UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
SISTEMAS DE INFORMAÇÃO

Gabriel Aristeu Cabral

Assinatura de Documentos com Certificado de Uso Único e Chaves Geradas no Cliente

Florianópolis
2023

Gabriel Aristeu Cabral

Assinatura de Documentos com Certificado de Uso Único e Chaves Geradas no Cliente

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Sistemas da Informação e aprovado em sua forma final pelo curso de Graduação em Sistemas de Informação.

Florianópolis, 14 de Junho de 2023.

Banca Examinadora:

Prof. Ricardo Custódio, Dr.
Orientador
Universidade Federal de Santa Catarina

Lucas Mayr de Athayde
Avaliador

Frederico Schardong
Avaliador

Thaís Bardini Idalino
Avaliador

Gabriel Aristeu Cabral

Assinatura de Documentos com Certificado de Uso Único e Chaves Geradas no Cliente

Trabalho de Conclusão de Curso submetido ao Curso de Graduação em Sistemas de Informação do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito para obtenção do título de Bacharel em Sistemas da Informação.
Orientador: Prof. Ricardo Custódio, Dr.
Coorientador: Lucas Mayr de Athayde

Florianópolis
2023

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

Cabral, Gabriel Aristeu
Assinatura de Documentos com Certificado de Uso Único e Chaves Geradas no Cliente / Gabriel Aristeu Cabral ; orientador, Ricardo Felipe Custódio, coorientador, Lucas Mayr de Athayde, 2023.
68 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Sistemas de Informação, Florianópolis, 2023.

Inclui referências.

1. Sistemas de Informação. 2. Assinatura Digital. 3. Infraestrutura de chave públicas. 4. Certificados Digitais. 5. Certificados de Uso Único. I. Custódio, Ricardo Felipe. II. Athayde, Lucas Mayr de. III. Universidade Federal de Santa Catarina. Graduação em Sistemas de Informação. IV. Título.

AGRADECIMENTOS

Os agradecimentos principais são dedicados a todos os professores que contribuíram para minha formação, em especial a toda a equipe do LabSec que me ajudou neste trabalho. Destaco um agradecimento ao Prof. Custódio pela oportunidade de realização deste trabalho assim como ao Lucas Mayr por toda a contribuição na elaboração do mesmo.

Outros agradecimento especiais são direcionados aos meus pais, que me auxiliaram desde o momento em que nasci até hoje. Assim como um agradecimento a Diovana Rodrigues e Artur Carmezini e todos os amigos que fiz na duração do curso. E também a Camile Cerezolli, que esteve ao meu lado e me deu todo o suporte possível nos últimos anos.

Can’t keep my eyes from the circling skies
Tongue-tied and twisted, just an earth-bound misfit, I
Pink Floyd

RESUMO

Com o avanço tecnológico e digitalização de diversos serviços mundialmente se faz necessário democratizar o acesso a estes serviços, entre eles as assinaturas digitais. Mesmo se tratando de um serviço moderno as assinaturas digitais introduzem uma barreira de conhecimento quanto ao seu uso correto. Esta barreira de conhecimento existe principalmente devido à maneira como a solução foi pensada. Com o aumento da popularidade de assinaturas não houve um aumento no nível de informatização de seus usuários, com os mesmos sendo encarregados de manter seus certificados e chaves privadas em dispositivos como *smart cards* ou no disco rígido de seus computadores pessoais. Entretanto, muitas vezes um usuário pode não saber o procedimento correto ao ocorrer um extravio ou perca de seu certificado, e mesmo comunicando da maneira correta, o jeito com que a solução de revogação é realizada pode permitir que assinaturas falsas sejam feitas em nome desta pessoa. Muitas aplicações de assinaturas digitais, realizam a assinatura em um servidor, com a necessidade de *upload* do arquivo para uma nuvem, o que não é uma boa prática logo que não há maneira dos usuários confirmarem que o servidor não guardou uma cópia do arquivo assinado, tendo assim a sua privacidade invadida. Tendo em vista esta problemática propõe-se a criação de uma aplicação WEB para realização de assinaturas digitais usando certificados de uso único em documentos no formato PDF, os certificados não precisam ser mantidos pelos usuários finais e podem ser descartados ao fim do processo de assinatura. Está aplicação também realizará todo processo de assinatura no *browser* onde a aplicação WEB está rodando, com o documento nunca sendo enviado a um servidor, preservando assim a privacidade do mesmo. A ideia é de que o processo de assinatura seja simples fácil e rápido, com o usuário inserindo o arquivo na aplicação *WEB* e clicando um botão de assinatura, após isto o arquivo assinado seria devolvido ao usuário, e as primitivas criptográficas e certificado utilizados são descartados. Isto diminuiria a barreira de conhecimento, necessidade de manutenção dos certificados pelos usuários finais e conhecimento de processos no caso de extravio ou perca do certificado, consequentemente aumentando o acesso da população às assinaturas digitais.

Palavras-chave: Assinatura Digital. Infraestrutura de chaves públicas. Certificados digitais. Certificados de Uso Único.

LISTA DE FIGURAS

LISTA DE SÍMBOLOS

$\cup$	Concatenação
$=$	Atribuição
$==$	Igualdade

LISTINGS

SUMÁRIO

1 INTRODUÇÃO

Os sistemas de assinatura digital hoje necessitam que o usuário possua certo conhecimento sobre alguns conceitos como criptografia assimétrica, criação e revogação de certificados. Assuntos que não fazem parte do currículo de grande parte da população brasileira. Ao utilizar certificados de uso único para assinatura digital de documentos pode-se remover parte desta barreira.

1.1 CONTEXTO

Uma característica muito importante no contexto de assinatura digital é a Infraestrutura de Chaves Públicas (ICP, PKI), ela é responsável pela distribuição, revogação e manutenção de certificados digitais e criptografia de chaves públicas. Um exemplo de ICP é a ICP-Brasil, responsável pelo ecossistema de certificados brasileiros. Por mais que a assinatura digital seja um fim popular, certificados podem ter outras utilidades como, por exemplo, a confirmação de autenticidade de websites. Neste trabalho só será tratado do uso de certificados para a assinatura digital. Assinaturas digitais utilizam de criptografia assimétrica clássica para realizar a verificação de quem assinou um documento.

Na criptografia assimétrica clássica se tem um par de chaves sendo uma pública e outra privada, a ideia principal é que tudo que uma chave cifra a outra chave decifra e vice-versa. E enquanto a chave pública é aberta e conhecida por todos, a privada só pode ser conhecida pelo usuário que a possui. Logo para assinatura digital um usuário cifra com a chave privada e qualquer um pode verificar aquela assinatura utilizando a chave pública do assinador para decifrar. Com certificados de uso único é removida a necessidade do usuário guardar sua chave privada, gerando um par de chaves e um certificado para cada assinatura digital que for feita, com o certificado sendo válido apenas para aquele documento.

1.2 PROBLEMA

Tendo em vista o contexto atual de assinatura digital, pode-se compreender que a principal problemática das ICPs atuais é garantir a segurança da chave privada. Logo que muitas vezes os usuários finais são pessoas que não possuem conhecimento da importância das chaves privadas e negligenciam o seu armazenamento. Além de que os dispositivos para o armazenamento de chaves como Tokens e Smart Cards possuem um custo e requerem que o usuário tenha conhecimento de como acontece a revogação em caso de perda ou vazamento do conteúdo dos dispositivos. Outro problema é a privacidade do documento logo que muitas vezes a assinatura ocorre em um servidor remoto não sendo possível garantir que o sistema não guardou uma cópia do mesmo ao final do processo de assinatura.

1.3 MÉTODO DE PESQUISA

Os passos para o desenvolvimento deste projeto podem ser elencados por:

- Escolha de ferramentas para a elaboração do Front-End.
- Estudo do processo de assinatura de documentos digitais assim como a ICP.
- Construção do protótipo e implementação do algoritmo de assinatura.
- Estudo de métricas e limites da aplicação.

1.4 MOTIVAÇÃO

O aumento da digitalização no mundo trouxe a tona a necessidade de democratizar o acesso às tecnologias tidas como estado da arte de forma que as mesmas possam ser uteis a população, este trabalho visa diminuir a barreira de conhecimento necessária para uma pessoa realizar a assinatura de documentos digitais. Assim como fornecer a privacidade que o documento assinado nunca foi transmitido a nenhum servidor.

1.5 ESCOPO

O escopo deste trabalho é a criação de uma aplicação que realize assinaturas digitais em documentos no formato PDF, no *browser* em que a aplicação está rodando, sem a necessidade de *upload* do arquivo a algum servidor e utilizando certificados digitais de uso único.

1.6 CONTRIBUIÇÕES

A maior contribuição deste trabalho é a aplicação que realiza assinaturas digitais diretamente no *browser* utilizando certificados de uso único, assim como o relatório que explica diversos conceitos para um entendimento mais profundo de assinaturas digitais.

1.7 OBJETIVOS

Desenvolver um sistema de assinatura de documentos digitais de maneira a simplificar a barreira de conhecimento de usuários finais assim como garantir a segurança e autenticidade dos documentos assinados por um esquema de certificado de uso único.

1.7.1 Objetivos Específicos

- Desenvolver uma aplicação que utilize certificados de uso único, gerados por uma AC, para assinatura de documentos digitais;

- Garantir que a assinatura se dê no local de execução da aplicação e o documento a ser assinado não seja enviado a nenhum servidor, e que após uso na assinatura os artefatos criptográficos são descartados;
- Garantir que a assinatura ocorre em um tempo aceitável para o usuário final;
- Realizar testes de performance para diferentes algoritmos de assinatura;

1.8 ESTRUTURA DO TRABALHO

Este trabalho está divido em 5 partes, na introdução foi apresentado a problemática assim como os objetivos definidos para este trabalho. Na seção de Fundamentação Teórica são apresentados conceitos-chave para uma compreensão adequada da proposta. Já na seção *Portable Document File* (PDF) é explicado em maior detalhe a estrutura de um documento PDF assim como o processo de assinatura digital no mesmo. Na seção de Desenvolvimento, é explicada a proposta e como a mesma foi desenvolvida assim como experimentos realizados na mesma. Já na última seção de Conclusão e Trabalhos Futuros é feita uma reflexão sobre a proposta desenvolvida e os resultados obtidos em experimentos buscando entender se os objetivos definidos na Introdução foram alcançados.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são introduzidos os conceitos necessários para o embasamento teórico e compreensão da proposta.

2.1 CRIPTOGRAFIA ASSIMÉTRICA

A criptografia assimétrica é um conceito básico de segurança no mundo atual. Ela é utilizada para a navegação segura em páginas *WEB* com protocolos como o *Hypertext Transfer Protocol Secure (HTTPS)* e o *Secure Shell (SSH)*. A criptografia assimétrica consiste em utilizar um par de chaves para realizar a transmissão segura de uma mensagem, onde o que uma chave cifra a outra decifra e vice-versa, logo é possível deixar uma chave pública e de conhecimento de todos e outra privada da qual apenas o dono tem acesso (DIFFIE; HELLMAN, 1976).

Desde a criação do conceito de criptografia assimétrica, diversos métodos de criptografia assimétrica surgiram, segundo (STANDARDS; TECHNOLOGY, 2013) os algoritmos seguros para serem utilizados em aplicações são o DSA (STANDARDS; TECHNOLOGY, 2013), RSA (RIVEST; SHAMIR; ADLEMAN, 1978) e ECDSA (FAROOQ; HUSSAIN; USTUN, 2019), estes algoritmos também acabam sendo os mais utilizados hoje na internet. Entretanto, estes algoritmos possuem diferentes processos de criação de chave, assinatura, especificidades e diferentes desempenhos quando utilizados em diferentes aplicações e em diversos contextos (MERKLE..., 2023).

É possível comprovar a autenticidade de uma mensagem cifrada usando criptografia assimétrica realizando o processo de deciframento com o par de chaves de quem cifrou. Logo que esta mensagem cifrada pela chave privada só será decifrada com a chave pública do respectivo par. O uso de diferentes chaves de cifragem e decifragem num cripto-sistema é algo bom, tendo em vista que sistemas assimétricos permitem que a chave de cifragem seja pública e disponível a todos enquanto a de decifragem é restrita apenas aos que tem acesso e podem decifrar a informação (COUNCIL, 1991).

Defini-se que a criptografia assimétrica de chaves públicas são computadas uma chave de cifra EK e uma chave de deciframento DK, onde EK é o inverso de DK e é computacionalmente impossível computar DK obtendo EK. Permitindo deixar EK pública para qualquer pessoa que queira mandar uma mensagem que será decifrada com EK, logo um usuário poderia gerar esse par de chaves e publicar sua EK em um diretório público para receber mensagens que ninguém poderá decifrar além dele (DIFFIE; HELLMAN, 1976).

2.2 FUNÇÕES DE RESUMO CRIPTOGRÁFICO

Defini-se como uma Função de Resumo Criptográfico o seguinte. Um valor h é gerado for uma função H na forma: $h = H(M)$. Para diferentes tamanhos M o tamanho de h é fixo. Dado um h é computacionalmente impossível derivar M. Também é impossível achar um

par (x,y) onde $H(x) == H(y)$. Logo uma função de Resumo Criptográfico gera um resumo criptográfico para bytes inseridos, sendo possível confirmar a integridade de uma mensagem verificando seu resumo (STALLINGS, 2013). As funções de Resumo Criptográfico possuem justamente a função de confirmar que não houve modificação em um conteúdo, logo podem ser adicionadas ao final de uma mensagem m. Calcula-se então $hm = H(m)$, é construída então uma mensagem final mf, tendo em vista que α delimita o fim da mensagem m, na seguinte forma $mf = m \cup \alpha \cup hm$. O receptor de mf pode então separar a mensagem em α e obter os valores de (m', hm). Calcula-se $rhm = H(m')$ sendo possível comprovar que não houve alteração do conteúdo m com $rhm == hm$, logo que se houve alteração os valores do resumo calculado e do resumo recebido não serão iguais.

Funções de resumo criptográfico são utilizadas em diversos protocolos de troca de mensagens para garantir a integridade delas como no *HTTPS* por exemplo, uma prática comum na internet é a disponibilização do resumo de um arquivo de *download* para a confirmação que não houve nenhuma alteração em relação ao arquivo publicado e o que foi efetivamente baixado por um usuário (ORAM, 2022).

2.3 INFRAESTRUTURA DE CHAVES PÚBLICAS

Proposta em sua primeira iteração por (DIFFIE; HELLMAN, 1976) com a ideia de uma estrutura hierárquica para a criação, armazenamento e distribuição de chaves públicas com o intuito de viabilizar a criptografia assimétrica em um contexto de redes, em sua proposta um arquivo público guardaria chaves públicas e as distribuiria. (KOHNFELDER, 1978) então tentou implementar em sua época uma Infraestrutura de Chaves Públicas (ICP) e criou o conceito de um certificado, para contornar a necessidade mútua de comunicação com o objetivo de buscar as chaves no arquivo público, este certificado é emitido pelo arquivo público e assinado com sua chave privada e associa um par de chaves a um membro da comunicação, logo ambas as partes podem confirmar a veracidade do certificado usando a chave pública do arquivo público e confirmar que o certificado e chaves são de um membro da comunicação.

Mesmo com sua modernização das ICPs, alguns problemas devido à forma que a solução foi pensada ainda permanecem, por ser muito generalista, e sofrer muito com problemas relativos a caminhos infelizes no fluxo. Um destes problemas é a revogação de certificados, ela se baseia em uma lista de certificados que foram revogados e é disponibilizada pelo emissor, entretanto, não existe padrão de frequência de atualização da lista no emissor e nem no verificador, assim como a falta de conhecimento dos usuários em caso de extravio do par de chaves e certificado (GUTMANN, 2002).

2.3.1 Assinaturas Digitais

Uma assinatura digital tenta emular as mesmas características de uma assinatura feita a mão em papel. Documentos em papel e digitais têm que ser idênticos nos sentidos de validade

das assinaturas presentes. Os documentos em papel são certificados utilizando a forma física da assinatura. Já os digitais utilizam o retorno dos algoritmos de assinatura digital para cumprir o mesmo propósito de validação e autenticação dos documentos digitais (SUBRAMANYA; YI, 2006). Para garantir validação e autenticidade para documentos digitais, os algoritmos de assinatura digital utilizam diversos conceitos criptográficos como criptografia assimétrica, funções de Resumo Criptográfico e uma infraestrutura de chaves públicas.

O processo de assinatura de um documento digital pode ser resumido na Figura 1. Onde basicamente para uma mensagem M é utilizada uma função de resumo criptográfico para obter $resumo = H(m)$. É inserido então o resumo da mensagem numa função de cifragem assimétrica onde é utilizada a chave privada do assinador para obter a assinatura digital com $assinatura = C(resumo, chave\_privada)$. É então anexada a assinatura em conjunto a mensagem. Desta forma qualquer pessoa poderia verificar a integridade e autenticidade da assinatura decifrando com a chave pública. O processo de verificação de uma assinatura é descrito pela Figura 2. Onde se obtém o resumo cifrado $resumo\_obtido = D(assinatura, chave\_publica)$. Calcula-se então o resumo da mensagem recebida e compara-se com o resumo obtido para garantir que não houve nenhuma alteração na mensagem após a assinatura e confirma-se que o assinante é o detentor daquele par de chaves.

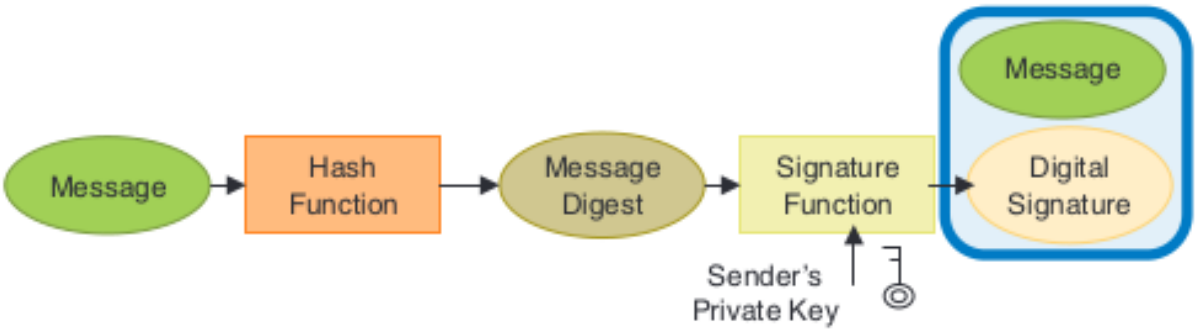

Figura 1 – Assinatura de uma Mensagem. Retirado de: (SUBRAMANYA; YI, 2006)

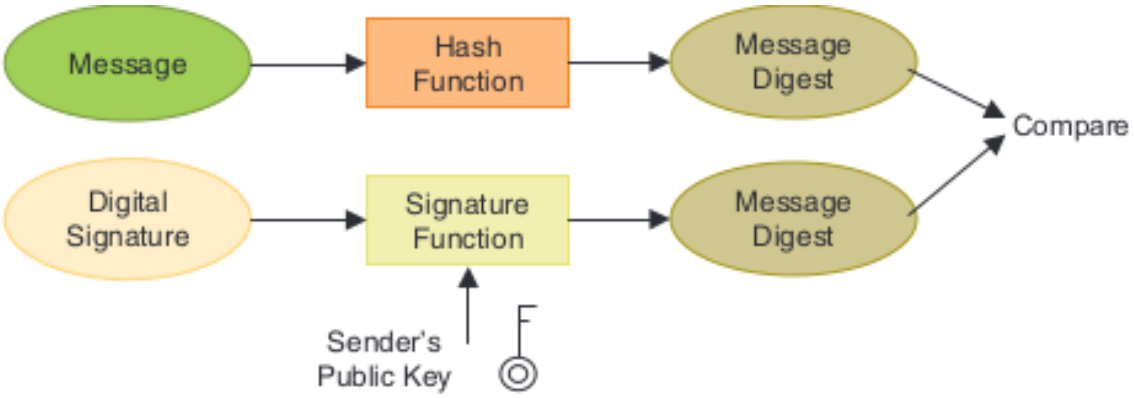

Figura 2 – Verificação de uma assinatura. Retirado de: (SUBRAMANYA; YI, 2006)

2.3.2 Certificado Digital

O conceito de certificado digital é introduzido por KOHNFELDER. Ele serve como uma atribuição de um par de chaves a uma entidade, uma pessoa vai a uma autoridade certificadora (AC) então a AC após confirmar a identidade de quem requisitou emite um certificado para o requisitante. Esse certificado é um arquivo digital definido em (MYERS et al., 1999). Ele é um arquivo X.509 e funciona similarmente a um dicionário com chaves, que contém informações sobre o arquivo e seu dono, como nome, e-mail, data de validade, endereço, entre outras possíveis extensões. Ele também contém a chave pública de seu dono e uma assinatura digital para comprovar a posse do par de chaves e caminho de certificação. Pode se então verificar tanto a assinatura do certificado digital pela AC final e o caminho de certificação para emissão do certificado podendo então confiar que aquele certificado foi emitido por uma instituição confiável.

2.3.2.1 Certificate Signing Request

Para obter um certificado é necessário criar um *Certificate Signing Request* (CSR) e o enviar para uma AC para emissão do certificado. Após a criação do par de chaves pode-se obter um CSR, ele contém a chave pública e é assinado usando a chave privada do requisitante do certificado, assim como informações a serem usadas para a emissão do certificado (NYSTORM; KALISKI, 2000).

2.3.2.2 PKCS7

O formato PKCS7 representa um conteúdo que contém criptografia aplicada a ele, como, por exemplo, um conteúdo que contém uma assinatura digital. Este formato de mensagem criptografada, pode ser usado para a representação de certificados digitais, e também para qualquer conteúdo assinado, inclusive recursivamente (KALISKI, 1998).

2.3.2.3 PKCS12

O PKCS12 é um arquivo digital que representa uma identidade digital, e serve para a transferência da mesma, contendo os certificados e pares de chaves de uma identidade digital. Permitindo então a importação e exportação de identidades facilmente. Este Arquivo suporta uma senha para proteção dos conteúdos internos que são dados sensíveis, como, por exemplo uma chave privada (MORIARTY et al., 2014).

2.3.3 Revogação de Certificados

Um dos problemas das ICPs atuais é a revogação de certificados inválidos. A estratégia utilizada no padrão atual da indústria é uma lista de certificados revogados geralmente mantida pela AC Final. Em caso de perda ou extravio o dono daquele certificado pede a AC Final a inclusão de seu certificado em uma lista de certificados revogados, a inclusão em si, pode demorar logo que geralmente estas listas são atualizadas em um intervalo fixo de tempo, após a inclusão um programa de verificação de assinaturas pode também atualizar suas listas em um intervalo fixo de tempo, porém, diferente do intervalo da AC Final o que acarretará o programa reconhecer como válidas assinaturas posteriores a revogação do certificado.

O fato destas listas de revogação se basearem em soluções antigas para cartões de crédito traz algumas problemáticas como a baixa frequência de emissão, alto custo de transmissão e verificação assim como a suscetibilidade a ataques de negação a serviço (GUTMANN, 2002). Outro problema que ocorre devido aos altos custos computacionais de verificar se um certificado está ou não na lista e de atualizar a lista periodicamente, o que faz com que muitos programas nem atualizem suas listas de Revogação, focando em desempenho ao invés de segurança.

2.3.4 Certificados de uso único

Os certificados digitais de uso único buscam resolver diversas problemáticas com as atuais infraestruturas de chaves públicas principalmente quando o assunto são assinaturas digitais com foco em pessoas físicas. Eles buscam remover a necessidade de guardar seu certificado em um dispositivo próprio para armazenamento, diminuindo não só o custo, mas também a barreira de conhecimento necessária para a manutenção correta de um certificado digital. Elimina-se a necessidade de manter um certificado e par de chaves que é reutilizado para várias assinaturas, e para cada documento assinado é criado um novo par de chaves utilizando também um novo certificado único emitido para aquele documento.

O certificado possui em suas extensões além da identificação de seu proprietário com informações como nome e e-mail também o resumo criptográfico do conteúdo a ser assinado. O resumo criptográfico é feito uma vez quando o documento está pronto para ser assinado, e enviado em conjunto com o CSR para a AC que emitirá o certificado. Este resumo não pode ser alterado uma vez no certificado digital, logo pode-se confirmar que este certificado foi emitido apenas para assinar aquele conteúdo. A cada assinatura então é criado um novo par de chaves e emitido um novo certificado e ao fim do processo de assinatura tanto o certificado quanto o par de chaves pode ser descartado, logo que o mesmo não pode ser reutilizado (MAYR et al., 2023).

Ao realizar uma assinatura o Certificado contendo a chave pública e incluído dentro do conteúdo de mensagem assinada, sendo possível então do ponto de vista do assinante descartar o par de chaves e certificados utilizados na assinatura, ou mesmo torna-lá pública em. Do ponto de vista de quem está verificando a assinatura, poderá encontrar dentro do objeto de assinatura

o certificado e o conteúdo da assinatura digital podendo então utilizar a chave pública dentro do certificado para realizar o processo de validação.

Caso um atacante consiga ter acesso a um certificado e um par de chaves de um certificado de uso único, se o mesmo tentar assinar um outro documento o resumo criptográfico do novo conteúdo assinado será diferente do contido dentro certificado, sendo possível então na verificação da assinatura constatar que o certificado não foi emitido para assinar aquele documento, tornando assim a assinatura inválida, entretanto, o validador tem que estar preparado para realizar a validação do resumo criptográfico, do contrário o mesmo dirá que assinatura é válida. Os verificadores atuais situados fora do contexto de certificados de uso único não realizam esta verificação. Esta abordagem também remove a necessidade de manutenção e uso de uma lista de certificados revogados, logo que se um certificado for compromissado caso outro documento seja assinado haverá disparidade do resumo assinado e do presente no certificado. A ausência de uma lista de revogação de certificados diminui também a complexidade do processo de verificação da assinatura, logo que não é necessário procurar a Lista de Certificados Revogados da AC emissora (MAYR et al., 2023).

2.4 APLICAÇÃO FRONT END

Um *Front End* (FE) se refere a parte de um sistema e que o usuário final interage diretamente (OXFORD. . . , 2023). Em se tratando de aplicações *WEB* o FE se trata do código que é executado diretamente no *browser* utilizado para acessar uma página.

2.4.1 React

O React é uma biblioteca de código aberto para *JavaScript*, que permite construir aplicações FE utilizando componentes, facilitando o controle de estado e atualização dinâmica das aplicações *WEB* (REACT, 2023). O código é escrito na linguagem JSX que combina elementos de *HTML* com *JavaScript* sendo possível criar aplicações inteiras apenas utilizando isto. A biblioteca *React* foi usada para a criação da aplicação devido a sua consolidação como principal biblioteca no mercado de trabalho com 49% do mercado a utilizando (JAVASCRIPT, 2023).

2.4.2 NextJS

O *NextJS* é um *framework* de desenvolvimento de aplicações WEB criado pela *Vercel*. Ele utiliza o *React* para criação da interface e fornece a possibilidade da criação de um *API* assim como integrações na autenticação com bibliotecas e renderização no lado do servidor (NEXTJS, 2023).

2.5 TRABALHOS EXISTENTES

Existem diversas soluções para a assinatura de documentos digitais tanto na iniciativa pública quanto na privada, visto que este é um serviço que proporciona a atualização e a desburocratização da assinatura de documentos. Uma assinatura digital é tão segura e válida quanto uma firma realizada em cartório, e hoje no Brasil e em diversos países ambas possuem mesma validade legal, para contratos e documentos (BRASIL, 2020).

2.5.1 Assinatura Digital em iniciativas públicas

A ICP-Brasil é a infraestrutura de chaves públicas criada em 2001 que regulamenta como é feita a emissão de certificados no Brasil, para serviços do estado e para seus cidadãos (ICP-BRASIL, 2023). Hoje é possível assinar documentos utilizando a conta gov.br. O processo de assinatura acontece utilizando um certificado emitido para um usuário que fica guardado em um servidor. Este certificado é reutilizado diversas vezes para a assinatura de diversos documentos até que ele expire, ao expirar é criado um novo par de chaves e um novo certificado é armazenado no servidor. Num possível vazamento de dados do site seria possível um atacante obter acesso a estas chaves e certificados e assinar documentos se passando pela pessoa dos quais o par de chaves e o certificado foram roubados (ASSINATURA. . . , 2023).

Outra iniciativa em assinatura digital é o assina UFSC que permite a assinatura de documentos digitais com um certificado da ICP-EDU como também com o certificado do gov.br. A ICP-EDU é um braço da ICP-Brasil com foco para emissão de certificados para membros da comunidade acadêmica. A estratégia de assinatura é a mesma que a do gov.br, que utiliza o mesmo certificado para diversas assinaturas, afinal mesmo com diversos problemas este ainda é o padrão utilizado pela indústria (E-UFSC, 2023).

2.5.2 Assinatura digital em iniciativas privadas

Muitos programas de assinatura digital rodam em servidores e são acessíveis via *browser* como, por exemplo o Docusing, estes programas seguem o padrão da indústria de ter um certificado de média duração e reutilizar o mesmo, mantendo este certificado e os pares de chave associados a ele em um servidor na nuvem, retirando completamente a responsabilidade do usuário por eles mas também sendo suscetível a ataques e vazamentos (DOCUSIGN, 2023).

Existem também programas de assinatura que rodam na máquina local como o *Adobe Acrobat* que também geralmente permitem a edição e montagem de arquivos PDF, muitas vezes estes pedem um certificado válido instalado na máquina ou disponível via leitor de certificados para assinar um documento com ele. Neste caso toda a responsabilidade pela manutenção do certificado é delegada ao usuário que mantém sua versão física em um *smart card* ou em um arquivo em sua máquina local. Isto acarreta uma necessidade do usuário em saber os procedimentos padrões a serem feitos em caso de extravio ou perda do arquivo, ou do dispositivo

que contém o certificado digital, como, por exemplo, comunicar o emissor para a revogação do certificado (ACROBAT, 2023).

2.5.3 Assinador Avançado

O Assinador Avançado é um projeto do LabSEC que visa a criação de uma ICP com certificados de uso único, conseguindo prover a assinatura de documentos digitais de maneira rápida e eficiente para a população, sem a necessidade da manutenção de um certificado, ou de guardar ele em um servidor. Como para cada documento assinado é emitido um novo certificado, elimina-se a responsabilidade do usuário e também como as chaves e certificados são apenas criados, usados e descartados ele acaba sendo uma alternativa viável e segura para a assinatura de documentos digitais.

O projeto hoje suporta a assinatura e verificação de documentos digitais, diferentemente do trabalho proposto o processo de assinatura ocorre no *backend* da aplicação, o que implica no *upload* do documento ser necessário para a assinatura em conjunto com a criação de chaves ocorrer no lado do servidor. O projeto é dividido em diversos microsserviços, dos quais os de ICP e de verificação de assinaturas foram utilizados sem nenhuma modificação como apoio para o desenvolvimento deste trabalho, o microsserviço de ICP tem a função de receber um CSR e emitir um respectivo certificado.

3 PORTABLE DOCUMENT FORMAT

O *Portable Document Format (PDF)* é um tipo de arquivo que representa, documentos e formulários no ambiente digital. Estes documentos possuem a característica de serem idênticos indiferentemente da aplicação utilizada para os ver ou processar. Não importa também o sistema operacional em que a aplicação utilizada está sendo executada. Os documentos PDF utilizam uma estrutura estrita e contém diversas informações sobre como o conteúdo deve ser apresentado ao usuário final (SELHAUSEN, 2018).

Por esta interoperabilidade o PDF acaba sendo o formato mais utilizado quando se trata de documentos digitais. O formato é definido pela ISO 32000, com um arquivo PDF sendo organizado em uma hierarquia de árvore com diversos dicionários, matrizes e *streams* para representação de conteúdos. Esta hierarquia de árvore começa no objeto *Catalog*, este possui referências para diversos outros tipos de objetos, como páginas e metadados sobre o documento. As páginas de um documento são representadas pelos objetos *Page*. Um documento PDF também pode ter diversos tipos de mídia dentro dele como textos, imagens, formulários que serão referenciados pelas suas respectivas páginas.

3.1 ESTRUTURA

Um documento PDF pode ser divido em algumas partes, como o Cabeçalho, o Corpo a tabela de Referências e o *Trailer*. Após os objetos de páginas e conteúdos existe um objeto de *Trailer* que anuncia o fim do documento assim como uma tabela de referências que guarda para cada objeto o *byte* aonde seu conteúdo começa. A Figura 3 ilustra a macro-estrutura de um arquivo PDF.

Figura 3 – Estrutura de um arquivo PDF. Adaptado de: (SELHAUSEN, 2018)

O Cabeçalho contém metadados e a raiz do documento, objetos como os de *Catalog* e *Page* se fazem presentes ali apontando para os conteúdos presentes no corpo do documento. Dentro do corpo se pode possuir imagens, textos, formulários. Todo conteúdo do PDF apre-

sentado ao usuário fica ali. A tabela de referências possui os bytes onde os objetos do PDF começam. E o trailer aponta o fim do documento seguido de uma sequência de caracteres que indica o fim do arquivo. Cada objeto de um arquivo PDF é um dicionário com atributos que caracterizam o seu tipo de objeto e conteúdo.

3.1.1 Objeto *Catalog*

O objeto *Catalog* é a raiz do documento como pode ser visto em Figura 4, ele é um dicionário com referências para metadados e páginas do documento. Ele pode conter uma entrada de *AcroForm* utilizada para referenciar campos de formulários e assinaturas (DOCUMENT. . . , 2020).

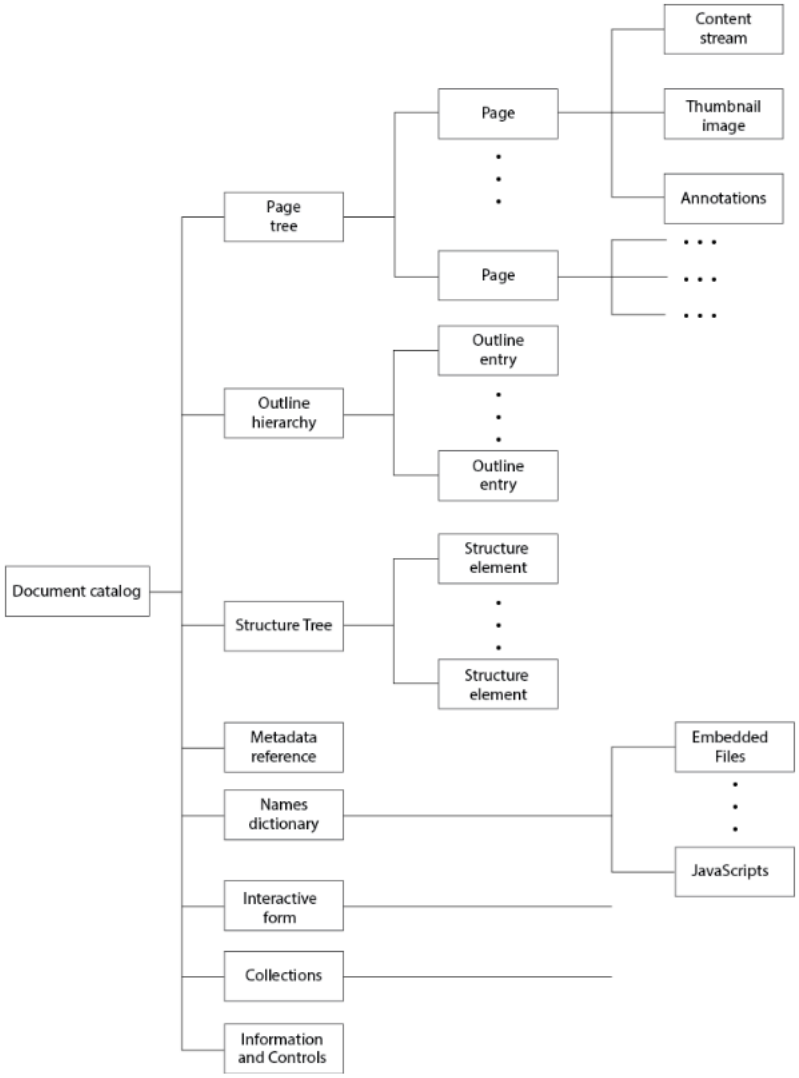

Figura 4 – Estrutura detalhada de um arquivo PDF. Retirado de: (DOCUMENT. . . , 2020)

```
7 0 obj
<<
/Type /Catalog
```

```
/Pages 6 0 R
/AcroForm <<
/Fields [12 0 R]
/SigFlags 3
>>
>>
endobj
```

Listing 3.1 – A estrutura padrão de um objeto *Catalog*

Pode-se observar que o objeto *Catalog* começa com um ID do objeto no documento e um número de revisão seguido da *keyword* obj. Após isto existe um dicionário de chaves e valores. $/Type$ indica que o objeto é do tipo *Catalog*, $/Pages$ aponta para o objeto *Page* presente no documento, $/Acroform$ contém um dicionário com as entradas $/Fields$ um *array* de campos do dicionário e $/SigFlags$ com uma *flag* sobre assinaturas desse formulário. Após estes conteúdos o dicionário é fechado e encerrado o objeto com o *keyword* $endobj$. Uma estrutura de exemplo de um objeto *Catalog* está disponível em *Listing* 3.1.

3.1.2 Objeto *Page*

Existem 2 objetos para representar páginas o de $/Page$ e o de $/Pages$ o de $/Pages$ contém um atributo $/Kids$ que contém um *array* com referência aos objetos de $/Page$ e um atributo $/Count$ com o número de entradas de $/Kids$. Já o de $/Page$ representa literalmente as páginas do PDF e possui diversos atributos relacionados a exibição e conteúdos assim como a fonte utilizada entre outras opções disponíveis em (DOCUMENT..., 2020). Na extensão deste trabalho será utilizado apenas o atributo $/Annots$ que adiciona uma anotação a esta página do PDF. Um exemplo dos objetos de *Page* e de *Pages* está disponível em *Listing* 3.2.

```
2 0 obj
<</Type /Page
/Resources <</ProcSet [/PDF /Text /ImageB /ImageC /ImageI]
/ExtGState <</G3 3 0 R>>
/Font <</F4 4 0 R>>>>
/MediaBox [0 0 612 792]
/Contents 5 0 R
/StructParents 0
/Parent 6 0 R>>
endobj
6 0 obj
<</Type /Pages
/Count 1
/Kids [2 0 R]>>
endobj
```

Listing 3.2 – A estrutura padrão de um objeto *Page* e de um objeto *Pages*

3.1.3 Objeto *Widget*

Uma anotação é um objeto que associa outro objeto como uma assinatura, um link ou algum tipo de mídia a uma localização em uma página de um PDF. Existem diversos tipos de anotação, entretanto o utilizado neste trabalho é a anotação de subtipo *Widget*. O *Widget* é utilizado principalmente para formulários e para referenciar objetos como assinaturas Digitais (DOCUMENT..., 2020).

3.1.4 Tabela de referências

A tabela de referências de um PDF é um objeto localizado no fim do arquivo, ele indica onde está localizado o início dos bytes de cada objeto dentro do arquivo. Um exemplo de Tabela de Referências está disponível em *Listing* 3.3.

```
1 xref
2 0 6
3 0000000003 65535 f
4 0000000017 00000 n
5 0000000081 00000 n
6 0000000000 00007 f
7 0000000331 00000 n
8 0000000409 00000 n
```

Listing 3.3 – A estrutura padrão de uma tabela de referências

Ela começa com o *keyword* $xref$ seguida após uma quebra de linha do primeiro elemento a ser referenciado, embora neste exemplo o primeiro número seja zero isto não é um requisito, logo após é indicado a quantidade de elementos que a tabela irá referenciar após o primeiro elemento. Nas seguintes linhas se tem o seguinte formato, um número de 10 dígitos que indica onde está localizado o primeiro byte do objeto dentro do documento, um número de 5 dígitos de geração que indica quantas vezes este elemento foi deletado e recriado e um caractere f ou n para indicar se aquele objeto livre (f) ou em uso (n).

3.1.5 Objeto Trailer

Um objeto de tipo *Trailer* simboliza o fim do documento PDF. Ele se localiza após a tabela de referências. Sua estrutura pode ser definida começando com uma linha com a *keyword* $trailer$. Após isto é definido um dicionário de chaves e valores, com diversas entradas possíveis como quantos objetos foram definidos na tabela de referências com a chave $/Size$, qual o *ID* do objeto de *Catalog* no atributo $/Root$, entre outras possíveis chaves. Após a definição do dicionário existe a *keyword* $startxref$ e após esta palavra o endereço de byte em que a tabela de referências começa. Por último é adicionado um $\%\%EOF$ para terminar o arquivo (DOCUMENT..., 2020). Um exemplo da estrutura de um objeto *Trailer* está disponível em *Listing* 3.4.

```
trailer
<<
/Size 22
/Root 2 0 R
/Info 1 0 R
>>
startxref
18799
%%EOF
```

Listing 3.4 – A estrutura padrão de um objeto *Trailer*

3.2 ATUALIZAÇÃO INCREMENTAL

Existem dois tipos de atualizações que podem ser feitas em um arquivo PDF já existente. Ao adicionar um objeto pode-se incluí-lo dentro do documento antes da tabela de referência e atualizá-la, adicionando assim o conteúdo atualizado, entretanto caso haja uma assinatura no documento isto a tornará inválida, logo que houve alterações no conteúdo assinado no documento após assinatura, quebrando a verificação usando uma função resumo criptográfico do mesmo. Atualizações incrementais possibilitam aplicações adicionarem conteúdo novo ou atualizarem algo já existente no fim de um arquivo PDF. Adicionalmente, isto permite modificar os documentos PDF que contém assinaturas digitais sem invalidar a assinatura, pois o conteúdo originalmente assinado não é modificado (SELHAUSEN, 2018).

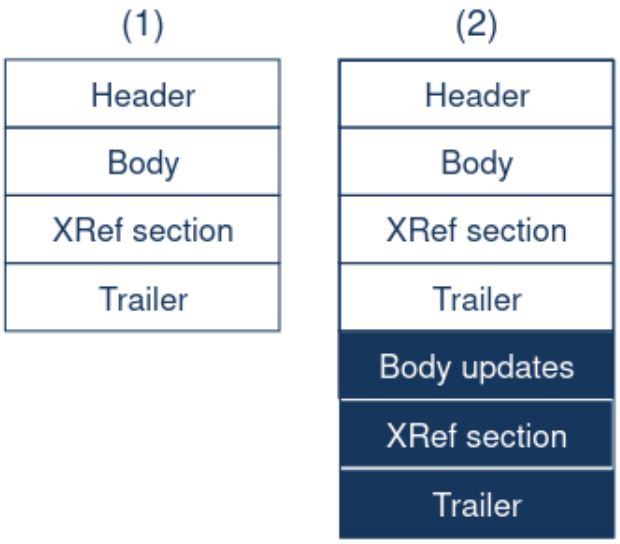

Figura 5 – Estrutura de um arquivo PDF com atualização incremental. Retirado de: (SELHAUSEN, 2018)

Ao atualizar um arquivo incrementalmente é adicionado a atualização dos conteúdos do corpo do PDF no fim do mesmo podendo até redefinir objetos previamente definidos, então é adicionada uma nova tabela de referência para quaisquer elementos atualizados ou adicionados

e finaliza-se com um novo objeto trailer que aponta para o último Trailer disponível. O resultado deste processo é mostrado em Figura 5.

3.3 ASSINATURAS DIGITAIS EM PDF

Ao realizar uma assinatura em um documento PDF é necessário utilizar alguns elementos previamente explicados como os objetos de *Catalog*, *Page*, *Widget*. Assim como o processo de atualização incremental e o objeto de *Sig*.

3.3.1 Objeto Sig

Um objeto *Sig* contém os conteúdos de uma assinatura digital em um documento PDF. O objeto *Sig* é representado com um dicionário com diversas entradas. A chave $/Contents$ possuí o conteúdo da assinatura. A chave $/ByteRange$ é um *array* com 4 entradas representando as seções do documento que foram assinadas, com o primeiro número sendo o *byte* do começo do conteúdo assinado (Geralmente 0), o segundo número sendo o *byte* que somado ao primeiro chega ao começo do valor da chave de $/Contents$ do objeto $/Sig$. O terceiro número sendo o *byte* partir do fim da assinatura, e o quarto um número que somado ao terceiro resulta no último *byte* assinado. Desta forma o objeto de *Sig* geralmente engloba todo o conteúdo do arquivo menos o conteúdo da assinatura do mesmo (DOCUMENT..., 2020).

A chave $/Type$ tem seu valor como *Sig* para representar que o objeto é uma assinatura. A chave $/Filter$ indica qual o validador da assinatura, alguns valores possíveis são Adobe.PPKLite, Entrust.PPKEF, CICI.SignIt, e VeriSign.PPKVS (DOCUMENT..., 2020). A chave de $/SubFilter$ indica a codificação do conteúdo da assinatura, o $/SubFilter$ utilizado neste trabalho foi o adbe.pkcs7.detached. Já a chave *M* representa a data da assinatura. Existem outras chaves utilizadas para representar diferentes atributos de uma assinatura disponíveis em (DOCUMENT..., 2020). A chave $/Contents$ para o $Subfilter$ selecionado neste trabalho irá conter uma mensagem criptografada *PKCS#7* contendo não só a assinatura mas também o certificado utilizado na assinatura e o caminho de certificação, com este conteúdo binário sendo codificado em DER. Pode-se então utilizar o certificado contido dentro da chave $/Contents$ para a verificação da assinatura. A estrutura de um objeto *Sig* pode ser vista em *Listing* 3.5.

```
13 0 obj
<<
/Type /Sig
/Filter /Adobe.PPKLite
/SubFilter /adbe.pkcs7.detached
/M (D:20230410141326-03'00')
/Contents <Conteudo>
/ByteRange [0 23675 42621 15919]
>>
```

```
10 endobj
```

Listing 3.5 – A estrutura padrão de um objeto *Sig*

3.3.2 Processo de assinatura

No ponto de vista técnico, o processo de assinatura pode ser descrito pelo seguinte algoritmo:

1. Lê o conteúdo do arquivo PDF.

2. Adiciona um objeto *Sig* após o *Trailer* do arquivo, o valor da chave *Contents* deste objeto *Sig* é populado com um valor *placeholder*. Este valor pode ser a letra A repetida "n"vezes. A chave *ByteRange* é populada levando este valor de *placeholder* em consideração.

3. Adiciona um objeto do tipo *Widget* que referencia o objeto do Tipo *Sig* definido.

4. Redefine os objetos de *Catalog* e de *Page* adicionando uma referencia ao objeto *Widget* adicionado.

5. Adiciona uma tabela de referencias no final do arquivo assim como um objeto *Trailer* que aponta para o *Trailer* anteriormente definido.

6. Atualiza os valores de *ByteRange* do objeto de *Sig* levando em conta a redefinição dos outros objetos.

7. Calcula o resumo criptográfico referente ao *ByteRange* definido.

8. Assina o resumo criptográfico utilizando a chave privada do assinante. E a codifica em relação ao valor definido na chave *Subfilter* do objeto *Sig*.

9. Substitui o valor *placeholder* na chave *Contents* do objeto de *Sig* pelo valor codificado. Se o *placeholder* for maior que o tamanho do conteúdo da assinatura é adicionado o carácter 0 *n* vezes para que ambos tenham o mesmo tamanho e não haja mudança no *ByteRange*.

O processo de assinatura pode incluir processos mais específicos de acordo da complexidade de sua aplicação e uso (SELHAUSEN, 2018). Em uma infraestrutura de certificados de uso único entre as etapas 7 e 8 do processo haveria a criação de um par de chaves e um *Certificate Signing Request* (CSR) incluindo o resumo criptográfico do documento para uma ICP, que por sua vez, responderia com o certificado, este então seria utilizado para o processo de assinatura.

3.3.3 Processo de verificação

O processo de verificação de uma assinatura digital em um documento PDF é similar ao de verificação de uma assinatura de uma mensagem assinada. Com a diferença de que a assinatura encontra-se no meio do documento assinado. Deve-se então calcular o resumo criptográfico dos conteúdos explicitados pelo *ByteRange* da assinatura e comparar com o valor assinado dentro da chave *Contents* do objeto *Sig*. Após confirmar a integridade da assinatura se deve analisar a cadeia de certificação do certificado assinado, garantindo que exista um emissor confiável no caminho de certificação e garantindo que o mesmo não foi revogado (SELHAUSEN, 2018). Em uma infraestrutura de certificados de uso único deve-se garantir que o resumo criptográfico presente dentro do certificado é o mesmo resumo assinado, pois o certificado foi emitido apenas para assinatura daquele documento.

4 DESENVOLVIMENTO

Nesta seção do trabalho será apresentada a proposta, seu processo de construção, ferramentas utilizadas e experimentos realizados.

4.1 PROPOSTA

O objetivo deste trabalho se trata da criação de uma aplicação *Front End* que realize a assinatura de documentos digitais. O processo de assinatura deve acontecer apenas no *browser* onde a aplicação está rodando. Será utilizada uma infraestrutura de certificados de uso único para assinatura dos documentos, contendo o resumo do conteúdo assinado em uma das extensões do certificado gerado para assinatura daquele documento. O documento digital a ser assinado também não sairá do local de execução, assim como o par de chaves gerado para o certificado, com apenas a chave pública trafegando em rede contida em um CSR. Sem a necessidade de fazer o *upload* do documento digital a ser assinado para um servidor, conserva-se a privacidade do documento assinado.

Ao verificar uma assinatura gerada pela aplicação será possível validar o resumo criptográfico presente no certificado com o conteúdo assinado, garantindo então que o certificado foi emitido para aquele documento e não foi reutilizado retirando a necessidade de manutenção por longo tempo do mesmo pelo usuário ou pelo servidor. A ideia da aplicação é ser o mais simples e sucinta o possível, com o usuário se autenticando, fornecendo seu documento digital a ser assinado, clicando no botão assinar e recebendo o mesmo documento assinado de volta.

Esta aplicação rodará como um micro serviço de uma ICP sendo seu braço de assinatura, e toda vez que um documento for assinado ela irá criar um CSR e fará sua única requisição externa mandando apenas para o microsserviço de PKI o CSR contendo resumo criptográfico do documento e a chave pública dos pares de chave gerados. O PKI então retornará o certificado requisitado em conjunto com o caminho de certificação que o emitiu. A aplicação então realiza a assinatura no local de execução usando a resposta do microsserviço de PKI. Nenhuma das ações feitas pelo PKI foi modificada neste trabalho em comparação com o assinador avançado disponibilizado pelo LabSec. Será utilizado um provedor de autenticação do LabSec que já funciona em conjunto aos microsserviços existentes, para realizar a autenticação e controle de identidade no protótipo, o provedor é a aplicação *KeyCloak*.

O fluxo de assinatura da aplicação pode ser traduzido em um algoritmo que roda quando o botão de assinar é clicado. De maneira geral o procedimento de assinatura de um documento digital usando certificados de uso único é descrito pelo Pseudo-Algoritmo disponível em *Listing* 4.1.

```
1 Funcao Assinar (BytesPDF, nome){
2     BytesPDF <- Adiciona_Objetos_PDF_da_Assinatura(BytesPDF)
3     Resumo <- Calcula_Resumo_Dos_bytes_a_serem_Assinados(BytesPDF)
4     Par_chaves <- Gera_Par_Chaves()
```

```
5       CSR <- Cria_CSR(Par_Chaves, Resumo, nome)
6       PKCS7 <- Envia_CSR_A_ICP(CSR)
7       PKCS12 <- CriaPKCS12(Par_chaves.privada, PKCS7)
8       Bytes_assinados <- AssinaPDF(BytesPDF, PKCS12)
9       RETORNA Bytes_assinados
10  }
```

Listing 4.1 – Pseudo-Algoritmo de assinatura de documento digital PDF

A função *Adiciona_Objetos_PDF_da_Assinatura*, irá fazer uma atualização incremental no documento PDF recebido. Adicionando dois objetos, um *Sig* que representa a assinatura que neste momento possui um valor de *placeholder* em sua chave */Contents*. E um objeto de *Widget* para referenciar o objeto *Sig* adicionado. Está função também redefine os objetos de *Catalog* e *Page* para referenciar o *Widget* adicionado como assinatura. Após isto é definida uma Tabela de referências para conteúdos adicionados e atualizados e um objeto de *Trailer*. Então se Atualiza o *ByteRange* do objeto *Sig* de forma com que ele envolva todo o documento, menos o conteúdo da assinatura que agora possui um *placeholder*.

Após retornar os Bytes do PDF da função anterior é calculado o resumo criptográfico do conteúdo explicitado no *ByteRange* da assinatura. Obtendo então o resumo com o valor a ser assinado, gerando então um par de chaves. A aplicação cria um CSR utilizando o nome recebido do provedor de autenticação, incluindo a chave pública e o resumo criptográfico em uma extensão do CSR. Logo após é realizada a assinatura do CSR usando a chave privada. Então é estabelecida uma conexão HTTPS com o servidor da aplicação FE com o conteúdo do CSR, está por sua vez se conecta via HTTPS ao microsserviço de PKI uma vez que o mesmo se encontra em uma rede privada e não pode ser acessado diretamente pelo *Front End*.

O microsserviço de PKI então cria um certificado e inclui sua cadeia de certificação nele, respondendo ao Servidor da aplicação *Front End*, que por sua vez responde à aplicação *Front End* com o conteúdo do certificado. A resposta do microsserviço por sua vez é um conteúdo PKCS7, que deve ser convertido para um conteúdo PKCS12, utilizando a chave privada, para realizar o processo de assinatura. O processo de assinatura é então realizado, utilizando o certificado fornecido assim como os bytes do PDF. É calculado o resumo criptográfico e utilizada as chaves contidas dentro do PKCS12 para gerar o conteúdo assinado. Uma vez com o conteúdo assinado é então substituída a chave *Contents* do objeto *Sig* adicionado no primeiro passo, pelo conteúdo real da assinatura. Retornando então os bytes do PDF para o usuário.

Pode-se implementar um assinador que execute assinatura com certificados de uso único com a assinatura ocorrendo diretamente no local de execução, ou em um servidor seguindo este pseudo algoritmo. Logo o escopo delimitado para este trabalho é a construção de um protótipo que executasse este algoritmo diretamente no *browser* realizando assim a assinatura de um documento digital, e que fosse possível determinar sua validade.[1]

[1] O código do protótipo funcional está disponível publicamente em: https://github.com/cavrau/assinador-tcc

4.2 FERRAMENTAS

O *JavaScript* é a linguagem de programação padrão para o desenvolvimento de sistemas de *Front End* que rodam no *Browser* e Sites em geral. Entretanto, outra opção é usar o TypeScript, que adiciona tipagem estática ao *JavaScript* melhorando a legibilidade e manutenção do código, logo foi utilizado o *TypeScript* para elaboração do código do projeto. A biblioteca *React* também foi utilizada, ela é focada na construção de interfaces do usuário reativas a mudanças de dados, fazendo isto de forma fácil e intuitiva. Quando comparada com outra ferramenta como o *Framework* Angular ela é em um primeiro momento menos robusta em funcionalidades, devido ao escopo do projeto estas funcionalidades não foram necessárias, mas mesmo que fossem, existem diversas bibliotecas mantidas por terceiros que cobrem as funcionalidades extras não presentes na biblioteca padrão do *React*. Outra opção seria o *Vue.js* que possui muita similaridade com o *React*, mas não possui uma comunidade tão grande.

Foi utilizado o *NextJs* para a elaboração do protótipo, ele engloba o *React* servindo o mesmo e dando a possibilidade da criação de rotas de *API* no mesmo projeto. Foi utilizada a biblioteca *next-auth* para integração com o provedor de autenticação. Para o desenvolvimento do algoritmo de assinatura na parte criptográfica foram utilizadas bibliotecas como o *pkijs* para a geração do CSR, *asn1js* pois ela é uma dependência da *pkijs* e ajuda com diversas funções de conversão de valores. A biblioteca *Node-Forge* para lidar com conversões como entre a resposta da ICP de um padrão PKCS7 para o padrão PKCS12 e assinatura do PDF. Também foi utilizada a própria biblioteca *Crypto* nativa do *JavaScript*, disponível em *browsers* e responsável pela geração do par de chaves e execução das funções de resumo criptográfico. Para construção do protótipo foram selecionados como algoritmos de criptografia assimétrica e Resumo criptográfico o RSASSA-PKCS1-V1.5 e o SHA-384.

Para a realização da requisição a ICP com o envio de arquivo contendo o CSR foi utilizada a biblioteca *axios*. O código desenvolvido para a adição de assinatura *placeholder* e assinatura foi baseado na biblioteca *node-signpdf*, mas devido às peculiaridades como a necessidade de obter o resumo criptográfico dos conteúdos do documento após a adição da assinatura *placeholder* e o fato da biblioteca ter sido feita para execução em servidor não em *browser* foram realizadas alterações nas funções descritas anteriormente para fácil integração.

4.3 DESENVOLVIMENTO DO PROTÓTIPO

Em um primeiro momento foi elaborado a página WEB vista pelo usuário, com possibilidade de ao clicar na área de assinatura ou arrastando um documento para ela fosse possível ver o mesmo. Após terminar a estrutura visual da página começou então o processo de criação da assinatura. Não foram encontradas bibliotecas que realizem a manipulação de um arquivo PDF utilizando a atualização incremental. Logo se a manipulação do PDF fosse feita utilizando uma biblioteca adicionar uma assinatura a um documento que já contém uma assinatura invalidaria a mesma, pois o conteúdo previamente assinado seria modificado com a adição de mais

elementos.

Devido à ausência de bibliotecas foi necessário realizar as atualizações de conteúdos de PDF utilizando código próprio, entendendo a estrutura do documento e a navegando utilizando Expressões Regulares para o processo de adicionar os elementos da assinatura ao documento digital. Foi então elaborada uma rota de API para aplicação para geração do certificado, onde o código rodando no *browser* realiza uma conexão HTTPS com o próprio servidor. Esta rota de API recebe o CSR e passa para o micro serviço de PKI e retorna o conteúdo para aplicação. Após este desenvolvimento foi então feita a parte de conversão e assinatura do documento. Com o fim do desenvolvimento do processo de assinatura tendo os documentos assinados válidos pelo verificador do assinador avançado, foi então integrada uma camada de autenticação para aplicação redirecionando para o provedor de autenticação para realização do *login*, sendo então possível assinar um documento somente se o usuário estiver autenticado.

4.4 FLUXO DA APLICAÇÃO

Todas as Figuras referenciadas nesta seção estão disponíveis em Apêndice A. Após o acesso ao protótipo um usuário veria a Tela Inicial da aplicação presente em Figura 11, ao clicar no botão de realizar autenticação, o usuário é redirecionado a um provedor de identidade externa. O resultado desta ação é visível em Figura 12. Ao realizar sua autenticação com suas credenciais o usuário é redirecionado para a tela de assinatura, como visível em Figura 13. O usuário é apresentado com uma área onde ele pode arrastar documentos ou clicar para navegar nos arquivos de seu computador. Ao clicar na área indicada ou arrastar um documento PDF até ela a aplicação então muda para a Tela de PDF que apresenta o arquivo escolhido e dá duas opções a de assinatura e a de remoção do arquivo o que voltaria para Tela Inicial, o que é visível em Figura 14.

Ao clicar no botão de assinar todo o pseudo algoritmo descrito é previamente realizado, com a adição dos objetos que representam a assinatura dentro do documento PDF. Após a adição é criado o CSR contendo o resumo criptográfico do documento a ser assinado. É então feita a requisição para o servidor *Next* contendo o CSR e enviando em conjunto o *Token* de autenticação recebido na etapa de autenticação. O servidor então encaminha esta requisição ao microsserviço de PKI do assinador avançado. Ocorre então validação do *Token* de autenticação e emissão de um certificado de uso único pelo microsserviço de PKI. O servidor então recebe o certificado e o passa como resposta para o código executado no *browser* que iniciou a requisição.

O processo de assinatura então continua, com sua etapa final sendo o download do documento para os arquivos da máquina do usuário. O assinador não possui representação visual da assinatura, logo que a mesma pode ser enganosa, para um usuário que não possuí conhecimento de como as assinaturas em documentos PDF funcionam podendo achar que a representação visual da assinatura no documento é assinatura em si.

4.5 EXPERIMENTOS

Nesta seção são descritos e apresentados os resultados dos experimentos realizados no protótipo e nos documentos assinados pelo mesmo.

4.5.1 Validação da assinatura

O objetivo principal do trabalho pode ser validado realizando a assinatura e verificando se ela é válida, existem diversas ferramentas para a validação de assinaturas e as utilizadas foram a *pdfsig* e o verificador do Assinador Avançado.

```
pdfsig Downloads/IGI_14_assinado.pdf
Digital Signature Info of: Downloads/IGI_14_assinado.pdf
Signature #1:
  - Signer Certificate Common Name: Gabriel Cabral
  - Signer full Distinguished Name: E=gabriel.aristeu.cabral@gmail.com,
   CN=Gabriel Cabral,C=BR
  - Signing Time: Jun 09 2023 16:25:07
  - Signing Hash Algorithm: SHA-384
  - Signature Type: adbe.pkcs7.detached
  - Signed Ranges: [0 - 23448], [43450 - 44167]
  - Total document signed
  - Signature Validation: Signature is Valid.
  - Certificate Validation: Certificate issuer isn't Trusted.
```

Listing 4.2 – Validação no pdfsig

O *pdfsig* é um programa de terminal Ubuntu que verifica assinaturas PDF. Foi possível verificar a validade da assinatura usando o programa *pdfsig* como visível no Código 4.2. Pode-se analisar que a assinatura foi emitida para a pessoa Gabriel Cabral, e também possui informações como o e-mail. Estas informações vêm do provedor de autenticação do assinador avançado e estão ligadas a emissão do certificado no microsserviço de PKI. Pode-se analisar também outras informações como hora da assinatura, função de resumo criptográfico utilizada, assim como o alcance da mesma que neste exemplo encobre todo o documento. Isto também demonstra que programas tradicionais de verificação de assinatura funcionam com a aplicação desenvolvida possuindo uma retro-compatibilidade, única coisa que não é validada é o resumo contido dentro do certificado de uso-único.

```
pdfsig Downloads/IGI_14_assinado_assinado.pdf
Digital Signature Info of: Downloads/IGI_14_assinado_assinado.pdf
Signature #1:
  - Signer Certificate Common Name: Gabriel Cabral
  - Signer full Distinguished Name: E=gabriel.aristeu.cabral@gmail.com,
   CN=Gabriel Cabral,C=BR
  - Signing Time: Jun 09 2023 16:25:07
  - Signing Hash Algorithm: SHA-384
  - Signature Type: adbe.pkcs7.detached
```

```
  - Signed Ranges: [0 - 23448], [43450 - 44167]
  - Not total document signed
  - Signature Validation: Signature is Valid.
  - Certificate Validation: Certificate issuer isn't Trusted.
Signature #2:
  - Signer Certificate Common Name: Gabriel Cabral
  - Signer full Distinguished Name: E=gabriel.aristeu.cabral@gmail.com,
    CN=Gabriel Cabral,C=BR
  - Signing Time: Jun 09 2023 16:25:19
  - Signing Hash Algorithm: SHA-384
  - Signature Type: adbe.pkcs7.detached
  - Signed Ranges: [0 - 44308], [64310 - 65041]
  - Total document signed
  - Signature Validation: Signature is Valid.
  - Certificate Validation: Certificate issuer isn't Trusted.
```

Listing 4.3 – Validação de duas assinaturas no pdfsig

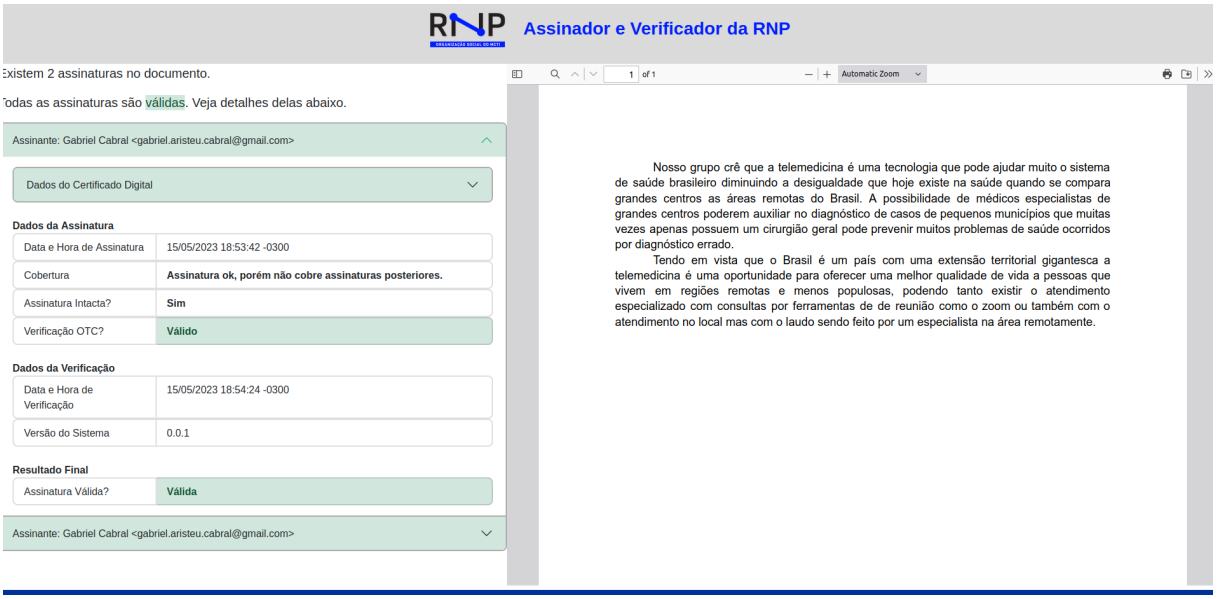

Figura 6 – Validação Assinador Avançado

Na *Listing* 4.3 pode-se ver 1 exemplo com 2 assinaturas. Neste exemplo é possível destacar a diferença da cobertura em uma assinatura e outra, logo que na realizada primeiro observa-se que nem todo o documento foi assinado. Isto acontece, pois o conteúdo da segunda assinatura foi adicionado após a realização da primeira com a atualização incremental do documento, logo que a primeira assinatura contínua válida após a adição da segunda. Também foi utilizada a ferramenta de validação de assinaturas do assinador avançado, logo que a mesma contempla também a validação do resumo criptográfico do documento incluído no certificado de uso único em sua criação. O resultado da verificação está disponível em Figura 6. Este documento é o mesmo validado na *Listing* 4.3 sendo possível ver que as duas assinaturas são válidas, e que a validação de OTC também é válida. Confirmando que o resumo criptográfico assinado é o mesmo que se encontra no certificado de cada assinatura.

4.5.2 Métricas da aplicação

Para um melhor entendimento das métricas de tempo na duração do processo de assinatura, o mesmo foi separado em 8 etapas as de: criação de *placeholder*, resumo criptográfico, criação de chaves, criação de CSR, resposta do microsserviço de ICP, preparação para assinatura, assinatura e *download* que se refere a transferência do arquivo do *browser* para o sistema de arquivos do sistema operacional. Após a separação foram então elencados 10 documentos PDF para realização de múltiplas assinaturas. Estes documentos variam de 23KB até 100MB. O limite superior de tamanho do documento para assinatura foi definido em 100MB, logo que as assinaturas neste documento levaram em média 45 segundos, tendo um alto consumo de memória e CPU durante o processo, com arquivos maiores o tempo de assinatura e consumo de memória e processamento seria maior o que prejudicaria a experiência de usuários finais. As assinaturas foram feitas usando o algoritmo de assinatura RSA e a função de resumo criptográfica utilizada foi o SHA-384.

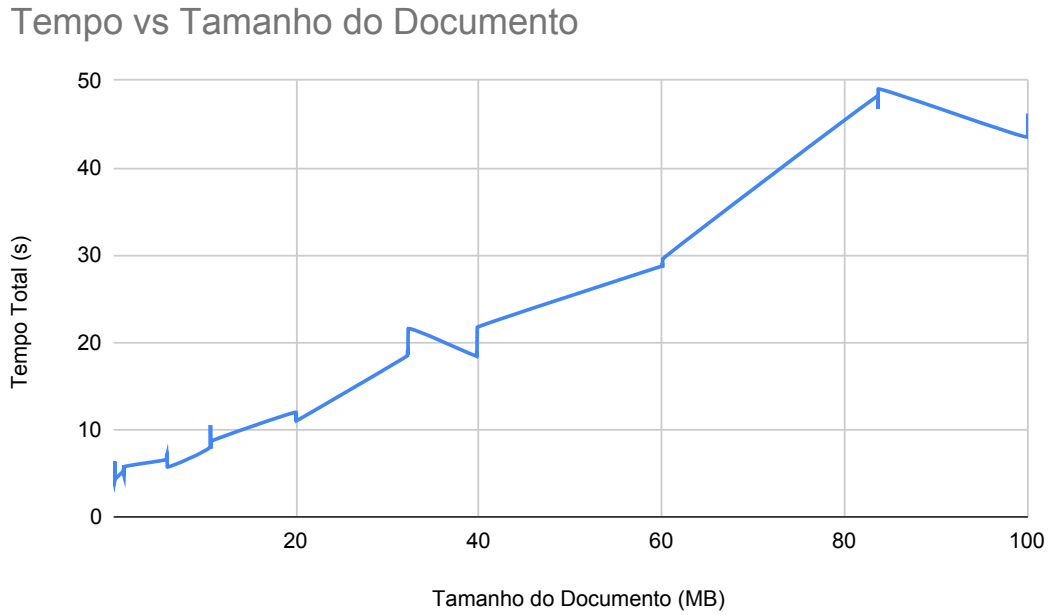

Figura 7 – Tempo Vs Tamanho do Documento

Como se observa em Figura 7, o tempo total da assinatura tende a acompanhar linearmente o tamanho do arquivo assinado, entretanto é possível observar que o tempo de assinatura pode variar até diminuindo para um arquivo maior, uma possibilidade para este comportamento é como a estrutura do arquivo está organizada. Outros 2 fatores importantes que podem variar muito são a latência entre enviar o CSR e receber a resposta do certificado e a geração de um par de chaves RSA. Logo em Figura 8 onde foi subtraído o tempo de resposta do Certificado e o tempo de criação do par de chaves, pode-se observar uma linearidade ainda maior sem tantas variações especialmente em tamanhos até 20 MB.

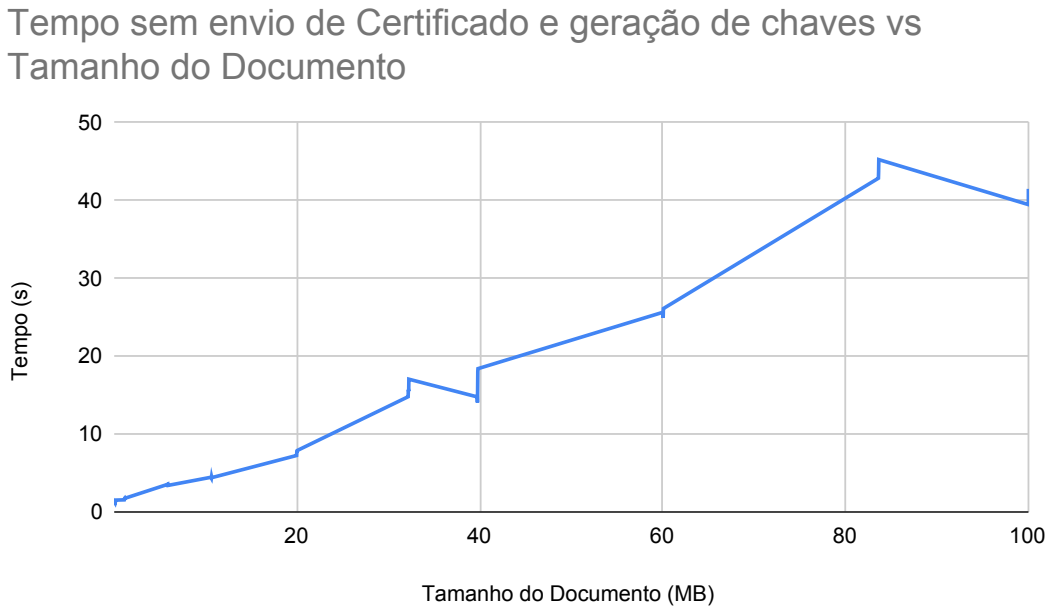

Figura 8 – Tempo sem envio de Certificado e geração de chaves Vs Tamanho do Documento

Em Figura 9 pode-se observar a porcentagem de cada etapa no processo de assinatura em comparação ao total do tempo. É notável a diminuição da importância da criação de chaves e resposta do microsserviço com o aumento do tamanho do arquivo, pois estas são etapas que não dependem do tamanho do documento ao ser assinado. É possível observar que em etapas como a de assinatura, resumo criptográfico e criação de *placeholder* existe uma forte correlação de tempo de duração da etapa com o tamanho do PDF.

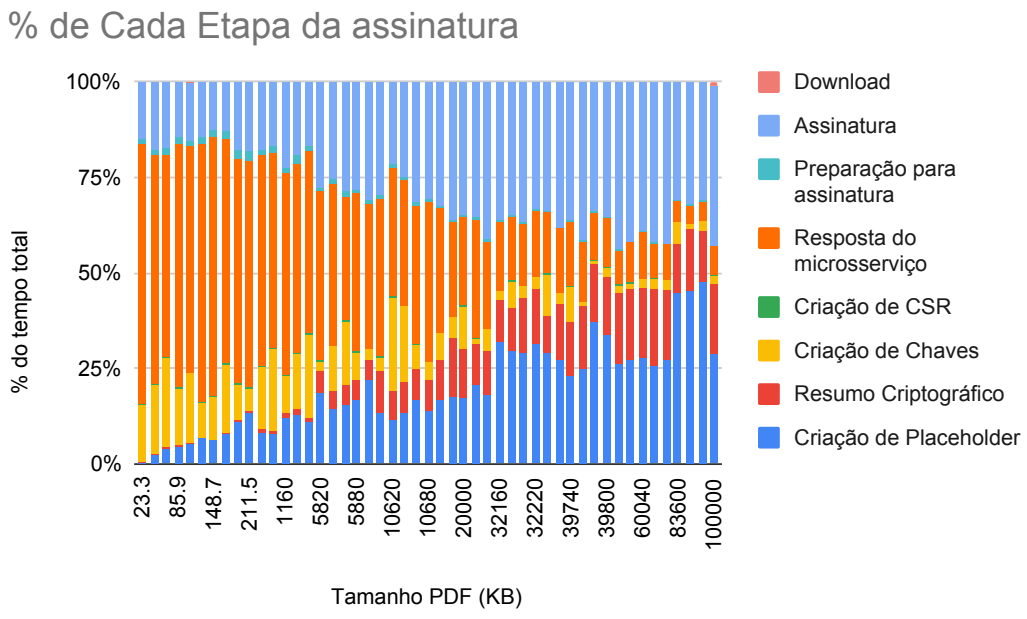

Figura 9 – Porcentagem de cada Etapa Vs Tamanho do Documento

Em Figura 10 pode-se observar o número de assinaturas presentes no documento naquele processo, em conjunto com a porcentagem do tempo de criação de *placeholder*, podendo inferir que o tempo necessário para o *parsing* do conteúdo e adição do *placeholder* tem uma correlação não só com o tamanho do PDF mas também com cada atualização incremental deixando o processo na maioria das vezes um pouco mais lento. Cada documento foi assinado várias vezes e diferentes documentos estão representados com diferentes cores no gráfico.

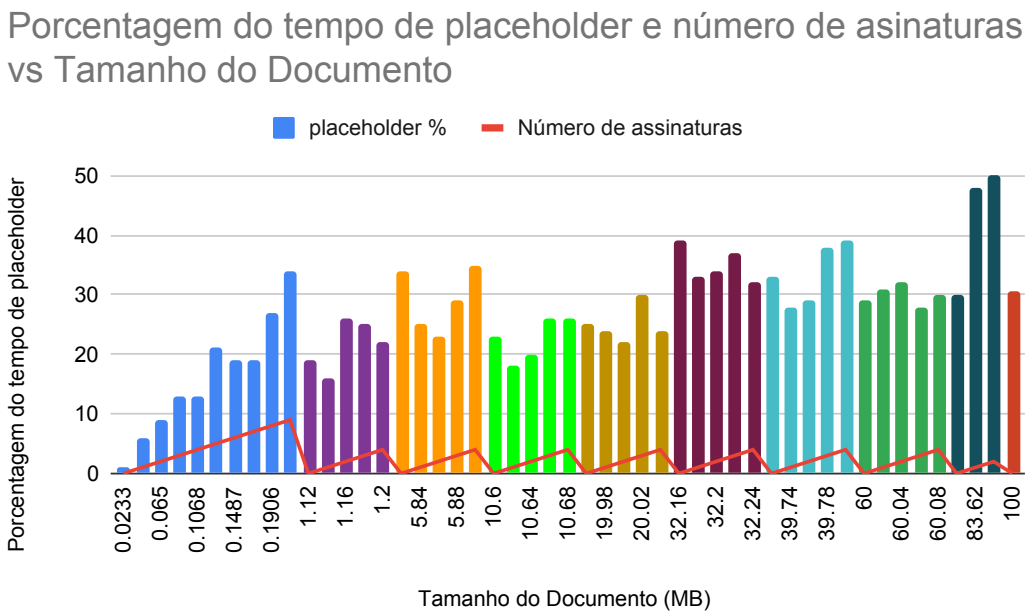

Figura 10 – Porcentagem do tempo de criação do *Placeholder* e número de assinaturas Vs Tamanho do Documento

Uma ideia que ia ser experimentada e apresentada era a comparação entre os algoritmos RSA e ECDSA nos quesitos de criação de chave e assinatura, porém infelizmente evidenciando mais uma vez a falta de suporte para aplicações de cunho criptográfico em *browsers*, não foi possível realizar o processo de assinatura utilizando o ECDSA. A biblioteca utilizada para a preparação da assinatura realizando as transformações de PKCS7 para PKCS10 no certificado recebido pela ICP não possui suporte para o algoritmo ECDSA não sendo possível então realizar esta comparação em busca do uso de um algoritmo mais performático.

5 CONCLUSÃO E TRABALHOS FUTUROS

Neste capítulo conclui-se o trabalho com uma reflexão dos resultados obtidos em relação aos objetivos propostos, assim como apresentação de problemáticas encontradas. Também se elencam possíveis trabalhos futuros que podem utilizar as contribuições realizadas neste trabalho.

5.1 CONCLUSÃO

Tendo em vista os objetivos propostos na introdução, o trabalho desenvolvido e os experimentos realizados, pode-se afirmar que o trabalho cumpre em grande parte os objetivos propostos. Foi desenvolvida uma aplicação que utilize certificados de uso único, gerados por uma AC, para assinatura de documentos digitais, foi confirmada a compatibilidade de verificação de assinatura tanto com validadores que levam em conta certificados de uso único como programas padrão. Garantiu-se também que a assinatura se deu no local de execução da aplicação e que o documento não foi enviado a nenhum servidor, com o descarte do certificado e chaves após o uso, garantindo também a privacidade do documento assinado. Pode-se constatar pelo experimentos feitos que a assinatura ocorre em um tempo aceitável para a maioria dos documentos PDF, entretanto a aplicação possui um limite de 100MB para os arquivos, de forma a melhorar a experiencia de uso da aplicação. Infelizmente não foi possível realizar testes de performance com diferentes algoritmos de assinatura clássicos devido a limitações técnicas nas bibliotecas utilizadas para manipulação de primitivas criptográficas. De forma geral, o trabalho atinge os principais objetivos propostos, e quando não atinge é devido a falta de suporte de bibliotecas para aplicações que executam no *browser* logo que usualmente ações criptográficas e de manipulação de arquivos ocorrem ou em servidores ou aplicações *desktop* próprias para isto, o que introduziu uma necessidade de compreensão da estrutura de arquivos PDF elevada neste trabalho, assim como as operações realizadas criptograficamente durante o processo de assinatura.

5.2 TRABALHOS FUTUROS

Este trabalho abre uma frente para diversos trabalhos futuros, que utilizam a assinatura de documentos digitais no cliente de execução, mas uma melhoria direta seria a elaboração de um sistema de gerenciamento de assinaturas, tendo a possibilidade de convidar pessoas a assinarem um mesmo documento e ter o controle de quais usuários requisitados já assinaram ou não o documento. Outro trabalho futuro é um sistema mais generalista que possa assinar diversos tipos de documentos logo que o escopo deste trabalho foi delimitado apenas a documentos PDF. O desenvolvimento de uma biblioteca que consiga fazer a atualização incremental e *parsing* de um documento PDF em *browser* também é um trabalho futuro interessante a ser elencado. A investigação de uso de algoritmos de assinatura pós-quânticas é uma possibilidade interessante

a ser investigada, entretanto, é necessário esperar que estes algoritmos estejam disponíveis nos *browsers* ou desenvolver o código deles neste contexto.

REFERÊNCIAS

ACROBAT. 2023. https://helpx.adobe.com/acrobat/using/signing-pdfs.html. Acesso em: 2023-06-12.

ASSINATURA Eletrônica do GOV.BR. 2023. https://www.gov.br/governodigital/pt-br/assinatura-eletronica. Acesso em: 2023-06-12.

BRASIL. **LEI 14063**. 2020. Acesso em 2022-12-10. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2019-2022/2020/lei/l14063.htm.

COUNCIL, N. R. **Computers at Risk: Safe Computing in the Information Age**. Washington, DC: The National Academies Press, 1991. ISBN 978-0-309-04388-5. Disponível em: https://nap.nationalacademies.org/catalog/1581/computers-at-risk-safe-computing-in-the-information-age.

DIFFIE, W.; HELLMAN, M. New directions in cryptography. **IEEE transactions on Information Theory**, IEEE, v. 22, n. 6, p. 644–654, 1976.

DIGITAL around the world - datareportal – global digital insights. 2023. Disponível em: https://datareportal.com/global-digital-overview.

DOCUMENT management — Portable document format — Part 2: PDF 2.0. Geneva, CH, 2020. v. 2020.

DOCUSIGN. 2023. https://www.docusign.com/blog/what-electronic-signature. Acesso em: 2023-06-12.

E-UFSC. 2023. https://e.ufsc.br/. Acesso em: 2023-06-12.

FAROOQ, S. M.; HUSSAIN, S. S.; USTUN, T. S. Elliptic curve digital signature algorithm (ecdsa) certificate based authentication scheme for advanced metering infrastructure. In: **2019 Innovations in Power and Advanced Computing Technologies (i-PACT)**. [S.l.: s.n.], 2019. v. 1, p. 1–6.

GUTMANN, P. Pki: it's not dead, just resting. **Computer**, v. 35, n. 8, p. 41–49, 2002.

ICP-BRASIL. 2023. https://www.gov.br/iti/pt-br/assuntos/icp-brasil. Acesso em: 2023-06-12.

JAVASCRIPT. 2023. https://www.jetbrains.com/lp/devecosystem-2021/javascript/. Acesso em: 2023-05-09.

KALISKI, B. **PKCS #7: Cryptographic Message Syntax Version 1.5**. RFC Editor, 1998. RFC 2315. (Request for Comments, 7392). Disponível em: https://www.rfc-editor.org/info/rfc2315.

KOHNFELDER, L. Towards a practical public-key cryptosystem. 05 1978.

MAYR, L. et al. One-time certificates for reliable, inclusive and secure document signing. **ESORICS**, 2023.

MERKLE Town. 2023. https://ct.cloudflare.com/. Acesso em: 2023-05-08.

MORIARTY, E. K. et al. **PKCS #12: Personal Information Exchange Syntax v1.1**. RFC Editor, 2014. RFC 7292. (Request for Comments, 7392). Disponível em: https://www.rfc-editor.org/info/rfc7292.

MYERS, M. et al. **Internet X.509 Certificate Request Message Format**. RFC Editor, 1999. RFC 2511. (Request for Comments, 7392). Disponível em: https://www.rfc-editor.org/info/rfc2511.

NEXTJS. 2023. https://nextjs.org/. Acesso em: 2023-05-09.

NYSTORM, M.; KALISKI, B. **PKCS #10: Certification Request Syntax Specification Version 1.7**. RFC Editor, 2000. RFC 2986. (Request for Comments, 7392). Disponível em: https://www.rfc-editor.org/info/rfc2986.

ORAM, A. **How hashing and cryptography made the internet possible**. 2022. Disponível em: https://developers.redhat.com/articles/2022/09/20/how-hashing-and-cryptography-made-internet-possible.

OXFORD Dictionary. 2023. https://www.oxfordlearnersdictionaries.com/us/definition/english/front-end_1. Acesso em: 2023-05-09.

REACT. 2023. https://react.dev/. Acesso em: 2023-05-09.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. M. A method for obtaining digital signatures and public-key cryptosystems. **Commun. ACM**, v. 21, n. 2, p. 120–126, 1978. Disponível em: http://dblp.uni-trier.de/db/journals/cacm/cacm21.html#RivestSA78.

SELHAUSEN, K. M. zu. **Security of PDF Signatures**. 2018.

STALLINGS, W. **Cryptography and Network Security: Principles and Practice**. 6th. ed. USA: Prentice Hall Press, 2013. ISBN 0133354695.

STANDARDS, N. I. of; TECHNOLOGY. **Digital Signature Standard (DSS)**. Washington, D.C., 2013.

SUBRAMANYA, S.; YI, B. Digital signatures. **IEEE Potentials**, v. 25, n. 2, p. 5–8, 2006.

A TELAS DA APLICAÇÃO

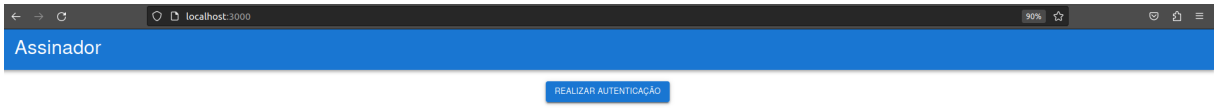

Figura 11 – Tela Inicial

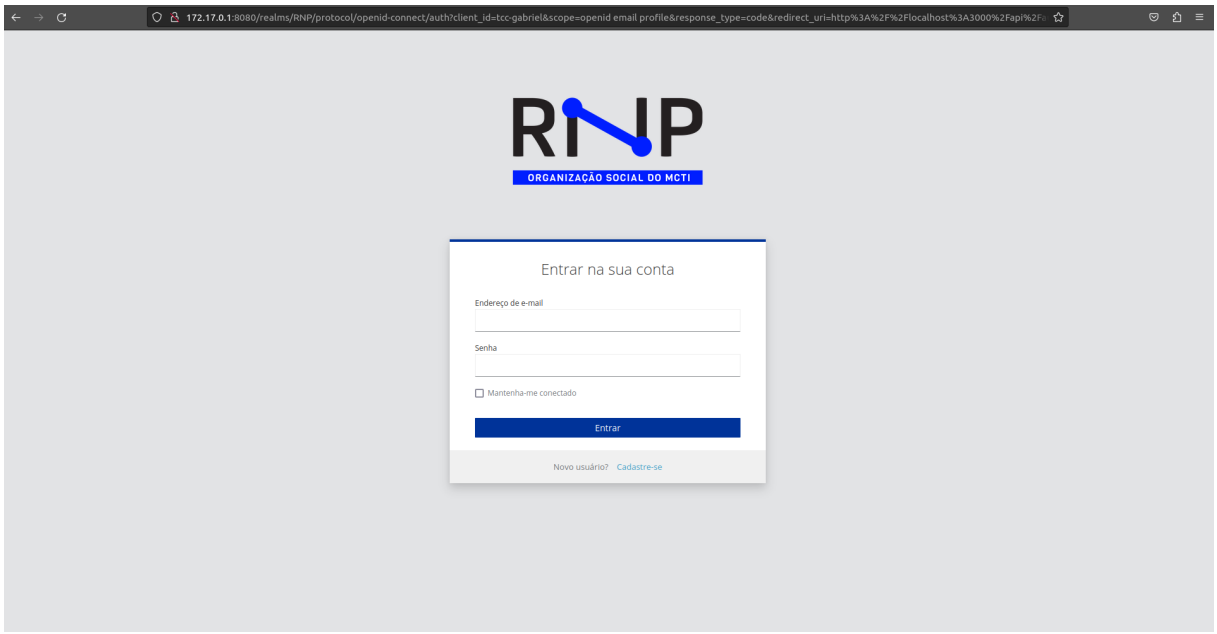

Figura 12 – Tela Da Autenticação

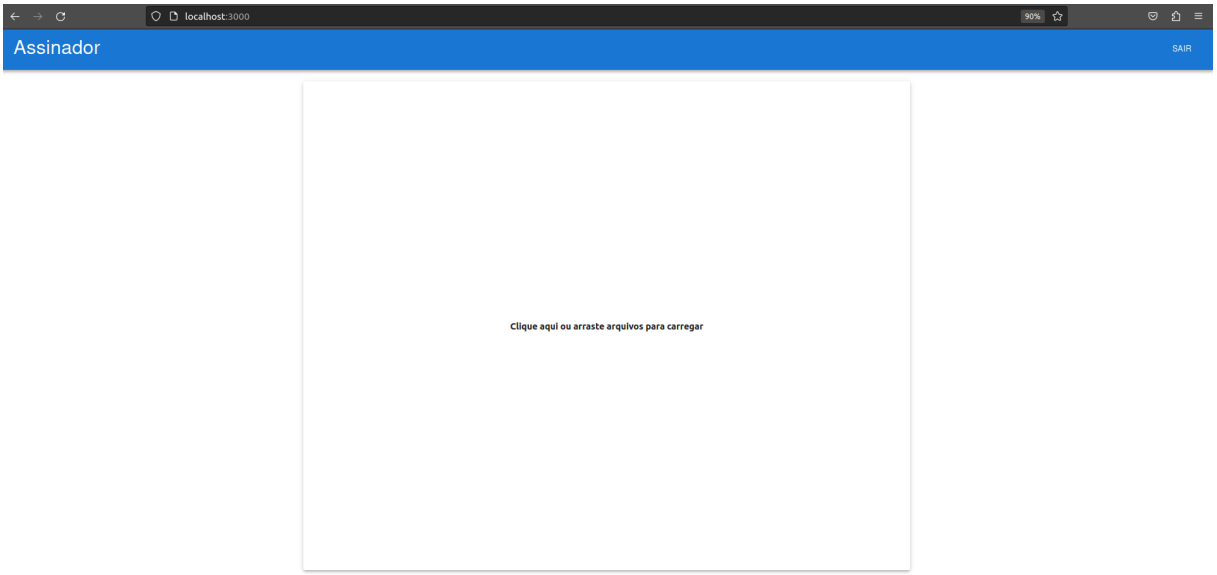

Figura 13 – Tela com usuário Autenticado

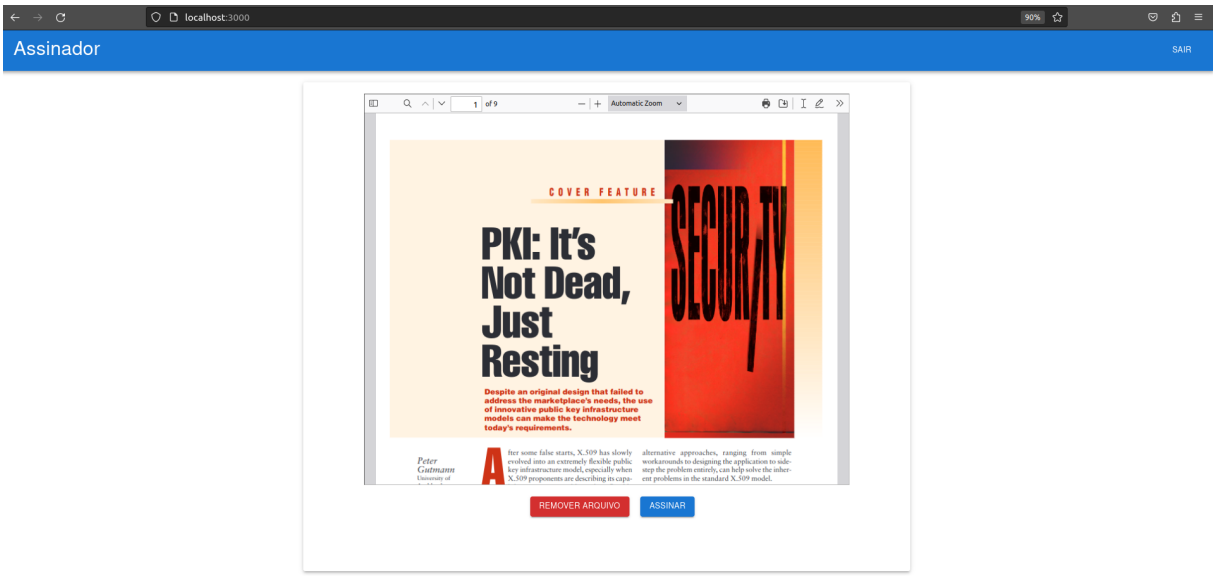

Figura 14 – Tela com Documento Carregado

B ARTIGO SBC

Assinatura de Documentos com Certificado de Uso Único e Chaves Geradas no Cliente

Gabriel Aristeu Cabral[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)

Abstract. *Given the continuous advances in technology and the digitalization of several services globally, it is necessary to democratize access to these services, including digital signatures. Digital signatures introduce a knowledge barrier to their correct use. A lot of digital signature applications do the whole signature process on the backend; in that case, the upload of the file to be signed is necessary, which can invade the signer's privacy. With that in context, it is proposed to create a web application for signing digital documents in PDF format using one-time certificates, with the signature and key creation process occurring in the browser.*

Resumo. *Com o avanço tecnológico e digitalização de diversos serviços mundialmente se faz necessário democratizar o acesso a estes serviços, entre eles as assinaturas digitais. As assinaturas digitais introduzem uma barreira de conhecimento quanto ao seu uso correto. Muitas aplicações de assinaturas digitais, realizam a assinatura em um servidor, com a necessidade de* upload *do arquivo para uma nuvem, o que pode invadir a privacidade do assinante. Tendo em vista esta problemática propõe-se a criação de uma aplicação WEB para realização de assinaturas digitais usando certificados de uso único em documentos no formato PDF com todo o processo de assinatura e criação de chaves no cliente.*

1. Introdução

Tendo em vista o contexto atual de assinatura digital, pode-se compreender que a principal problemática das ICPs atuais é garantir a segurança da chave privada. Logo que muitas vezes os usuários finais são pessoas que não possuem conhecimento da importância das chaves privadas e negligenciam o seu armazenamento. Além de que os dispositivos para o armazenamento de chaves como Tokens e Smart Cards possuem um custo e requerem que o usuário tenha conhecimento de como acontece a revogação em caso de perda ou vazamento do conteúdo dos dispositivos. Outro problema é a privacidade do documento logo que muitas vezes a assinatura ocorre em um servidor remoto não sendo possível garantir que o sistema não guardou uma cópia do mesmo ao final do processo de assinatura.

O aumento da digitalização no mundo trouxe a tona a necessidade de democratizar o acesso às tecnologias tidas como estado da arte de forma que as mesmas possam ser uteis a população, este trabalho visa diminuir a barreira de conhecimento necessária para uma pessoa realizar a assinatura de documentos digitais. Assim como fornecer a privacidade que o documento assinado nunca foi transmitido a nenhum servidor. O escopo deste trabalho é a criação de uma aplicação que realize assinaturas digitais em documentos no formato PDF, no *browser* em que a aplicação está rodando, sem a necessidade de *upload* do arquivo a algum servidor e utilizando certificados digitais de uso único.

2. Conceitos

Nesta seção são apresentados conceitos base para um bom entendimento da proposta.

2.1. Criptografia Assimétrica

A criptografia assimétrica é um conceito básico de segurança no mundo atual. Ela é utilizada para a navegação segura em páginas *WEB* com protocolos como o *Hypertext Transfer Protocol Secure (HTTPS)* e o *Secure Shell (SSH)*. A criptografia assimétrica consiste em utilizar um par de chaves para realizar a transmissão segura de uma mensagem, onde o que uma chave cifra a outra decifra e vice-versa, logo é possível deixar uma chave pública e de conhecimento de todos e outra privada da qual apenas o dono tem acesso [Diffie and Hellman 1976].

É possível comprovar a autenticidade de uma mensagem cifrada usando criptografia assimétrica realizando o processo de deciframento com o par de chaves de quem cifrou. Logo que esta mensagem cifrada pela chave privada só será decifrada com a chave pública do respectivo par. O uso de diferentes chaves de cifragem e decifragem num cripto-sistema é algo bom, tendo em vista que sistemas assimétricos permitem que a chave de cifragem seja pública e disponível a todos enquanto a de decifragem é restrita apenas aos que tem acesso e podem decifrar a informação [Council 1991].

2.2. Função de Resumo Criptográfico

Defini-se como uma Função de Resumo Criptográfico o seguinte. Um valor h é gerado for uma função H na forma: $h = H(M)$. Para diferentes tamanhos M o tamanho de h é fixo. Dado um h é computacionalmente impossível derivar M. Também é impossível achar um par (x, y) onde $H(x) == H(y)$. Logo uma função de Resumo Criptográfico gera um resumo criptográfico para bytes inseridos, sendo possível confirmar a integridade de uma mensagem verificando seu resumo [Stallings 2013]. As funções de Resumo Criptográfico possuem justamente a função de confirmar que não houve modificação em um conteúdo, logo podem ser adicionadas ao final de uma mensagem m. Calcula-se então $hm = H(m)$, é construída então uma mensagem final mf, tendo em vista que α delimita o fim da mensagem m, na seguinte forma $mf = m \cup \alpha \cup hm$. O receptor de mf pode então separar a mensagem em α e obter os valores de (m', hm). Calcula-se $rhm = H(m')$ sendo possível comprovar que não houve alteração do conteúdo m com $rhm == hm$, logo que se houve alteração os valores do resumo calculado e do resumo recebido não serão iguais.

2.3. Infraestrutura de Chaves Públicas

Proposta em sua primeira iteração por [Diffie and Hellman 1976] com a ideia de uma estrutura hierárquica para a criação, armazenamento e distribuição de chaves públicas com o intuito de viabilizar a criptografia assimétrica em um contexto de redes, em sua proposta um arquivo público guardaria chaves públicas e as distribuiria. [Kohnfelder 1978] então tentou implementar em sua época uma Infraestrutura de Chaves Públicas (ICP) e criou o conceito de um certificado, para contornar a necessidade mútua de comunicação com o objetivo de buscar as chaves no arquivo público, este certificado é emitido pelo arquivo público e assinado com sua chave privada e associa um par de chaves a um membro da comunicação, logo ambas as partes podem confirmar a veracidade do certificado usando a chave pública do arquivo público e confirmar que o certificado e chaves são de um membro da comunicação.

2.4. Assinatura Digital

Uma assinatura digital tenta emular as mesmas características de uma assinatura feita a mão em papel. Documentos em papel e digitais têm que ser idênticos nos sentidos de validade das assinaturas presentes. Os documentos em papel são certificados utilizando a forma física da assinatura. Já os digitais utilizam o retorno dos algoritmos de assinatura digital para cumprir o mesmo propósito de validação e autenticação dos documentos digitais [Subramanya and Yi 2006]. Para garantir validação e autenticidade para documentos digitais, os algoritmos de assinatura digital utilizam diversos conceitos criptográficos como criptografia assimétrica, funções de Resumo Criptográfico e uma infraestrutura de chaves públicas.

O processo de assinatura de um documento digital pode ser resumido na Figura 1. Onde basicamente para uma mensagem M é utilizada uma função de resumo criptográfico para obter $resumo = H(m)$. É inserido então o resumo da mensagem numa função de cifragem assimétrica onde é utilizada a chave privada do assinador para obter a assinatura digital com $assinatura = C(resumo, chave\_privada)$. É então anexada a assinatura em conjunto a mensagem. Desta forma qualquer pessoa poderia verificar a integridade e autenticidade da assinatura decifrando com a chave pública. Para o processo de verificação de uma assinatura se obtém o resumo cifrado $resumo\_obtido = D(assinatura, chave\_publica)$. Calcula-se então o resumo da mensagem recebida e compara-se com o resumo obtido para garantir que não houve nenhuma alteração na mensagem após a assinatura e confirma-se que o assinante é o detentor daquele par de chaves.

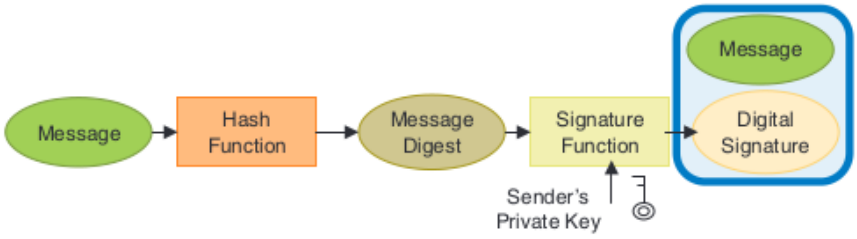

Figura 1. Assinatura de uma Mensagem. Retirado de: [Subramanya and Yi 2006]

2.5. Certificado Digital

O conceito de certificado digital é introduzido por [Kohnfelder 1978]. Ele serve como uma atribuição de um par de chaves a uma entidade, uma pessoa vai a uma autoridade certificadora (AC) então a AC após confirmar a identidade de quem requisitou emite um certificado para o requisitante. Esse certificado é um arquivo digital definido em [Myers et al. 1999]. Ele é um arquivo X.509 e funciona similarmente a um dicionário com chaves, que contém informações sobre o arquivo e seu dono, como nome, e-mail, data de validade, endereço, entre outras possíveis extensões. Ele também contém a chave pública de seu dono e uma assinatura digital para comprovar a posse do par de chaves e caminho de certificação. Pode se então verificar tanto a assinatura do certificado digital pela AC final e o caminho de certificação para emissão do certificado podendo então confiar que aquele certificado foi emitido por uma instituição confiável.

2.6. Revogação de Certificados

Um dos problemas das ICPs atuais é a revogação de certificados inválidos. A estratégia utilizada no padrão atual da indústria é uma lista de certificados revogados geralmente mantida pela AC Final. Em caso de perda ou extravio o dono daquele certificado pede a AC Final a inclusão de seu certificado em uma lista de certificados revogados, a inclusão em si, pode demorar logo que geralmente estas listas são atualizadas em um intervalo fixo de tempo, após a inclusão um programa de verificação de assinaturas pode também atualizar suas listas em um intervalo fixo de tempo, porém, diferente do intervalo da AC Final o que acarretará o programa reconhecer como válidas assinaturas posteriores a revogação do certificado.

O fato destas listas de revogação se basearem em soluções antigas para cartões de crédito traz algumas problemáticas como a baixa frequência de emissão, alto custo de transmissão e verificação assim como a suscetibilidade a ataques de negação a serviço [Gutmann 2002]. Outro problema que ocorre devido aos altos custos computacionais de verificar se um certificado está ou não na lista e de atualizar a lista periodicamente, o que faz com que muitos programas nem atualizem suas listas de Revogação, focando em desempenho ao invés de segurança.

3. Certificado Digital de uso-único

Os certificados digitais de uso único buscam resolver diversas problemáticas com as atuais infraestruturas de chaves públicas principalmente quando o assunto são assinaturas digitais com foco em pessoas físicas. Eles buscam remover a necessidade de guardar seu certificado em um dispositivo próprio para armazenamento, diminuindo não só o custo, mas também a barreira de conhecimento necessária para a manutenção correta de um certificado digital. Elimina-se a necessidade de manter um certificado e par de chaves que é reutilizado para várias assinaturas, e para cada documento assinado é criado um novo par de chaves utilizando também um novo certificado único emitido para aquele documento.

O certificado possui em suas extensões além da identificação de seu proprietário com informações como nome e e-mail também o resumo criptográfico do conteúdo a ser assinado. O resumo criptográfico é feito uma vez quando o documento está pronto para ser assinado, e enviado em conjunto com o CSR para a AC que emitirá o certificado. Este resumo não pode ser alterado uma vez no certificado digital, logo pode-se confirmar que este certificado foi emitido apenas para assinar aquele conteúdo. A cada assinatura então é criado um novo par de chaves e emitido um novo certificado e ao fim do processo de assinatura tanto o certificado quanto o par de chaves pode ser descartado, logo que o mesmo não pode ser reutilizado [Mayr et al. 2023].

Ao realizar uma assinatura o Certificado contendo a chave pública e incluído dentro do conteúdo de mensagem assinada, sendo possível então do ponto de vista do assinante descartar o par de chaves e certificados utilizados na assinatura, ou mesmo torna-lá pública em. Do ponto de vista de quem está verificando a assinatura, poderá encontrar dentro do objeto de assinatura o certificado e o conteúdo da assinatura digital podendo então utilizar a chave pública dentro do certificado para realizar o processo de validação.

Caso um atacante consiga ter acesso a um certificado e um par de chaves de um certificado de uso único, se o mesmo tentar assinar um outro documento o resumo criptográfico do novo conteúdo assinado será diferente do contido dentro certificado, sendo

possível então na verificação da assinatura constatar que o certificado não foi emitido para assinar aquele documento, tornando assim a assinatura inválida, entretanto, o validador tem que estar preparado para realizar a validação do resumo criptográfico, do contrário o mesmo dirá que assinatura é válida. Os verificadores atuais situados fora do contexto de certificados de uso único não realizam esta verificação. Esta abordagem também remove a necessidade de manutenção e uso de uma lista de certificados revogados, logo que se um certificado for compromissado caso outro documento seja assinado haverá disparidade do resumo assinado e do presente no certificado. A ausência de uma lista de revogação de certificados diminui também a complexidade do processo de verificação da assinatura, logo que não é necessário procurar a Lista de Certificados Revogados da AC emissora [Mayr et al. 2023].

3.1. Assinador Avançado

O Assinador Avançado é um projeto do LabSEC que visa a criação de uma ICP com certificados de uso único, conseguindo prover a assinatura de documentos digitais de maneira rápida e eficiente para a população, sem a necessidade da manutenção de um certificado, ou de guardar ele em um servidor. Como para cada documento assinado é emitido um novo certificado, elimina-se a responsabilidade do usuário e também como as chaves e certificados são apenas criados, usados e descartados ele acaba sendo uma alternativa viável e segura para a assinatura de documentos digitais.

O projeto hoje suporta a assinatura e verificação de documentos digitais, diferentemente do trabalho proposto o processo de assinatura ocorre no *backend* da aplicação, o que implica no *upload* do documento ser necessário para a assinatura em conjunto com a criação de chaves ocorrer no lado do servidor. O projeto é dividido em diversos microsserviços, dos quais os de ICP e de verificação de assinaturas foram utilizados sem nenhuma modificação como apoio para o desenvolvimento deste trabalho, o microsserviço de ICP tem a função de receber um CSR e emitir um respectivo certificado.

4. PDF

O *Portable Document Format (PDF)* é um tipo de arquivo que representa, documentos e formulários no ambiente digital. Estes documentos possuem a característica de serem idênticos indiferentemente da aplicação utilizada para os ver ou processar. Não importa também o sistema operacional em que a aplicação utilizada está sendo executada. Os documentos PDF utilizam uma estrutura estrita e contém diversas informações sobre como o conteúdo deve ser apresentado ao usuário final [zu Selhausen 2018]. Um documento PDF pode ser divido em algumas partes, como o Cabeçalho, o Corpo a tabela de Referências e o *Trailer*. Após os objetos de páginas e conteúdos existe um objeto de *Trailer* que anuncia o fim do documento assim como uma tabela de referências que guarda para cada objeto o *byte* aonde seu conteúdo começa. A Figura 2 ilustra a macro-estrutura de um arquivo PDF.

4.1. Atualização Incremental

Existem dois tipos de atualizações que podem ser feitas em um arquivo PDF já existente. Ao adicionar um objeto pode-se incluí-lo dentro do documento antes da tabela de referência e atualizá-la, adicionando assim o conteúdo atualizado, entretanto caso haja uma

Figura 2. Estrutura de um arquivo PDF. Adaptado de: [zu Selhausen 2018]

assinatura no documento isto a tornará inválida, logo que houve alterações no conteúdo assinado no documento após assinatura, quebrando a verificação usando uma função resumo criptográfico do mesmo. Atualizações incrementais possibilitam aplicações adicionarem conteúdo novo ou atualizarem algo já existente no fim de um arquivo PDF. Adicionalmente, isto permite modificar os documentos PDF que contém assinaturas digitais sem invalidar a assinatura, pois o conteúdo originalmente assinado não é modificado [zu Selhausen 2018].

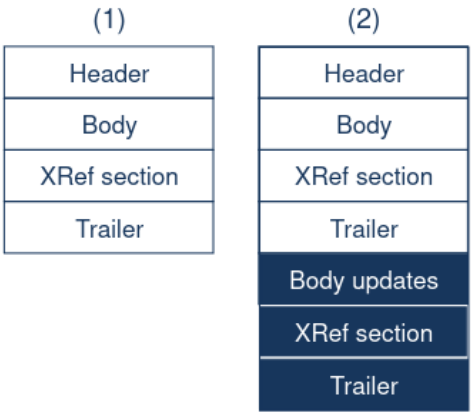

Figura 3. Estrutura de um arquivo PDF com atualização incremental. Retirado de: [zu Selhausen 2018]

Ao atualizar um arquivo incrementalmente é adicionado a atualização dos conteúdos do corpo do PDF no fim do mesmo podendo até redefinir objetos previamente definidos, então é adicionada uma nova tabela de referência para quaisquer elementos atualizados ou adicionados e finaliza-se com um novo objeto trailer que aponta para o último Trailer disponível. O resultado deste processo é mostrado em Figura 3.

5. Assinaturas Digitais em PDF

Ao realizar uma assinatura em um documento PDF é necessário utilizar alguns elementos previamente explicados como os objetos de *Catalog*, *Page*, *Widget*. Assim como o processo de atualização incremental e o objeto de *Sig*.

5.1. Objeto Sig

Um objeto *Sig* contém os conteúdos de uma assinatura digital em um documento PDF. O objeto *Sig* é representado com um dicionário com diversas entradas. A chave $/Contents$ possuí o conteúdo da assinatura. A chave $/ByteRange$ é um *array* com 4 entradas representando as seções do documento que foram assinadas, com o primeiro número sendo o *byte* do começo do conteúdo assinado (Geralmente 0), o segundo número sendo o *byte* que somado ao primeiro chega ao começo do valor da chave de $/Contents$ do objeto $/Sig$. O terceiro número sendo o *byte* partir do fim da assinatura, e o quarto um número que somado ao terceiro resulta no último *byte* assinado. Desta forma o objeto de *Sig* geralmente engloba todo o conteúdo do arquivo menos o conteúdo da assinatura do mesmo [ISO 2020].

A chave $/Type$ tem seu valor como *Sig* para representar que o objeto é uma assinatura. A chave $/Filter$ indica qual o validador da assinatura, alguns valores possíveis são Adobe.PPKLite, Entrust.PPKEF, CICI.SignIt, e VeriSign.PPKVS [ISO 2020]. A chave de $/SubFilter$ indica a codificação do conteúdo da assinatura, o $/SubFilter$ utilizado neste trabalho foi o adbe.pkcs7.detached. Já a chave M representa a data da assinatura. Existem outras chaves utilizadas para representar diferentes atributos de uma assinatura disponíveis em [ISO 2020]. A chave $/Contents$ para o $Subfilter$ selecionado neste trabalho irá conter uma mensagem criptografada $PKCS\#7$ contendo não só a assinatura mas também o certificado utilizado na assinatura e o caminho de certificação, com este conteúdo binário sendo codificado em DER. Pode-se então utilizar o certificado contido dentro da chave $/Contents$ para a verificação da assinatura. A estrutura de um objeto *Sig* pode ser vista em *Listing* 1.

Listing 1. A estrutura padrão de um objeto *Sig*

```
13 0 obj
<<
/Type /Sig
/Filter /Adobe.PPKLite
/SubFilter /adbe.pkcs7.detached
/M (D:20230410141326-03'00')
/Contents <Conteudo>
/ByteRange [0 23675 42621 15919]
>>
endobj
```

5.2. Processo de assinatura

No ponto de vista técnico, o processo de assinatura pode ser descrito pelo seguinte algoritmo:

1. Lê o conteúdo do arquivo PDF.

2. Adiciona um objeto Sig após o $Trailer$ do arquivo, o valor da chave $Contents$ deste objeto Sig é populado com um valor $placeholder$. Este valor pode ser a letra A repetida "n"vezes. A chave $ByteRange$ é populada levando este valor de *placeholder* em consideração.
3. Adiciona um objeto do tipo $Widget$ que referencia o objeto do Tipo Sig definido.
4. Redefine os objetos de $Catalog$ e de $Page$ adicionando uma referencia ao objeto $Widget$ adicionado.
5. Adiciona uma tabela de referencias no final do arquivo assim como um objeto $Trailer$ que aponta para o $Trailer$ anteriormente definido.
6. Atualiza os valores de $ByteRange$ do objeto de Sig levando em conta a redefinição dos outros objetos.
7. Calcula o resumo criptográfico referente ao $ByteRange$ definido.
8. Assina o resumo criptográfico utilizando a chave privada do assinante. E a codifica em relação ao valor definido na chave $Subfilter$ do objeto Sig.
9. Substitui o valor $placeholder$ na chave $Contents$ do objeto de Sig pelo valor codificado. Se o $placeholder$ for maior que o tamanho do conteúdo da assinatura é adicionado o carácter 0 n vezes para que ambos tenham o mesmo tamanho e não haja mudança no $ByteRange$.

O processo de assinatura pode incluir processos mais específicos de acordo da complexidade de sua aplicação e uso [zu Selhausen 2018]. Em uma infraestrutura de certificados de uso único entre as etapas 7 e 8 do processo haveria a criação de um par de chaves e um *Certificate Signing Request* (CSR) incluindo o resumo criptográfico do documento para uma ICP, que por sua vez, responderia com o certificado, este então seria utilizado para o processo de assinatura.

5.3. Processo de verificação

O processo de verificação de uma assinatura digital em um documento PDF é similar ao de verificação de uma assinatura de uma mensagem assinada. Com a diferença de que a assinatura encontra-se no meio do documento assinado. Deve-se então calcular o resumo criptográfico dos conteúdos explicitados pelo $ByteRange$ da assinatura e comparar com o valor assinado dentro da chave $Contents$ do objeto Sig. Após confirmar a integridade da assinatura se deve analisar a cadeia de certificação do certificado assinado, garantindo que exista um emissor confiável no caminho de certificação e garantindo que o mesmo não foi revogado [zu Selhausen 2018]. Em uma infraestrutura de certificados de uso único deve-se garantir que o resumo criptográfico presente dentro do certificado é o mesmo resumo assinado, pois o certificado foi emitido apenas para assinatura daquele documento.

6. Desenvolvimento

Nesta seção será detalhada a proposta, assim como o processo de criação da mesma e os resutlados obtidos.

6.1. Proposta

O objetivo deste trabalho se trata da criação de uma aplicação *Front End* que realize a assinatura de documentos digitais. O processo de assinatura deve acontecer apenas no *browser* onde a aplicação está rodando. Será utilizada uma infraestrutura de certificados de uso único para assinatura dos documentos, contendo o resumo do conteúdo assinado

em uma das extensões do certificado gerado para assinatura daquele documento. O documento digital a ser assinado também não sairá do local de execução, assim como o par de chaves gerado para o certificado, com apenas a chave pública trafegando em rede contida em um CSR. Sem a necessidade de fazer o *upload* do documento digital a ser assinado para um servidor, conserva-se a privacidade do documento assinado.

Esta aplicação rodará como um micro serviço de uma ICP sendo seu braço de assinatura, e toda vez que um documento for assinado ela irá criar um CSR e fará sua única requisição externa mandando apenas para o microsserviço de PKI o CSR contendo resumo criptográfico do documento e a chave pública dos pares de chave gerados. O PKI então retornará o certificado requisitado em conjunto com o caminho de certificação que o emitiu. A aplicação então realiza a assinatura no local de execução usando a resposta do microsserviço de PKI. Nenhuma das ações feitas pelo PKI foi modificada neste trabalho em comparação com o assinador avançado disponibilizado pelo LabSec. [1]

6.2. Ferramentas

O *JavaScript* é a linguagem de programação padrão para o desenvolvimento de sistemas de *Front End* que rodam no *Browser* e Sites em geral. Entretanto, outra opção é usar o TypeScript, que adiciona tipagem estática ao *JavaScript* melhorando a legibilidade e manutenção do código, logo foi utilizado o *TypeScript* para elaboração do código do projeto. A biblioteca *React* também foi utilizada, ela é focada na construção de interfaces do usuário reativas a mudanças de dados, fazendo isto de forma fácil e intuitiva. Foi utilizado o *NextJs* para a elaboração do protótipo, ele engloba o *React* servindo o mesmo e dando a possibilidade da criação de rotas de *API* no mesmo projeto. Foi utilizada a biblioteca *next-auth* para integração com o provedor de autenticação.

Para o desenvolvimento do algoritmo de assinatura na parte criptográfica foram utilizadas bibliotecas como o *pkijs* para a geração do CSR, *asn1js* pois ela é uma dependência da *pkijs* e ajuda com diversas funções de conversão de valores. A biblioteca *Node-Forge* para lidar com conversões como entre a resposta da ICP de um padrão PKCS7 para o padrão PKCS12 e assinatura do PDF. Também foi utilizada a própria biblioteca *Crypto* nativa do *JavaScript*, disponível em *browsers* e responsável pela geração do par de chaves e execução das funções de resumo criptográfico. Para construção do protótipo foram selecionados como algoritmos de criptografia assimétrica e Resumo criptográfico o RSASSA-PKCS1-V1.5 e o SHA-384.

Para a realização da requisição a ICP com o envio de arquivo contendo o CSR foi utilizada a biblioteca *axios*. O código desenvolvido para a adição de assinatura *placeholder* e assinatura foi baseado na biblioteca *node-signpdf*, mas devido às peculiaridades como a necessidade de obter o resumo criptográfico dos conteúdos do documento após a adição da assinatura *placeholder* e o fato da biblioteca ter sido feita para execução em servidor não em *browser* foram realizadas alterações nas funções descritas anteriormente para fácil integração.

6.3. Desenvolvimento do Protótipo

Em um primeiro momento foi elaborado a página WEB vista pelo usuário, com possibilidade de ao clicar na área de assinatura ou arrastando um documento para ela fosse possível

[1] O código do protótipo funcional está disponível publicamente em: https://github.com/cavrau/assinador-tcc

ver o mesmo. Após terminar a estrutura visual da página começou então o processo de criação da assinatura. Não foram encontradas bibliotecas que realizem a manipulação de um arquivo PDF utilizando a atualização incremental. Logo se a manipulação do PDF fosse feita utilizando uma biblioteca adicionar uma assinatura a um documento que já contém uma assinatura invalidaria a mesma, pois o conteúdo previamente assinado seria modificado com a adição de mais elementos.

Devido à ausência de bibliotecas foi necessário realizar as atualizações de conteúdos de PDF utilizando código próprio, entendendo a estrutura do documento e a navegando utilizando Expressões Regulares para o processo de adicionar os elementos da assinatura ao documento digital. Foi então elaborada uma rota de API para aplicação para geração do certificado, onde o código rodando no *browser* realiza uma conexão HTTPS com o próprio servidor. Esta rota de API recebe o CSR e passa para o micro serviço de PKI e retorna o conteúdo para aplicação. Após este desenvolvimento foi então feita a parte de conversão e assinatura do documento. Com o fim do desenvolvimento do processo de assinatura tendo os documentos assinados válidos pelo verificador do assinador avançado, foi então integrada uma camada de autenticação para aplicação redirecionando para o provedor de autenticação para realização do *login*, sendo então possível assinar um documento somente se o usuário estiver autenticado.

6.4. Fluxo da aplicação

Após o acesso ao protótipo um usuário veria a Tela Inicial da aplicação, ao clicar no botão de realizar autenticação, o usuário é redirecionado a um provedor de identidade externa. Ao realizar sua autenticação com suas credenciais o usuário é redirecionado para a tela de assinatura, como visível em Figura 4. O usuário é apresentado com uma área onde ele pode arrastar documentos ou clicar para navegar nos arquivos de seu computador. Ao clicar na área indicada ou arrastar um documento PDF até ela a aplicação então muda para a Tela de PDF que apresenta o arquivo escolhido e dá duas opções a de assinatura e a de remoção do arquivo o que voltaria para Tela Inicial, o que é visível em Figura 5.

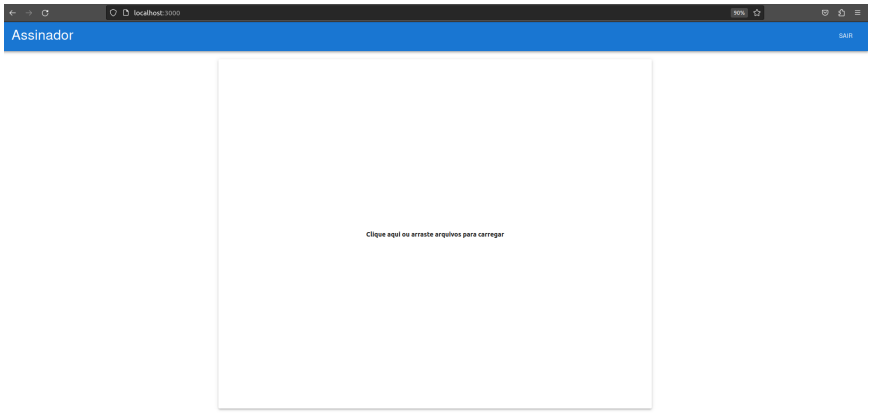

Figura 4. Tela com usuário Autenticado

Ao clicar no botão de assinar todo o pseudo algoritmo descrito é previamente realizado, com a adição dos objetos que representam a assinatura dentro do documento PDF. Após a adição é criado o CSR contendo o resumo criptográfico do documento a ser assinado. É então feita a requisição para o servidor *Next* contendo o CSR e enviando em conjunto o *Token* de autenticação recebido na etapa de autenticação. O servidor então encaminha esta requisição ao microsserviço de PKI do assinador avançado. Ocorre então validação do *Token* de autenticação e emissão de um certificado de uso único pelo microsserviço de PKI. O servidor então recebe o certificado e o passa como resposta para o código executado no *browser* que iniciou a requisição.

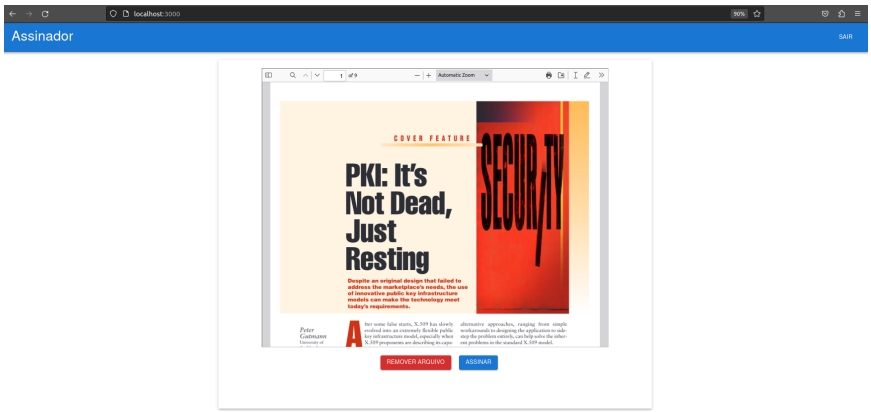

Figura 5. Tela com Documento Carregado

O processo de assinatura então continua, com sua etapa final sendo o download do documento para os arquivos da máquina do usuário. O assinador não possui representação visual da assinatura, logo que a mesma pode ser enganosa, para um usuário que não possuí conhecimento de como as assinaturas em documentos PDF funcionam podendo achar que a representação visual da assinatura no documento é assinatura em si.

6.5. Resultados

O objetivo principal do trabalho pode ser validado realizando a assinatura e verificando se ela é válida, existem diversas ferramentas para a validação de assinaturas e as utilizadas foram a *pdfsig* e o verificador do Assinador Avançado.

Listing 2. Validação no pdfsig

```
pdfsig Downloads/IGI_14_assinado.pdf
Digital Signature Info of: Downloads/IGI_14_assinado.pdf
Signature #1:
  - Signer Certificate Common Name: Gabriel Cabral
  - Signer full Distinguished Name:
  E=gabriel.aristeu.cabral@gmail.com,CN=Gabriel Cabral,C=BR
  - Signing Time: Jun 09 2023 16:25:07
```

```
– Signing Hash Algorithm: SHA-384
– Signature Type: adbe.pkcs7.detached
– Signed Ranges: [0 – 23448], [43450 – 44167]
– Total document signed
– Signature Validation: Signature is Valid.
– Certificate Validation:
Certificate issuer isn't Trusted.
```

O *pdfsig* é um programa de terminal Ubuntu que verifica assinaturas PDF. Foi possível verificar a validade da assinatura usando o programa *pdfsig* como visível no Código 2. Pode-se analisar que a assinatura foi emitida para a pessoa Gabriel Cabral, e também possui informações como o e-mail. Estas informações vêm do provedor de autenticação do assinador avançado e estão ligadas a emissão do certificado no microsserviço de PKI. Pode-se analisar também outras informações como hora da assinatura, função de resumo criptográfico utilizada, assim como o alcance da mesma que neste exemplo encobre todo o documento. Isto também demonstra que programas tradicionais de verificação de assinatura funcionam com a aplicação desenvolvida possuindo uma retro-compatibilidade, única coisa que não é validada é o resumo contido dentro do certificado de uso-único.

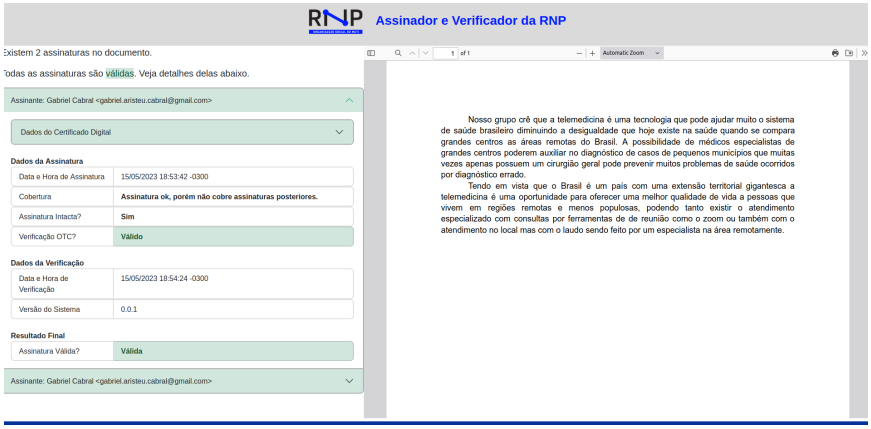

Figura 6. Validação Assinador Avançado

Também foi utilizada a ferramenta de validação de assinaturas do assinador avançado, logo que a mesma contempla também a validação do resumo criptográfico do documento incluído no certificado de uso único em sua criação. O resultado da verificação está disponível em Figura 6. Como pode-se observar neste exemplo com 2 assinaturas ambas estão válidas com a verificação do resumo criptográfico contido dentro do certificado também sendo válida.

7. Conclusão e Trabalhos Futuros

Tendo em vista os objetivos propostos na introdução, o trabalho desenvolvido e os experimentos realizados, pode-se afirmar que o trabalho cumpre em grande parte os objetivos

propostos. Foi desenvolvida uma aplicação que utilize certificados de uso único, gerados por uma AC, para assinatura de documentos digitais, foi confirmada a compatibilidade de verificação de assinatura tanto com validadores que levam em conta certificados de uso único como programas padrão. Garantiu-se também que a assinatura se deu no local de execução da aplicação e que o documento não foi enviado a nenhum servidor, com o descarte do certificado e chaves após o uso, garantindo também a privacidade do documento assinado.

Este trabalho abre uma frente para diversos trabalhos futuros, que utilizam a assinatura de documentos digitais no cliente de execução, mas uma melhoria direta seria a elaboração de um sistema de gerenciamento de assinaturas, tendo a possibilidade de convidar pessoas a assinarem um mesmo documento e ter o controle de quais usuários requisitados já assinaram ou não o documento.

Referências

[ISO 2020] (2020). Document management — portable document format — part 2: Pdf 2.0. Standard, International Organization for Standardization, Geneva, CH.

[Council 1991] Council, N. R. (1991). *Computers at Risk: Safe Computing in the Information Age*. The National Academies Press, Washington, DC.

[Diffie and Hellman 1976] Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654.

[Gutmann 2002] Gutmann, P. (2002). Pki: it's not dead, just resting. *Computer*, 35(8):41–49.

[Kohnfelder 1978] Kohnfelder, L. (1978). Towards a practical public-key cryptosystem.

[Mayr et al. 2023] Mayr, L., Zambonin, G., Schardong, F., and Custódio, R. (2023). One-time certificates for reliable, inclusive and secure document signing. *ESORICS*.

[Myers et al. 1999] Myers, M., Adams, C., Verisign, Solo, D., Kemp, D., Citicorp, and Techonologies, E. (1999). Internet X.509 Certificate Request Message Format. RFC 2511.

[Stallings 2013] Stallings, W. (2013). *Cryptography and Network Security: Principles and Practice*. Prentice Hall Press, USA, 6th edition.

[Subramanya and Yi 2006] Subramanya, S. and Yi, B. (2006). Digital signatures. *IEEE Potentials*, 25(2):5–8.

[zu Selhausen 2018] zu Selhausen, K. M. (2018). Security of PDF Signatures.