user.

3 ONE-TIME CERTIFICATES

We propose a new cryptographic key management model to address the revocation and key management issues discussed in Section 2.7. The main differences between our model and the traditional PKI are threefold: (i) we bind a digital certificate to a single document, preventing loss of authenticity from private key compromises; (ii) we reduce the lifecycle of end-user private keys, lowering the complexity associated with their management; and (iii) we forego revocation mechanisms at the end-user level. We discuss those traits below and compare them to the traditional model.

After key pair generation, the public key must be certified by a CA and bound to a certificate. This process does not change between a classic PKI and our proposal. However, to ensure that signed documents are validated correctly for long periods of time, we propose to bind a document to a certificate via a signed attribute containing the cryptographic hash of the original document. To validate a digitally signed document in our model, in addition to the traditional verification steps, the digest of the original document is matched against the digest embedded in the certificate. This binding may be seen exemplified in Figure 2 as the narrowing of the space of documents a certificate is able to verify.

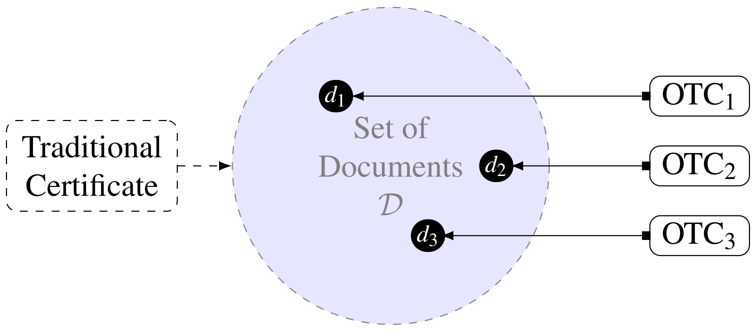

Figure 2 – Comparison of the one-to-many nature of traditional certificates and the one-to-one characteristic of OTCs.

The signing process of a document under the *one-time certificate* (OTC) model (MAYR et al., 2024) is exemplified in Figure 3. We observe that most steps and artifacts follow the traditional procedures for generating a signed digital document. The only change is the addition of step ii: the embedding of the document hash in the certificate along with the user identity information, effectively creating a unique binding between end user, key pair, and document.

In our model, while a private key can be used to generate any number of signatures, only the document identified in the corresponding certificate will be correctly validated. Thus, a private key associated only with an OTC has no further *effective* use and may be safely *destroyed* after document signing. Consequently, we are able to use unique key pairs for each signature generation procedure. The private key lifecycle is not abruptly interrupted by compromises due to the aforementioned unique binding. In other words, a leaked private key may only be used to sign a specific document. Furthermore, document signature and private key deletion do not

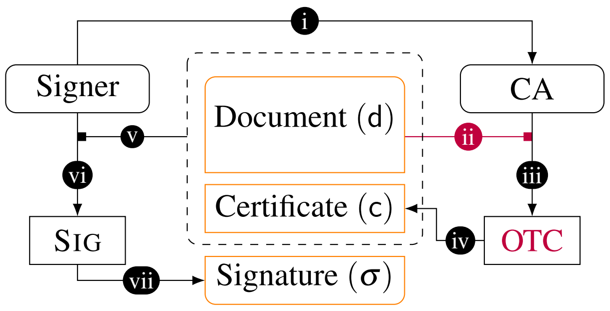

Figure 3 – Ordered steps to create a digitally signed document under the OTC model. We assume that a key pair has been previously generated by the signer. In step i, the public key is sent by the signer to the CA. In step ii, the signer computes the hash of the original document and sends it to the CA. In step iii, the CA issues an OTC with fresh attributes to the signer. In step iv, the signer appends their OTC to the original document. In step v, the signer computes the hash h of the tuple (d, c). In step vi, the signer signs h with their algorithm of choice. In step vii, the signature is appended to the document, creating the final artifact (d, c, σ) in orange.

need to be atomic operations, as the OTC binding prevents effective key misuse even if the key is leaked between these operations.

A natural consequence of the unique binding between signer, certificate, and document is that the underlying public key must be certified immediately before every signature generation. As the tuple of signer, digital certificate, and document is unique, issuance happens whenever a signature must be generated. As previously mentioned, it is the responsibility of the CA to ensure that attributes are not stale throughout the lifetime of the certificate. Therefore, when a certificate is issued, every attribute is valid. In our model, since certificate issuance happens immediately before signature generation, all attributes can be assumed valid at the time of signature generation T_σ. Therefore, timestamps are not required when using OTCs. Figure 4 shows the OTC lifetime.

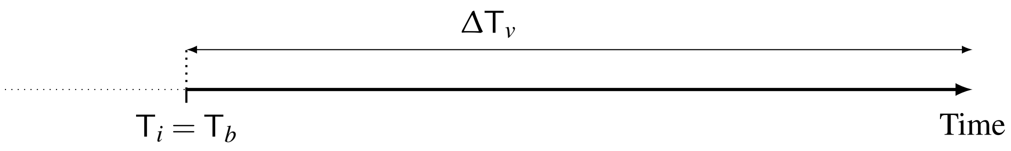

Figure 4 – Certificate time references for an OTC.

Consequently, by removing the eventual staleness of the attributes and preventing private key compromises due to a short lifecycle derived from the unique binding between certificates and documents, we argue that revocation mechanisms at the end user level are unnecessary. In short, we remove the association between the lifecycle of the private key and the validity period of the corresponding digital certificate. We note that OTCs are *not ephemeral* since they are valid and may be verified for the lifecycle of the signature, which is only bound by the cryptographic strength of the underlying algorithm.

3.1 SPECIFICATIONS & INTEROPERABILITY

Our proposal is technically simple and can be put into practice with minor overhead. It is backward-compatible with current PKI standards, such as the well-known RFC 5280 profile (BOEYEN et al., 2008), signature validation applications (SVAs), and conventional cryptographic algorithms. One-time certificates are pursuant to the X.509 profile, considering the following characteristics. The `notAfter` entry of the `Validity` ASN.1 structure inside `TBSCertificate`, which represents T_a in Figure 1, is set to an arbitrarily remote date in the future, beyond the intended lifetime of any digitally signed document.

OTCs do not require any revocation mechanisms, such as CRLs or OCSP queries. However, there must exist such revocation information available in specific X.509 certificate extensions, either through `id-ce-cRLDistributionPoints` (BOEYEN et al., 2008, Sec. 4.2.1.13) in the case of CRLs or `id-pe-authorityInfoAccess` (BOEYEN et al., 2008, Sec. 4.2.2.1) for OCSP servers. Therefore, a CA that issues OTCs should also issue a single, empty CRL. The revocation list must inherit the validity period of the CA. Alternatively, an OTC-compliant OCSP server may be deployed. OCSP responses must return a `CertStatus` entry of `good` for any queries about OTCs issued by CAs it is responsible for or `unknown` for queries about other certificates.

One-time certificates must contain an attribute representing the binding between the user, the certificate, and the to-be-signed document. It is a critical certificate extension, i.e., an ASN.1 structure of type `Extension`, whose `extnValue` is of type `OneTimeCertificateHash`, according to the syntax below, pursuant to RFC 5912 (HOFFMAN; SCHAAD, 2010). The object identifier `extnID` of the extension is yet to be defined in future standards. The choice of the cryptographic hash algorithm is collected from the end user by the signature creation application (SCA).

```
OneTimeCertificateHash ::= SEQUENCE {
  hashAlgorithm       AlgorithmIdentifier,
  documentHash        OCTET STRING
}
```

As previously mentioned, the binding between a document and an end-user through the certificate is a major aspect that grants OTCs the ability to waive revocation and secure key pair generation. Thus, SVAs *must* validate this binding in the form of a comparison between the contents of the `OneTimeCertificateHash` extension and the digest of the original document. Otherwise, legacy SVAs may be tricked into accepting electronic documents with signatures that are correctly *verified* by the public key in the OTC but incorrectly *validated* due to mismatching digests. We recall that SVAs are required to reject certificates with unknown critical extensions. Thus, by setting the `OneTimeCertificateHash` to critical, we prevent any compliant legacy SVAs from being tricked.

3.2 CERTIFICATE ISSUANCE & AUTHENTICATION

While a thorough analysis of different authentication methods is out of the scope of this work, as OTCs function independently from the registration and authentication methods used by the infrastructure, automating the issuance process after a user has been registered greatly increases the usability of OTCs and the PKI in general. Nonetheless, there are several adjustments that can be made in the registration and authentication steps of the issuance process to fit any particular environment. Smaller PKIs might still opt for manual registration and authentication, local PKIs may opt for manual registration and automatic certificate issuance through an identity provider (IdP), and a high-load widespread PKI is able to offer both manual and automated registration and certificate issuance with different levels of assurance.

We assume the user has already been registered in the IdP by whatever methods are required by the PKI. An example of the routine performed by an SCA is given by Figure 5 as follows: (i) the user accesses its SCA; (ii) the SCA redirects the user to a trusted IdP; (iii) the user authenticates itself and allows the sharing of its information; (iv) the user submits the document to be signed; (v) the SCA generates a key pair, a certificate signing request, and the hash of the document and forwards them to the CA along with the signed IdP token; (vi) the CA issues a certificate containing up-to-date user information and the hash of the document and returns it to the SCA; and (vii) the SCA generates the signature according to the chosen policy and returns the signed document to the user.

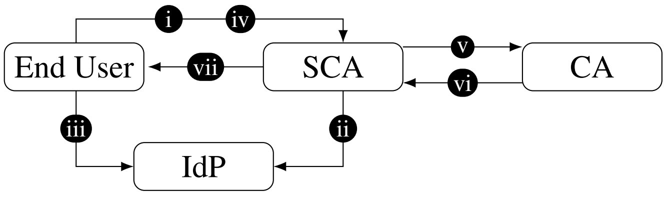

Figure 5 – Example of a document signature routine under the OTC model.

r

We observe that a direct consequence of requiring frequent authorization from an IdP leads to a stronger assurance that the original holder is the one who signed a document when compared to traditional models. That is, traditional PKIs make several assumptions regarding users and their key management routines. One such assumption refers to the use of the private key, which should only be used by the original holder of the certificate, i.e., “lending” a private key should not happen. Alas, such assumptions are hardly realistic as there is little guarantee of who is using the private key at a given moment.

Conversely, under the OTC model, users are required to authenticate themselves to sign new documents. A compliant PKI may employ IdPs with several different mechanisms to reliably ascertain that a given user is accessing the system. IdPs may require users to offer sufficient proof that they are the ones authenticating themselves, such as multi-factor authentication methods, enhancing the assurance that the original holder is the one signing the document.

3.3 PRIVACY CONSIDERATIONS

In the traditional model, users are able to acquire certificates and sign any number of documents without revealing any information about the documents they sign. However, the inclusion of the hash of the document in the certificate in the OTC model reveals some, albeit minimal, information about the user and the document it wishes to sign. In this manner, CAs could track high-profile public documents and refuse to issue certificates for users who wish to sign them.

PKIs that are worried about the possibility of this denial of service attack for specific documents are able to circumvent it by including a *nonce* in the signature, appending it to the original document before generating the hash to be included in the certificate. SVAs that detect the presence of this nonce must append it to the document before generating the hash to be validated with the certificate. In this manner, offending CAs are not able to track public documents and refuse to issue certificates in a document-by-document case.

3.4 SECURITY ANALYSIS

We argue that our proposal is at least as secure as the traditional PKI model, as it restricts the number of documents that can be certified by a digital certificate. We emphasize that threats arising from breaches of trusted parties, such as CAs and RAs, are not in the scope of our proposal. These issues are commonly addressed via the Certificate Transparency (LAURIE et al., 2021) framework. Hereafter, we show that OTCs solve the problem of stale attributes and private key management, eliminating the need for revocation mechanisms at the end-user level.

Definition 3. An attribute is considered *stale* or *outdated* if it is embedded within the digital certificate and does not correctly identify the entity when a signature is generated.

Proposition 1. OTC attributes cannot be stale.

Proof. Let pk be a public key and c its corresponding OTC. We recall that T_i is the moment when c is created from pk and attributes and T_b is the start of its validity period. Such attributes are valid at T_i due to the role of a CA. In our model, the validity period of c starts immediately after issuance, such that $\mathsf{T}_i \lesssim \mathsf{T}_b$. Therefore, the attributes contained in c and validated at T_b are guaranteed to be accurate. □

A consequence of Proposition 1 is that certificate issuance acts as a time-mark, and document binding prevents certificate reuse. Therefore, signatures under instances of our model do not require timestamps to achieve the same characteristics of other policies. Additionally, we note that any changes to user attributes would be reflected in new OTCs. Furthermore, consider T_σ to be the moment a signature σ is generated with sk. While we expect $\mathsf{T}_b \lesssim \mathsf{T}_\sigma$, it is not a prerequisite since attributes are validated at T_b.

Definition 4 ((BELLARE; MINER, 1999))**.** The *forward security* property indicates that it is computationally impractical for any attacker to forge a valid signature for a time period before the compromise of the corresponding private key.

Traditionally, forward security proofs include a forger F that conducts an adaptive chosen message attack (GOLDWASSER; MICALI; RIVEST, 1988) on requested messages of its choice until it chooses to "break-in" at time T_b by requesting the private key sk and generating a distinct signed message m at time T_σ such that $\mathsf{T}_\sigma < \mathsf{T}_b$; that is, the forged signature must be generated for a time before its compromise. The forgery is successful if the signature is considered valid (ITKIS; REYZIN, 2001).

We extend this traditional notion of forward security to include certificate policies and show that documents signed using the OTC model are forward-secure. In our model, the forger F makes an adaptive chosen message attack on requested documents receiving the corresponding signed document σ_i and OTC $\mathsf{o_i}$ until it requests the secret key sk at time T_b; the forgery is considered valid if the forger is able to sign a distinct document d and successfully validate it using any of the issued OTC c under the correct policy for any time before T_b.

Proposition 2. The OTC model is forward-secure.

Proof. Let $\mathcal{D}$ be the set of documents requested by the forger, $\mathcal{S}$ the set of signatures, and $\mathcal{O}$ the set of OTCs received by the forger such that d_i is bound to c_i and the tuple $(\sigma_i \in \mathcal{S}, \mathsf{d}_i \in \mathcal{D}, \mathsf{c}_i \in \mathcal{O})$ is valid. The forgery is considered a success if the forger is able to generate a valid tuple $(\sigma_i, \mathsf{d}_f, \mathsf{c}_i)$ such that $\mathsf{d}_f \neq \mathsf{d}_i$. However, as a consequence of the OTC binding, for $(\sigma_i, \mathsf{d}_f, \mathsf{c}_i)$ to be valid, $\mathsf{d}_f = \mathsf{d}_i$. □

The threat modeling for the forward security property can be expanded to include *any* signatures made by the forger, not only those before T_b. The removal of this constraint from the attacker leads to the stronger security notion of *unforgeability* (GOLDWASSER; MICALI; RIVEST, 1988); that is, no signature made by the forger is considered valid. Extending this concept for forward security with certificate policies is straightforward, requiring the removal of the same constraint, and shows that a signed document is unforgeable under a policy.

We note that the proof for Proposition 4 does not depend on any time frames to be correct, neither from the forger or the oracle; the forger may generate a forgery for any time, and the forgery would still be unsuccessful as a result of the restrictions imposed by the OTC model and its binding. We show that restricting the number of valid verifications to a single document per digital certificate hinders forgeries in such a manner that not only is our model forward secure, but also that signed documents associated with OTCs are unforgeable.

Proposition 3. Signatures linked to OTCs are unforgeable.

Proof. Assume trustworthy issuer CAs and identity providers, a secure cryptographic hash function, and a compliant signature verification application that correctly validates the digest of the signed document against the embedded hash. Let $(\mathsf{sk}, \mathsf{pk})$ be a key pair associated only with

OTCs, an OTC c that certifies pk and a document d, and $\mathcal{S}$ to be the set of signatures signed by sk that are correctly validated with c. We recall that forgeries must differ from existing signatures and thus require that $|\mathcal{S}| > 1$. However, the OTC model shrinks $\mathcal{S}$ to one signature. Therefore, as $|\mathcal{S}| = 1$ under the OTC model, any other signatures generated with sk fail to validate using c; these signatures cannot be forged. □

We note from Proposition 3 that a private key associated only with OTCs cannot be *effectively* compromised by any outside attacker; that is, an attacker with access to the user's private key cannot generate new valid signatures. Furthermore, Proposition 3 allows the relaxation of the security requirements for the user signature, i.e., as long as the signature algorithm and the private key used by the CA are secure, the breaking of the algorithm used by the user would not invalidate previously issued certificates and signed certificates.

This property allows key pair generation not to require secure private containers, which usually have lower key generation throughput. Hence, private keys can be generated where they are most convenient, in environments such as embedded devices, cloud services, and personal computers, improving usability. We note that key pair reuse is not forbidden and does not hinder the security of our model due to the unique binding between a certificate and a document. However, to keep usability in the foreground, we assume key pairs are not stored and thus must be obtained immediately prior to a signature.

Furthermore, Proposition 3 also enables the certification of one-time signatures under the traditional model of CSR generation before certificate issuance. Traditionally, the second use of the OTS private key would reveal information about the private key. However, under the OTC model, no other signatures can be valid, negating the issues of leaking private key information of OTS schemes. We highlight that our proposal is crypto-agile and does not require the use of OTS schemes, but it does not prevent their use.

Nevertheless, it is worth noting that while OTCs protect the end user from forgeries, it does not solve traditional security issues resulting from corrupt or negligent issuer CAs; that is, issuer CAs can still generate valid certificates for any given combination of user and document without being given explicit consent from the user. Furthermore, we remark that the security of OTCs largely depends on the security of the hash function used to bind the document to the certificate; we recommend the use of hash functions that are at least as secure as the ones used in the signature process.

3.5 PERFORMANCE EVALUATIONS

Generating a key pair and issuing a new digital certificate before signature generation has heavy implications for the performance of creating digitally signed electronic documents. In this section, we measure the overhead caused by generating key pairs on demand, discuss its consequences, and propose alternative key generation methods to evade such performance costs. We enumerate how many signatures are needed to successfully sign and validate a document using traditional signature policies and OTCs. Finally, we calculate the expected throughput of document signing and validation and present our deployment recommendations considering several use cases.

In the traditional PKI model, an end user is expected to possess a certified key pair already. Consequently, only the performance of operations on signature is a relevant metric. However, when assessing the performance of OTCs, we must also consider the performance of key generation when estimating the number of signed documents an end user is able to create and validate with no detriment to their workflow.

We first measure the performance of several digital signature schemes already used in PKIs for various security levels based on the estimations in (BARKER, 2020, Table 2). Hereafter, our measurements are performed in a GNU/Linux machine with kernel version 6.5.7, equipped with an AMD Ryzen™ 5700G @ 3.8GHz, “turbo boost” disabled, and OpenSSL 3.1.3. All operations on signatures use the SHA-256 cryptographic hash function when applicable. Table 1 shows that requiring key pair generation before every document signature introduces a bottleneck in the document signature process. Thus, it is clear that key generation speed becomes a critical performance component for OTCs.

Table 1 – Average performance of 2^{10} iterations, in microseconds, for each operation of several commonly used digital signature algorithms with various security levels n.

	$n = 128$			$n = 192$	$n = 224$
	RSA-3072	P-256	Ed25519	P-384	Ed448
KEYGEN	122864.62	21.04	34.91	612.64	176.16
SIG	1448.06	17.17	30.78	636.99	173.60
VER	29.84	49.32	84.14	530.75	194.34

We present two methods to address this trade-off, loosely referred to as policies. The first is the generation of key pairs on demand, hereafter called OTC-D. This is the model presented at the beginning of this chapter, which allows us to omit key management. We recall that Proposition 3 removes the potential of private key compromises. Therefore, private keys and CSRs can be generated in advance and stored for later use. We call this method of pre-generating artifacts *bulk key generation* or OTC-B. This method introduces some light key management but manages to avoid most of the overhead introduced by our model, i.e., the on-demand generation of key pairs. We note that these CSRs are built only to show private key ownership and do not require user identity information or the document hash; these are sent to the CA when requesting a new certificate.

To sign a document using OTC-B, the signer requests a pre-generated CSR, requests an OTC, signs the document, and discards the private key. Furthermore, in both methods, key generation and storage can be delegated to other entities, allowing low-power devices to sign documents using OTCs in a timely manner. However, we note that neither OTC-B nor OTC-D prevent the overhead of repeated certificate issuance at the CA level; this is an unavoidable trade-off of our model.

3.5.1 Comparison to traditional signature policies

We recall that signature policies can augment a digitally signed document with further signed artifacts, such as revocation information and timestamps. In this section, we discuss how OTCs can be used in the context of Advanced Electronic Signatures. We compare the number of signatures required by European policies (ETSI, 2021a, Table 1) to process digitally signed documents to semantically equivalent documents created with OTCs. Finally, we use the measurements of Table 1 to estimate the performance of our model in practical use cases.

Timestamps greatly affect the performance of processing digitally signed documents, as seen in Section 2.4. Distinct policies require a different number of timestamps (ETSI, 2021b, Table 1). The set of B policies does not require timestamps and requires the least amount of information to be created; T-type signatures include a time-mark in the form of a timestamp to provide information about when the signature was created; policies of type LT additionally include certificates from the whole chain to the set of signed artifacts of T signatures; the archival signature policy LTA includes an additional timestamp and complete certification and revocation artifacts to the signature. Furthermore, every policy requires the inclusion of the user signing certificate as a signed attribute.

We recall from Proposition 1 that OTCs act as time-marks due to their issuance procedures. Let ς be the minimal amount of signatures a document carries when created according to a policy p. We split this number into quantities representing how many signatures are created for every step in the PKI model: (i) σ_d is the number of signatures that cover the original document; (ii) σ_c is the number of signatures needed in a CSR; (iii) σ_i is the number of signatures needed to issue a certificate; and (iv) σ_t is the number of timestamps required. The number of operations required by each policy is shown in Table 2.

Table 2 – Number of key and signature operations required by a policy p in the lifecycle of a digitally signed electronic document for a certificate chain of depth c.

p	GEN	SIG					VER	
		σ_d	σ_c	σ_i	σ_t	ς	$c = 3$	$c = 4$
B	**0**	1	0	0	0	**1**	**6**	**8**
T	**0**	1	0	0	1	**2**	**12**	**16**
LT	**0**	1	0	0	1	**2**	**12**	**16**
LTA	**0**	1	0	0	2	**3**	**18**	**24**
OTC-B	**0**	1	0	1	0	**2**	**5**	**7**
OTC-D	**1**	1	1	1	0	**3**	**5**	**7**

We note that the removal of timestamps in the OTC model lowers the amount of signatures required to be validated down to $s = 2c$. Additionally, SVAs compatible with our proposal need not support revocation processing at the signer level according to the certification path validation algorithm of RFC 5280 (BOEYEN et al., 2008, Sec. 6.3). Hence, one less signature must be verified throughout the algorithm: the empty CRL or OCSP response, further reducing the number of validations to $s = 2c - 1$.

Finally, we present an analytical throughput measurement for each policy and key generation method and give our deployment recommendations based on the intrinsic characteristics of our model. Table 3 estimates how many digitally signed documents can be created and validated per second by using the algorithm measurements of Table 1 and the number of operations per policy in Table 2.

Table 3 – Throughput measurements, in **documents per second**, of the creation and validation of digitally signed documents with a policy p and $c = 4$, considering commonly used digital signature algorithms with various security levels n.

p	$n = 128$						$n = 192$		$n = 224$	
	RSA-3072		P-256		Ed25519		P-384		Ed448	
	Sig	Ver	Sig	Ver	Sig	Ver	Sig	Ver	Sig	Ver
B	690.58	4188.76	58251.33	2534.55	32484.22	1485.57	1569.89	235.07	5760.25	643.21
T	345.29	2094.38	29125.67	1267.28	16242.11	742.79	784.94	117.54	2880.13	321.61
LT	345.29	2094.38	29125.67	1267.28	16242.11	742.79	784.94	117.54	2880.13	321.61
LTA	230.19	1396.25	19417.11	844.85	10828.07	495.19	523.30	78.36	1920.08	214.40
OTC-B	345.29	4787.15	29125.67	2896.63	16242.11	1697.80	784.94	268.65	2880.13	735.10
OTC-D	2.68	4787.15	8724.40	2896.63	5073.85	1697.80	266.75	268.65	953.02	735.10

As expected, creating signed documents under OTC-D is hindered by the performance of key generation, especially notable when using the RSA signature algorithm. On the other hand, OTC-B offers competitive performance compared to all policies that enforce timestamps. We note that both OTC policies excel at the document validation step since there are fewer certificate chains and revocation artifacts to be verified. This is highly sought since documents are much more often validated than digitally signed.

Ultimately, each application must decide on the acceptable throughput for their systems and which key generation mechanism better suits its purpose. Applications that sign documents once every few minutes might want to choose the simpler on-demand method. At the same time, systems with low processing power might need to delegate this task to a server, and applications that require high throughput may pre-generate key pairs. Systems may distribute the load between signature creation and validation routines, and stakeholders may provide only one of the two services. Our model does not hinder such adjustments.

Furthermore, we remark that the load increase to issuer CAs caused by our model is comparable to the standard load of KMaaS applications; we recall that such applications must activate and make use of an end-user private key to generate a signature for each document. Likewise, an OTC issuer CA must employ its own private key to issue new OTCs.

The benefits of using OTCs cannot be overstated. Revocation at the end-user level has been removed, private key containers are no longer a concern, and signature policies have been made much simpler, particularly through the redundancy of timestamps. As a whole, the creation and validation of digitally signed documents is simplified at the expense of lower signature generation performance. However, it can be tweaked to desirable levels using specific key generation strategies and different digital signature algorithms.

4 THE OTSCHAIN FRAMEWORK

To solve the issue of effectively monitoring digitally signed electronic documents and, consequently, key pair usage, we propose a blockchain-based strategy where final CAs share a distributed ledger of issued digital certificates and message hashes. In our strategy, multiple compliant PKI infrastructures contribute to the same blockchain network since users may have one or more digital certificates issued by unrelated CAs. To improve the authenticity and non-repudiation of transactions, we propose to identify CAs in the network with the same key pair they use to issue digital certificates. As a result, CAs can read and write data on the blockchain. When reading data, CAs and other interested parties, such as end users and signature validation applications (SVAs), act as monitors. To properly validate a signed document, our strategy requires a connection to the blockchain in addition to the usual access to certificate chains and revocation data by the SVA.

While the use of this framework is not exclusive to one-time signatures, we remark that our choice of hash-based signatures enhances many aspects of this proposal, such as quantum resistance. Furthermore, as our proposal effectively limits a key pair to a single signature, the use of one-time signature schemes becomes natural. Moreover, the use of one-time signature schemes in our framework improves usability for end users, such that private keys need not be stored after the first document is signed. Additionally, we are able to prevent effective repeated OTS private key usage via blockchain semantic validation, addressing the main constraint of such schemes. A direct consequence of our restriction is the higher load for PKIs, particularly in certificate issuance and the key generation step. We argue that hash-based OTS schemes have efficient key generation when considering sufficient and available entropy, compared to classic methods and even other quantum-resistant alternatives; only random bits are needed, with no additional constraints imposed (HüLSING et al., 2018).

Certificates used in the context of the CT protocol may include a certificate extension α of the signed timestamp response from the CT logs. Analogously, the usage of OTSCHAIN may also be identified by the presence of an extension β embedded into the certificate. While α generates unwanted certificate bloat, as it contains one or more digital signatures, β may hold a short list of ledger URIs, or it can be empty when such information is already known via out-of-band mechanisms. Therefore, we are able to minimize the overhead caused by CT-like proposals while maintaining the desired properties.

Any legislation that recognizes advanced signatures similarly to the European standard ETSI EN 319 102-1 (ETSI, 2021b) is able to use this proposal without any changes to its regulations. That is, advanced and qualified signatures and derived requirements are agreed upon by all parties involved in processing digitally signed documents. Notably, the Brazilian civil registry PKI allows the use of any certificate extension and is currently using a proof-of-concept version of OTSCHAIN within their existing legal frameworks.

We define the entities and data that belong to the protocol for any $i, k \in \mathbb{N}$. The actors are a certification authority CA_i, an end-user U_i, and a CT-like monitor M_i. Data inserted in the blockchain is denoted as follows: a message hash H_i to be signed by U_i, a digital certificate C_i carrying a set of attributes $\alpha_i = (\alpha_i^1, \alpha_i^2, \dots)$, owned by U_i and issued by CA_i, and a tuple $\mathsf{L}_i = (\mathsf{C}_i, \mathsf{H}_i)$ linking certificate and message hash. Finally, we define the blockchain elements: an asset A_i carrying a tuple L_i, a transaction payload P_i encoding A_i, the signed payload σ_{P_i}, a transaction request $\mathsf{R}_i = (\mathsf{P}_i, \sigma_{\mathsf{P}_i})$, a transaction output $\mathsf{O}_i = (\mathsf{A}_1, \mathsf{A}_2, \dots)$ (or $\mathsf{O}_i = \emptyset$), a transaction T_i, and a block B_k containing T_i. The flowchart in Figure 6 depicts the interactions among these entities.

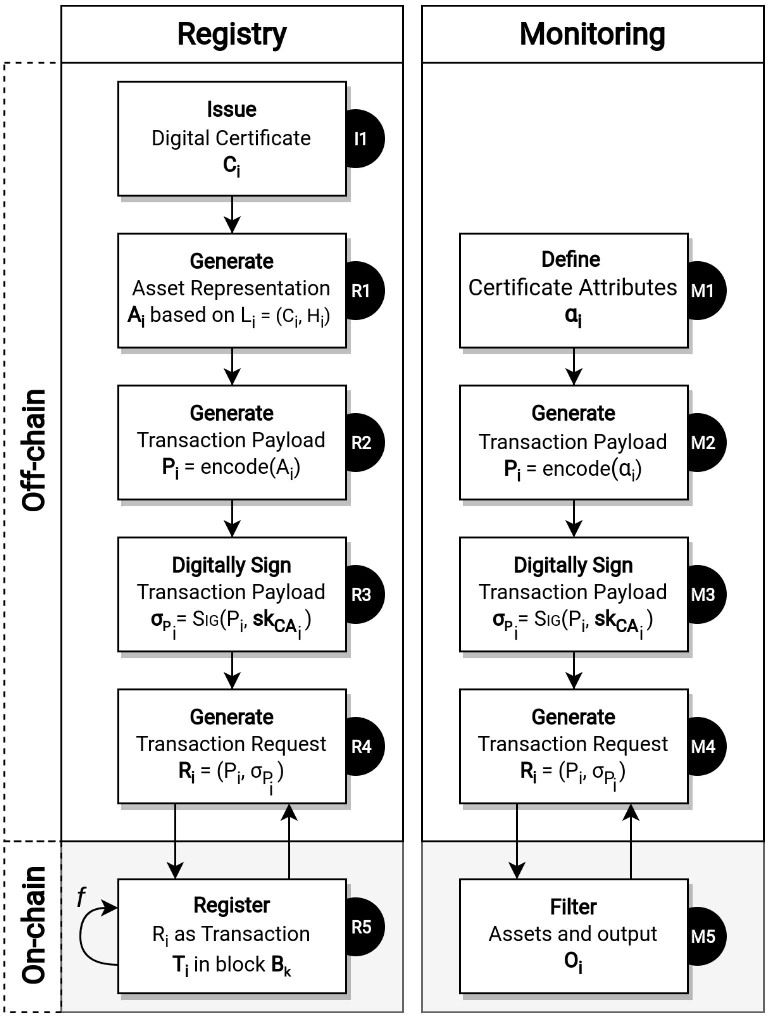
Registry
Monitoring
Off-chain
On-chain
Issue Digital Certificate C_i
I1
Generate Asset Representation A_i based on $L_i = (C_i, H_i)$
R1
Generate Transaction Payload P_i = encode(A_i)
R2
Digitally Sign Transaction Payload σ_{P_i} = Sig(P_i, sk_{CA_i})
R3
Generate Transaction Request $R_i = (P_i, \sigma_{P_i})$
R4
Register R_i as Transaction T_i in block B_k
R5
f
Define Certificate Attributes α_i
M1
Generate Transaction Payload P_i = encode(α_i)
M2
Digitally Sign Transaction Payload σ_{P_i} = Sig(P_i, sk_{CA_i})
M3
Generate Transaction Request $R_i = (P_i, \sigma_{P_i})$
M4
Filter Assets and output O_i
M5

Figure 6 – Flowchart of the registering and monitoring steps of OTSCHAIN.

In the context of the blockchain network roles, we propose a network where every PKI can maintain a copy of the blockchain. Furthermore, root and intermediate CAs are not required to be part of the network or publish data since our proposal focuses on end-user certificates. However, every final CA is encouraged to maintain a copy of the blockchain and engage in the consensus protocol. SVAs should be nodes in the network as well, as they act as monitors, querying the existence of assets A_i to validate the document corresponding to H_i. Additionally, having a local copy of the blockchain speeds up document validation, which brings further usability to end users.

The left side of Figure 6 illustrates the general steps toward registering digital certificates in the blockchain. We recall that CA_i is the final CA that issued C_i. Step I1 in the flowchart represents the certificate issuance, which does not depend on the active blockchain network. The message hash H_i is obtained via an out-of-band mechanism, for instance, via the signature creation application. We remark that there are no interventions to the traditional certificate issuance procedures; it is merely depicted for completeness.

The first step R1 of the registry process consists of the generation of an asset representation A_i of the end-user digital certificate and message hash by the CA. Then, CA_i is ready to generate a transaction payload P_i in step R2, encoding the asset using any format required by the blockchain network. Subsequently, in step R3 of the protocol, $\mathsf{CA_i}$ signs the transaction payload using its private key, which enables the generation of the transaction request $\mathsf{R_i}$ in step R4. Previous steps execute outside of the blockchain network and culminate in step R5, which consists of storing the data as a transaction to be consumed in the monitoring process at a later time.

We note that between steps $\mathsf{R_4}$ and $\mathsf{R_5}$, the transaction proposal must be broadcast by $\mathsf{CA_i}$ so it reaches the other nodes in the network, making the copies cohesive. The consensus protocol by which nodes will agree on the new state of the blockchain depends on the arrangements between nodes. For instance, they may agree on a majority policy, meaning that the data is stored in the blockchain if more than 50% of the nodes agree on the transaction request. We present a detailed discussion about deployment recommendations and measurements in Section 4.3.

Let ℓ be the set of all possible L_i. Step R5 contains the application of a function $f : \ell \to \{0,1\}$. Consequently, our framework may be customized to check certain semantic restrictions. Examples of such verifications are: (i) how recent was the certificate issued, to prevent backdating; (ii) certificate issuance rules according to the content of α_i, for instance, to allow only certain organizational units to participate in the network; and (iii) restrict node operations, for example, to impose upper bounds on attempts to register certificates by a faulty CA. The transaction fails if f returns a false response, which prevents the insertion of T_i in any block. We opt to register the offending request to improve the detection of malpractice and possible attacks.

In our proposal, f is set to control additional registration of certificates containing a previously logged public key. As previously discussed, we move log information out of the certificate itself to the blockchain. We require compliant SVAs to check the blockchain for the existence of the correct tuple L_i. In this manner, even if an OTS private key is used multiple times, only the registered hash associated with the submitted signed document will be valid under this policy.

After the registering process, $\mathsf{A_i}$ is available to be read by any node with permission, which enables the start of the monitoring process. The right side of Figure 6 details such procedures. In step M1, a monitor M_i defines a set of attributes α_i to search. Such attributes are encoded in a transaction payload in step M2 and digitally signed in step M3 to create a transaction request in step M4, analogous to register operations. Finally, in step M5, the network consults the set ℓ for a matching L_i, and returns the suitable asset list to M_i.

We emphasize that monitors are independent; further external validation by these entities, such as the processing of specific documents, are by design out of the scope of our framework. Such behavior is particular to how a monitor operates and which services they offer. Moreover, as we require the registration of every certificate in the blockchain to properly validate a document, monitors are able to detect and warn users of every certificate that has been issued in their name. In other words, whenever an attacker uses a compromised credential, it must inevitably reveal itself by having the resulting certificate registered in the blockchain. Thus, monitors and end users may take proper steps to deal with this security breach quickly.

Our strategy is suitable for a wide range of use cases, especially when a long-term registry of signed documents is a hard requirement. For instance, consider the issuance of degree certificates by educational institutions, which may be subject to fraud (PALMA et al., 2019). In this case, as long as the blockchain network is online, students and other interested parties can validate a degree certificate by checking the registry of the signature on the OTSCHAIN. Moreover, since our framework ties any digital certificate to a single document, compromised private keys would not allow the issuance of valid degree certificates. Furthermore, through the use of quantum-proof OTS schemes, degrees can be protected from quantum attackers long before they become a real threat.

4.1 PRIVACY IMPACT

As our proposal requires the inclusion of entire end-user certificates in a blockchain, we must consider the ramifications of facilitating access to attributes contained in such certificates, considering data protection laws such as the EU GDPR (European Parliament and the Council of the European Union, 2016) and the Brazilian LGPD (BRASIL, 2018). We discuss the differences between registering TLS and end-user certificates, the intrinsic trade-offs between privacy and monitoring services, the differences between a public and private blockchain privacy-wise, and how to use OTSCHAIN when pseudo-anonymity is desired.

Bernabe et al. (BERNABE et al., 2019) proposes eleven privacy-preserving challenges and four general solutions. We highlight “non-erasable data & on-chain data privacy” as the most relevant challenge to our proposal. Since blockchains tend to be immutable databases, the authors claim that even encrypted or hashed data are unsafe in the face of broken cryptographic primitives. That is even more complicated when referring to public blockchains, where data is publicly accessible by virtually anyone.

In this context, some blockchain-based implementations have proposed to store only public data in the blockchain and maintain private counterparts in an external database, such as the InterPlanetary File System (IPFS) (BENET, 2014). Moreover, some private blockchain networks present an integrated solution to the problem. For instance, Hyperledger Fabric offers “private collections”, where nodes maintain the public part of the data in the blockchain and the private portion on a local database where they can control which other nodes in the network may have access.

We remark that there exists a natural trade-off between privacy and transparency in our proposal due to the generation of evidence that shows the intent behind the signing of a document. Thus, if a signed document is contested in court, a robust monitoring system may help to prove that the signature was indeed generated by the original certificate holder. This logging framework can be augmented to include further user data to increase the amount of evidence for the receiver, such as IP addresses and identity provider tokens. However, we note that privacy is decreased with the amount of evidence provided.

Conversely, techniques that are able to obfuscate the data contained in the certificate, such as zero-knowledge proofs (FIEGE; FIAT; SHAMIR, 1987), or logging the hash of the certificate instead of itself, are able to improve privacy at the cost of monitoring capabilities. We note, however, that such techniques might damage monitoring to a degree where misissuance can no longer be detected and should be carefully considered before adoption in the context of our proposal.

Ultimately, digital certificates associate a key pair to an entity and thus are seen as public documents in the traditional PKI model. Logging certificates through our proposal facilitates document spread, a desirable property in many PKI implementations. We observe that the link between the entity and signature is made through the digital certificate itself; logging mechanisms only assist in spreading the public certificates. Thus, we argue that a PKI that requires pseudo-anonymity and wishes to incorporate logging and monitoring services should ensure anonymity directly in the certificates. While pseudo-anonymity may harm monitoring effectiveness, multiple uses of the same OTS private key are still prevented. Furthermore, if the pseudonym used in the certificate can be traced back to the user by a trusted party, monitoring is provided on a smaller scale.

PKI architects are free to choose the best way to implement the OTSCHAIN framework according to the particular purposes of the models. Infrastructures that wish to keep end-user information private can opt for a mix of permissioned blockchain and digital certificates with pseudonyms. Alternatively, those who do not wish or cannot create their private blockchain can utilize pre-established blockchains, such as Ethereum, to increase the spread and monitoring capabilities of their logs. Finally, as data protection regulations vary wildly among countries, there must be considerable care in logging implementation pursuant to local laws.

4.2 SECURITY BENEFITS

In this section, we discuss how OTSCHAIN facilitates the administration of several scenarios where participants misbehave, helping to detect or outright prevent situations in which the security of a PKI may be compromised. We remark that there are two main incentives for participants to behave in the traditional PKI model. Trustworthy participants have increased financial benefits, as opposed to those who have misbehaved, as the PKI is ultimately built on trust. Additionally, the threat of legal action may be imposed by regulators if misbehavior is frequent or severe enough. Below, we list several situations addressed more effectively by our model than by a traditional PKI.

CA misbehavior. OTSCHAIN is built on top of the CT framework; consequently, it has the same intrinsic set of benefits as conventional implementations of CT, i.e., the detection of certificate misissuance by CAs. Those who are found to be misbehaving can be, in addition to any legal action, revoked in the traditional sense and, in the case of blockchains, have their write permissions removed from the blockchain network. A CA is deemed malicious if it does not follow the guidelines established by the protocol: (i) it does not register any binding L_i when a signature is generated, or (ii) registers L_i with incorrect information.

We recall that, for a document to be validated using our proposal, an SVA must obtain the L_i tuple containing the link between the message hash and the matching digital certificate. Thus, refusing to register a certificate, logging the wrong certificate, or failing to produce the correct hash all lead to an invalid signed document. We argue that adapting existing SVAs to be pursuant to OTSCHAIN can be done efficiently, as it amounts to querying a blockchain network and performing a small number of additional checks.

Our proposal may also be extended to broader threat models. We consider signature backdating attacks, in which rogue CAs publish signed claims allegedly made in the past but never before disclosed. To address this situation, only a small modification to the function f is necessary: certificates are required to have their issuance time attribute to be “close enough” to the logging time. Indeed, this is crucial when protecting classical CAs from quantum attacks in the event that signed artifacts are held until a conventional signature algorithm becomes insecure.

Monitor misbehavior. A third-party rogue monitor may opt to respond to queries about certificates and documents wrongfully, to individually prevent the validation of a document, or to collude with a malicious CA to validate a document that has not been correctly logged in the blockchain. In OTSCHAIN, SVAs are free to either (i) choose a trustworthy monitor, (ii) query multiple monitors to form a consensus, or (iii) become a monitor themselves to ensure the correct response to a query.

End-user private key compromise. A major benefit of our proposal, when compared against traditional CT implementations, is that it includes the tracking of end-user documents in addition to CA actions. In particular, we are able to detect when end-user credentials are stolen without any input from the certificate holder whatsoever. In this manner, we are able to increase protection to the most exposed and broadest class of entities in a PKI while still preserving usability.

4.3 DEPLOYMENT & MEASUREMENTS

This section presents two deployment scenarios for our proposal and briefly discusses the storage, performance, and monetary costs of each scenario. We first consider a private and permissioned blockchain where only authorized nodes can read and write data. In the second scenario, we consider a public and permission-less alternative where virtually anyone can participate. In both scenarios, we propose the deployment of a smart contract called OTSchain-

SC, where nodes can read and write data in the blockchain, respectively triggering the registering and the monitoring procedures detailed at the beginning of this chapter.

4.4 PRIVATE BLOCKCHAIN

The Hyperledger Fabric platform is chosen as the blockchain for the first scenario. In this environment, participants cooperate as a consortium of CAs and other interested parties to deploy a private blockchain network. Thus, participants agree on the definition of $\mathsf{OTSchain\text{-}SC}$, which specifies the function f and how assets are read and written. Consider id_i as a unique identifier of the binding L_i. We set id_i to the digest of the public key pk_i in the certificate C_i, as we are interested in checking the existence of any L_i that already contains pk_i. Then, the tuple $(\mathsf{id}_i, \mathsf{L}_i)$ is implemented as an asset of $\mathsf{OTSchain\text{-}SC}$. We store certificate data in the well-known DER encoding to make full use of smart contract programming languages available in Hyperledger Fabric (e.g., Go and Java), which already have robust libraries to handle X.509 certificates.

Moreover, owing to the permissioned aspect of Fabric, we restrict which nodes are able to start the registration procedure, as to only allow CAs to log data in OTSCHAIN. This registration process will necessarily trigger the most important procedure of the smart contract, which is the execution of function f. To execute the monitoring phase of the protocol, participants query the blockchain by calling the $\mathsf{OTSchain\text{-}SC}$ monitoring procedure, which in our implementation accepts a set of attributes submitted as JSON-encoded key-value statements. For instance, to obtain the list of certificates issued by Brazilian CAs, a monitor must send the statement `{"issuer":{"C":"BR"}}` to the monitoring function, which returns a list of filter-matching assets.

4.5 PUBLIC BLOCKCHAIN

For the second deployment scenario, we propose to use the public and permissionless Ethereum blockchain. This network allows the deployment of decentralized applications, “dapps”, using the Solidity programming language. In this case, the network does not identify participants nor enables default restrictions of which nodes can read or write data in the blockchain. Thus, any participant in the Ethereum network would have access to data stored in OTSCHAIN by default; indeed, assets in this scenario are entirely public. Nevertheless, it is possible to restrict the logging of L_i to CAs by defining a set of addresses allowed to execute the registering procedure in $\mathsf{OTSchain\text{-}SC}$ smart contract.

In other words, even in the public implementation of our proposal, it is possible to restrict the register operation only to trusted CAs. The only requirement is that every CA would need to previously inform the smart contract of its address in the Ethereum network. The register procedure in the public version creates the same kind of asset defined in the previous scenario. However, the monitor function works differently from the Hyperledger Fabric version since

Solidity does not support standard libraries to process X.509 certificates. Consequently, all such management routines must be implemented as native functions in OTSCHAIN.

4.6 DISCUSSION

A critical aspect to consider when choosing a blockchain network are the positive and negative impacts of the underlying consensus protocol. In the literature, authors have already investigated the high energy consumption of consensus protocols (SEDLMEIR et al., 2020), such as proof of work, which is adopted by a wide range of blockchain networks like Bitcoin (NAKAMOTO, 2008). However, this is not necessarily the case for the current default implementation of, e.g., Ethereum and Hyperledger Fabric. The first recently migrated to a proof of stake consensus protocol, and the second uses a crash fault-tolerant protocol called Raft (ONGARO; OUSTERHOUT, 2014).

Furthermore, blockchain technology is constantly evolving, which means that any OTSCHAIN deployment, public or private, may require an update in the future, either to avoid known breaches of current blockchain network implementations or to take advantage of new features. In this context, our strategy could be easily adapted to any other blockchain network with support for smart contracts since it does not require any particular feature that only exists in the recommended deployment scenarios, i.e., Ethereum and Hyperledger Fabric. In the migration process, in order to share data between networks, data-sharing strategies and sharding (LUU et al., 2016) may be employed.

We now turn to the practical consequences of OTSCHAIN. Storage costs are heavily dependent on the number and size of issued certificates. In our model, the number of certificates issued is exactly the number of digital signatures generated. Therefore, we estimate storage costs according to such usage statistics. To better reflect the feasibility of our framework, we consider several quantum-resistant signature schemes for CAs as chosen by NIST in its standardization process (ALAGIC et al., 2022) and two real-world PKIs as the basis of our analysis: Brazil, with approximately $\sigma_{\mathsf{low}} = 2^{24}$ signatures per year (Instituto Nacional de Tecnologia da Informação, 2022); and Estonia, with approximately $\sigma_{\mathsf{high}} = 2^{27}$ signatures per year (KUIK, 2022).

The size of a single tuple $\mathsf{L}_i = (\mathsf{C}_i, \mathsf{H}_i)$ depends on algorithm choice. We hereafter consider Winternitz to be the choice for end users due to the previously mentioned benefits. Winternitz has a public key size of $\frac{n}{8}$ bytes; let ψ be the size of the signature made by the CA. We let κ be the certificate "header" size in bytes, i.e., any attributes and necessary encoding, apart from the public key and signature. Then, the size of L_i is precisely $(\kappa + \psi + \frac{n}{8}) + \frac{n}{8}$ bytes. By separating the public key, signature, and other data in C_i, we are able to estimate the size of L_i considering several scenarios for κ and ψ. We define $\kappa_{\mathsf{min}} = 128$ for a minimal certificate with barely any semantic information, $\kappa_{\mathsf{avg}} = 512$ for a certificate with few attributes, and $\kappa_{\mathsf{max}} = 1536$ for a certificate with liberal use of extensions, attributes, and names. Our findings are summarized in Table 4.

We observe that the storage cost of L_i is dominated by the signature size ψ. Moreover,

Table 4 – Uncompressed storage size estimation of a blockchain containing bindings between certificates and hashes, in GB/year, considering various security levels n, several quantum-resistant signature algorithms and parameters, certificate sizes κ and two contrasting signature usage scenarios σ_{low} and σ_{high}.

n	Algorithm	σ_{low}			σ_{high}		
		κ_{min}	κ_{avg}	κ_{max}	κ_{min}	κ_{avg}	κ_{max}
256	Falcon-512	13.41	19.41	35.41	107.25	155.25	283.25
	Dilithium-2	40.81	46.81	62.81	326.50	374.50	502.50
	SPHINCS$^+$-128s	125.75	131.75	147.75	1006.00	1054.00	1182.00
384	Dilithium-3	54.95	60.95	76.95	439.62	487.62	615.62
	SPHINCS$^+$-192s	257.00	263.00	279.00	2056.00	2104.00	2232.00
512	Falcon-1024	24.00	30.00	46.00	192.00	240.00	368.00
	Dilithium-5	75.80	81.80	97.80	606.38	654.38	782.38
	SPHINCS$^+$-256s	469.50	475.50	491.50	3756.00	3804.00	3932.00

storage may be a problem in the long term, as the blockchain grows endlessly if the network does not decide on archival procedures. In the private scenario, creating a new instance of OTSCHAIN from time to time is allowed since every new instance of the smart contract can have its own blockchain in the Hyperledger Fabric network. Older chains can be kept at least until every certificate registered has expired. Hence, the network may agree on a size threshold to determine when to migrate to a new smart contract instance. This solution cannot be applied to the public version because Ethereum has only one chain of blocks for every smart contract created.

Furthermore, in the Ethereum network, storage is directly connected to monetary costs due to *gas* transaction costs, which in turn, reflect the amount of Ether spent to process an operation. For example, consider the implementation of step R5 in Figure 6, where the smart contract simply stores an array of assets. The *gas* cost is calculated using the metrics specified by Kostamis et al. (KOSTAMIS; SENDROS; EFRAIMIDIS, 2021). Considering Falcon-512 as the algorithm with the smallest ψ, OTSCHAIN requires 617 700 *gas* for κ_{min} and 1 590 100 *gas* for κ_{high}. We remark that there exists an upper bound of *gas* spent per block; Kostamis et al. set this bound to 8 000 000, which may be increased by the network in the future. Either way, algorithms with large ψ such as SPHINCS$^+$-192s, with a cost of 11 380 400 *gas* for κ_{min}, are generally unsuited to be used in Ethereum.

Another relevant aspect that must be touched on is performance. In the Hyperledger Fabric version, writing data in the blockchain strongly depends on the network configuration. More precisely, the response of endorsing nodes and the execution of the Raft consensus protocol define the time required to process a transaction. On the other hand, in the Ethereum implementation, writing data in the blockchain depends not only on the network configuration but also on the fees miners require to execute transactions. In general, higher fees increase the chances of a transaction being accepted in a new block.

The performance cost of reading data depends on the complexity of the query function since monitors may be nodes on the network independent of the deployment scenario. Therefore, we consider that reading data in the blockchain would be similar to reading data from a standard database, i.e., with minimal overhead to SVAs. Finally, we highlight that the proposal is feasible in Hyperledger Fabric and in the Ethereum network. However, the second option suffers primarily from the monetary aspect. Modifying OTSCHAIN to store $\mathsf{L}_i = (\mathcal{H}(\mathsf{C}_i), \mathsf{H}_i)$ would reduce these requirements for storage constrained systems, at the expense of deteriorating the monitoring capabilities of the framework.

5 FINAL REMARKS

Private key management and certificate revocation have been sources of complexity in PKIs since their inception. In this work, we introduced a new key management model capable of solving these issues at the end-user level through the unique binding of a digital certificate and a document using our one-time certificates. We show that many artifacts responsible for additional complexity in PKIs are no longer needed in our proposal, and security challenges associated with private key compromises and stale attributes are prevented in our model.

We maintain usual signature policy requirements when signing documents using OTCs while simultaneously greatly improving the performance of validation procedures. Trade-offs between usability and the performance of document creation can be lessened through clever algorithm choices and key generation strategies. These features enable the creation of a simple, efficient, and inclusive PKI, built for the rapid adoption of digitally signed electronic documents by laypeople, with several usability features and robust security properties.

The existence of rogue CAs, end-user credential theft, and human error are some of the problems a PKI implementation faces. We address the primary challenge associated with these issues, i.e., their swift detection, using OTSCHAIN. We define a framework that expands on the benefits of the CT protocol via the inclusion of a blockchain network while maintaining the certificate misissuance detection properties of CT. Our rationale is to include the monitoring of end-user signatures in an accessible and transparent manner by shifting the logging process to CAs. Furthermore, we enable the secure use of OTS schemes through semantic checks processed by smart contracts. We discuss the privacy and security implications of registering end-user certificates bound to message hashes in the blockchain. Storage cost and performance estimations show that our model is feasible and compatible with current PKI implementations, maintaining security while increasing usability to end-users and acting as a first responder.

5.1 FUTURE WORK

There exist several blockchain implementations with specific characteristics such that one cannot easily predict how they behave when paired with OTSCHAIN. Therefore, we suggest searching for an optimal selection of blockchain for each use case, such as specific consensus algorithms, throughput, and privacy requirements. Similarly, investigating advanced mechanisms such as blockchain sharding may improve the performance and scalability aspects of the deployment. Moreover, the function f may be augmented to include other semantic checks, such as a proxy signature enabler or adaptation of the framework to accept few-time signature schemes by allowing a specific number of uses of the private key. We note, however, that modifying f requires careful consideration as it might cause privacy and security implications.

Furthermore, the behavior of OTCs with quantum-safe signature schemes and hybrid certificates can also be explored; we anticipate that the size of public keys and signatures and the performance of operations on signatures will present unique challenges. However, just like

OTSCHAIN, one-time signatures seem well suited to these types of certificates and may be explored. Finally, large communities, such as countries with populations in the tens of millions, might see a sizable cost reduction in deploying and providing affordable digital signatures of electronic documents to citizens such that a monetary cost analysis may be explored. These savings come mainly from the inessential provision of private key containers for the populace and less management overhead at the final CA level due to revocation-less certificates.

While our model does not directly demand the use of IdPs in place of RAs, a survey on the benefits of different identity management frameworks, RAs, and systems may be conducted to pinpoint where each framework best serves its users. Finally, while the changes proposed to the validation process are straightforward, a formal analysis of the protocols may be conducted through protocol verification tools such as Tamarin (MEIER et al., 2013) or AVISPA (ARMANDO et al., 2005).

REFERENCES

AAS, J. et al. Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web. In: **CCS '19: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security**. [S.l.: s.n.], 2019. p. 2473–2487.

AAS, J.; GRAN, S. **Let's Encrypt Has Issued a Billion Certificates**. 2020. Disponível em: https://archive.ph/XkA2f.

AL-RIYAMI, S. S.; PATERSON, K. G. Certificateless Public Key Cryptography. In: **Advances in Cryptology - ASIACRYPT 2003**. [S.l.: s.n.], 2003. (Lecture Notes in Computer Science, v. 2894), p. 452–473.

ALAGIC, G. et al. **Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process**. [S.l.], 2022.

ARMANDO, A. et al. The avispa tool for the automated validation of internet security protocols and applications. In: SPRINGER. **Computer Aided Verification: 17th International Conference, CAV 2005, Edinburgh, Scotland, UK, July 6-10, 2005. Proceedings 17**. [S.l.], 2005. p. 281–285.

ÅRNES, A. Public key certificate revocation schemes. In: CITESEER. **Masters thesis**. [S.l.], 2000.

BARKER, E. **Recommendation for Key Management: Part 1 – General**. [S.l.], 2020.

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

BELLARE, M.; MINER, S. K. A forward-secure digital signature scheme. In: SPRINGER. **Annual international cryptology conference**. [S.l.], 1999. p. 431–448.

BENET, J. **IPFS - Content Addressed, Versioned, P2P File System**. 2014. ArXiv:1407.3561v1.

BERNABE, J. B. et al. Privacy-Preserving Solutions for Blockchain: Review and Challenges. **IEEE Access**, v. 7, p. 164908–164940, out. 2019.

BOEYEN, S. et al. **Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile**. [S.l.], 2008.

BOUR, G. et al. On the Certificate Revocation Problem in the Maritime Sector. In: **NordSec 2020: Secure IT Systems**. [S.l.: s.n.], 2020. (Lecture Notes in Computer Science, v. 12556), p. 142–157.

BRASIL. Lei nº 13.079, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). **Diário Oficial da União**, v. 157, n. 1, p. 59–64, ago. 2018. Disponível em: https://legislacao.presidencia.gov.br/atos/?tipo=LEI&numero=13709&ano=2018&ato=293QzZ61UeZpWT79e.

BUCHMANN, J. et al. On the Security of the Winternitz One-Time Signature Scheme. In: **Progress in Cryptology – AFRICACRYPT 2011**. [S.l.: s.n.], 2011. (Lecture Notes in Computer Science, v. 6737), p. 363–378.

CERENIUS, D. et al. Trust issue (r) s: Certificate revocation and replacement practices in the wild. 2024.

CHONG, K. W.; KIM, Y. S.; CHOI, J. A Study of Factors Affecting Intention to Adopt a Cloud-Based Digital Signature Service. **Information**, v. 12, n. 2, jan. 2021. Article id 60.

COMODO. **Comodo Report of Incident - Comodo detected and thwarted an intrusion on 26-MAR-2011**. 2011. Disponível em: https://archive.is/35RNv.

COOPER, D. A. A Model of Certificate Revocation. In: **Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)**. [S.l.: s.n.], 1999. p. 256–264.

DAMGåRD, I. et al. Secure Key Management in the Cloud. In: **IMACC 2013: Cryptography and Coding**. [S.l.: s.n.], 2013. (Lecture Notes in Computer Science, v. 8308), p. 270–289.

DODS, C.; SMART, N. P.; STAM, M. Hash Based Digital Signature Schemes. In: **Cryptography and Coding 2005: Cryptography and Coding**. [S.l.: s.n.], 2005. (Lecture Notes in Computer Science, v. 3796), p. 96–115.

ETSI. **Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures**. [S.l.], 2021.

ETSI. **Electronic Signatures and Infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation**. [S.l.], 2021.

European Parliament and the Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016. **Official Journal of the European Union**, L, n. 119, p. 1–88, maio 2016.

FIEGE, U.; FIAT, A.; SHAMIR, A. Zero knowledge proofs of identity. In: **Proceedings of the nineteenth annual ACM symposium on Theory of computing**. [S.l.: s.n.], 1987. p. 210–217.

GARBA, A. et al. BB-PKI: Blockchain-Based Public Key Infrastructure Certificate Management. In: **2020 IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)**. [S.l.: s.n.], 2020. p. 824–829.

GOLDREICH, O. **Foundations of Cryptography: Volume 2, Basic Applications**. 1st. ed. [S.l.]: Cambridge University Press, 2004. ISBN 9780521830843.

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

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

GUTMANN, P. PKI: It's Not Dead, Just Resting. **Computer**, v. 35, n. 8, p. 41–49, ago. 2002.

GUTMANN, P. Simplifying public key management. **Computer**, v. 37, n. 2, p. 101–103, fev. 2004.

HAN, K.; HWANG, S. O. A PKI without TTP based on conditional trust in blockchain. **Neural Computing and Applications**, v. 32, n. 17, p. 13097–13106, set. 2020.

HELMICH, P. Public key infrastructures: A panacea solution? **Network Security**, Elsevier, v. 2000, n. 5, p. 8–11, 2000.

HOFFMAN, P. E.; SCHAAD, J. **New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)**. [S.l.], 2010.

HYLA, T.; PEJAś, J. Long-term verification of signatures based on a blockchain. **Computers & Electrical Engineering**, jan. 2020. Article id 106523.

Hyperledger Foundation. **An Introduction to Hyperledger**. 2018. Disponível em: https://www.hyperledger.org/wp-content/uploads/2018/08/HL_Whitepaper_IntroductiontoHyperledger.pdf.

HüLSING, A. et al. **XMSS: eXtended Merkle Signature Scheme**. [S.l.], 2018.

HüLSING, A.; KUDINOV, M. Recovering the Tight Security Proof of SPHINCS$^+$. In: **ASIACRYPT 2022: Advances in Cryptology – ASIACRYPT 2022**. [S.l.: s.n.], 2022. (Lecture Notes in Computer Science, v. 13794), p. 3–33.

Instituto Nacional de Tecnologia da Informação. **Relatório de Gestão 2022**. 2022. Disponível em: https://www.gov.br/iti/pt-br/acesso-a-informacao/transparencia-e-prestacao-de-contas/relatorios-de-gestao/relatoriodegesta2022v2.pdf.

ITKIS, G.; REYZIN, L. Forward-secure signatures with optimal signing and verifying. In: SPRINGER. **Advances in Cryptology—CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19–23, 2001 Proceedings 21**. [S.l.], 2001. p. 332–354.

KAMPANAKIS, P. et al. **The Viability of Post-quantum X.509 Certificates**. 2018. Cryptology ePrint Archive, Paper 2018/063. Disponível em: https://eprint.iacr.org/archive/2018/063/20180127:042741.

KOCHER, P. C. On Certificate Revocation and Validation. In: **FC 1998: Financial Cryptography**. [S.l.: s.n.], 1998. (Lecture Notes in Computer Science, v. 1465), p. 172–177.

KOHNFELDER, L. M. **Towards a Practical Public-key Cryptosystem**. Dissertação (Bachelor's thesis) — Massachusetts Institute of Technology, maio 1978.

KOSTAMIS, P.; SENDROS, A.; EFRAIMIDIS, P. Exploring Ethereum's Data Stores: A Cost and Performance Comparison. In: **2021 3rd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)**. [S.l.: s.n.], 2021. p. 53–60.

KUBILAY, M. Y.; KIRAZ, M. S.; MANTAR, H. A. KORGAN: An Efficient PKI Architecture Based on PBFT Through Dynamic Threshold Signatures. **The Computer Journal**, v. 64, n. 4, p. 564–574, abr. 2021.

KUIK, S. **In 20 years, more than 800 million digital signatures have been given in Estonia | RIA**. 2022. Disponível em: https://www.ria.ee/en/news/20-years-more-800-million-digital-signatures-have-been-given-estonia.

KUMAR, R.; BHATIA, M. P. S. A Systematic Review of the Security in Cloud Computing: Data Integrity, Confidentiality and Availability. In: **2020 IEEE International Conference on Computing, Power and Communication Technologies (GUCON)**. [S.l.: s.n.], 2020. p. 334–337.

KUTYłOWSKI, M.; BłAśKIEWICZ, P. Advanced Electronic Signatures and eIDAS – Analysis of the Concept. **Computer Standards & Interfaces**, v. 83, jan. 2023. Article id 103644.

KUZMINYKH, I.; GHITA, B.; SHIAELES, S. Comparative Analysis of Cryptographic Key Management Systems. In: **NEW2AN 2020, ruSMART 2020: Internet of Things, Smart Spaces, and Next Generation Networks and Systems**. [S.l.: s.n.], 2020. (Lecture Notes in Computer Science, v. 12526), p. 80–94.

LAURIE, B. et al. **Certificate Transparency Version 2.0**. [S.l.], 2021.

LERMAN, L.; MARKOWITCH, O.; JR, J. N. Key Management as a Service. In: **Proceedings of the International Conference on Security and Cryptography (SECRYPT-2012)**. [S.l.: s.n.], 2012. p. 276–281.

LI, Z. et al. Pistis: Issuing Trusted and Authorized Certificates With Distributed Ledger and TEE. **IEEE Transactions on Parallel and Distributed Systems**, v. 33, n. 7, p. 1636–1649, jul. 2022.

LONGO, E.; STAPLETON, J. **Smart Cards**. [S.l.], 2002.

LUU, L. et al. A secure sharding protocol for open blockchains. In: **Proceedings of the 2016 ACM SIGSAC conference on computer and communications security**. [S.l.: s.n.], 2016. p. 17–30.

MADALA, D. S. V.; JHANWAR, M. P.; CHATTOPADHYAY, A. Certificate Transparency Using Blockchain. In: **2018 IEEE International Conference on Data Mining Workshops (ICDMW)**. [S.l.: s.n.], 2018. p. 71–80.

MAHANTA, H. J.; AZAD, A. K.; KHAN, A. K. Power analysis attack: A vulnerability to smart card security. In: IEEE. **2015 International Conference on Signal Processing and Communication Engineering Systems**. [S.l.], 2015. p. 506–510.

MAYR, L. et al. Monitoring key pair usage through distributed ledgers and one-time signatures. **Information**, MDPI, v. 14, n. 10, p. 523, 2023.

MAYR, L. et al. **One-Time Certificates for Reliable and Secure Document Signing**. 2024. Disponível em: https://arxiv.org/abs/2208.03951.

MEIER, S. et al. The tamarin prover for the symbolic analysis of security protocols. In: SPRINGER. **Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25**. [S.l.], 2013. p. 696–701.

MENEZES, A. J.; OORSCHOT, P. C. van; VANSTONE, S. A. **Handbook of Applied Cryptography**. 1st. ed. [S.l.]: CRC Press, 1996. ISBN 9780849385230.

MEULEN, N. van der. DigiNotar: Dissecting the First Dutch Digital Disaster. **Journal of Strategic Security**, v. 6, n. 2, p. 46–58, jun. 2013.

MUSHTAQ, M. F. et al. Cloud Computing Environment and Security Challenges: A Review. **International Journal of Advanced Computer Science and Applications**, v. 8, n. 10, p. 183–195, out. 2017.

NAKAMOTO, S. **Bitcoin: A Peer-to-Peer Electronic Cash System**. 2008. Disponível em: https://bitcoin.org/bitcoin.pdf.

NAOR, M.; NISSIM, K. Certificate Revocation and Certificate Update. In: **SSYM'98: Proceedings of the 7th conference on USENIX Security Symposium**. [S.l.: s.n.], 1998. Article id 17.

ONGARO, D.; OUSTERHOUT, J. In search of an understandable consensus algorithm. In: **2014 USENIX annual technical conference (USENIX ATC 14)**. [S.l.: s.n.], 2014. p. 305–319.

PADILHA, R.; PEDONE, F. Confidentiality in the Cloud. **IEEE Security & Privacy**, v. 13, p. 57–60, jan. 2015.

PALMA, L. M. et al. Blockchain and smart contracts for higher education registry in brazil. **International Journal of Network Management**, Wiley, v. 29, n. 3, jan. 2019. Disponível em: https://doi.org/10.1002/nem.2061.

Ponemon Institute. **2021 Global Encryption Trends Study**. [S.l.], 2021.

RAYA, M. et al. **Certificate Revocation in Vehicular Networks**. [S.l.], 2006.

RIVEST, R. L. Can We Eliminate Certificate Revocation Lists? In: **FC 1998: Financial Cryptography**. [S.l.: s.n.], 1998. (Lecture Notes in Computer Science, v. 1465), p. 178–183.

RIVEST, R. L.; LAMPSON, B. **SDSI – A Simple Distributed Security Infrastructure**. [S.l.], 1996.

SALYARDS, J. **Scourge of the Betrayer**. 1st. ed. [S.l.]: Night Shade, 2012. ISBN 9780521830843.

SANTESSON, S. et al. **X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP**. [S.l.], 2013.

SCHEIBELHOFER, K. PKI without Revocation Checking. In: **4th Annual PKI R&D Workshop "Multiple Paths to Trust" Proceedings**. [S.l.: s.n.], 2005. p. 52–65.

SEDLMEIR, J. et al. The energy consumption of blockchain technology: Beyond myth. **Business & Information Systems Engineering**, Springer Science and Business Media LLC, v. 62, n. 6, p. 599–608, jun. 2020. Disponível em: https://doi.org/10.1007/s12599-020-00656-x.

SHOR, P. 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.

TOPALOVIC, E. et al. Towards Short-Lived Certificates. In: **Proceedings of the IEEE Workshop on Web 2.0 Security & Privacy 2012 (W2SP 2012)**. [S.l.: s.n.], 2012.

VIVES, S. J. Synced Hash-Based Signatures: Post-Quantum Authentication in a Blockchain. In: **Actas del IX Congreso Iberoamericano de Seguridad Informática**. [S.l.: s.n.], 2017.

WANG, Q.; GAO, D.; CHEN, D. Certificate Revocation Schemes in Vehicular Networks: A Survey. **IEEE Access**, v. 8, p. 26223–26234, jan. 2020.

WOHLMACHER, P. Digital Certificates: A Survey of Revocation Methods. In: **MULTIMEDIA '00: Proceedings of the 2000 ACM workshops on Multimedia**. [S.l.: s.n.], 2000. p. 111–114.

WOOD, G. **Ethereum: a Secure Decentralised Generalised Transaction Ledger**. 2022. Berlin version beacfbd. Disponível em: https://ethereum.github.io/yellowpaper/paper.pdf.

XU, L. et al. KCRS: A Blockchain-Based Key Compromise Resilient Signature System. In: **BlockSys 2019: Blockchain and Trustworthy Systems**. [S.l.: s.n.], 2019. (Communications in Computer and Information Science, v. 1156), p. 226–239.

YUNAKOVSKY, S. E. et al. Towards security recommendations for public-key infrastructures for production environments in the post-quantum era. **EPJ Quantum Technology**, Springer Berlin Heidelberg, v. 8, n. 1, p. 14, 2021.

ZHENG, P. Tradeoffs in Certificate Revocation Schemes. **ACM SIGCOMM Computer Communication Review**, v. 33, n. 2, p. 103–112, abr. 2003.

ÅRNES, A. **Public Key Certificate Revocation Schemes**. Dissertação (Mestrado) — Norwegian University of Science and Technology, fev. 2000.