eWebAPI: uma API para assinar digitalmente lotes de certificados eletrônicos utilizando o e-certsDS

Alan Miguel Dorr Schulze[1], **Diego Kreutz**[1]

[1]Ciência da Computação (CC)
Programa de Pós-graduação em Engenharia de Software (PPGES)
Universidade Federal do Pampa (UNIPAMPA)

{alanschulze.aluno,diegokreutz}@unipampa.edu.br

Abstract. *e-certsDS is a tool for issuing and publishing electronic certificates using OpenPGP digital signatures. It works on the command line and presents some challenges in terms of usability and complexity. To simplify use and allow integration with other systems, we propose eWebAPI, an API to use e-certsDS as a service, eliminating complex installation steps and parameterized execution via the command line. The eWebAPI provides a list of endpoints that allows remote use of the service and integration with other Web systems (e.g., event and project management systems).*

Resumo. *A e-certsDS é uma ferramenta de emissão e publicação de certificados eletrônicos que utiliza assinaturas digitais OpenPGP. Ela funciona em linha de comando e apresenta alguns desafios em termos de usabilidade e complexidade. Para simplificar a utilização e permitir a integração com outros sistemas, propomos a eWebAPI, uma API para utilizar o e-certsDS como um serviço, eliminando etapas complexas de instalação e execução parametrizada via linha de comando. A eWebAPI fornece uma relação de endpoints que permitem a utilização remota do serviço e a integração com outros sistemas Web (e.g., sistemas de gestão de eventos e de projetos).*

1. Introdução

Atualmente, existem diversos softwares destinados à gestão de certificados eletrônicos, que desempenham um papel essencial em eventos, atividades acadêmicas e outras situações que demandam a emissão de múltiplos certificados eletrônicos. A principal função desses sistemas é gerar, emitir e validar certificados eletrônicos, como certificados de participação em eventos. Soluções como Certifier [Certifier, 2023], Gerador de Certificados [Gerador de Certificados, 2023], Gerar Certificado [Matiê, 2023], Doity [Doity, 2022] e SGCE [DTIC, 2014] são pagas ou limitam o número de certificados que podem ser emitidos, ou não utilizam assinaturas digitais, ou ainda carecem de APIs de utilização e integração.

A ferramenta e-certsDS [Uri et al., 2021] surgiu para resolver prioritariamente dois problemas. Primeiro, permitir a utilização de assinaturas digitais como forma de verificação da autenticidade dos certificados eletrônicos emitidos. Segundo, possibilitar a emissão e publicação, em lote, de quantidades quaisquer de certificados eletrônicos (e.g., 2.000 certificados de participação) sem qualquer custo para as instituições. O e-certsDS

tem sido utilizado na prática para emitir várias centenas de certificados e atestados anuais do Programa Clube Universidade Hacker (UniHacker.Club)[1].

O principal problema do e-certsDS é a baixa usabilidade e elevada complexidade. Por exemplo, o sistema utiliza diversas ferramentas e bibliotecas, sendo que algumas delas podem requerer conhecimentos específicos ou apresentar problemas de compatibilidade para o usuário (e.g., ser incompatíveis com versões instaladas no sistema do usuário). Adicionalmente, o e-certsDS funciona apenas em sistemas GNU/Linux e via linha de comando, exigindo ainda o conhecimento de diversos parâmetros de entrada. Por fim, não há uma forma simples e intuitiva de o usuário localizar, diagnosticar e corrigir eventuais problemas de execução, que podem ocorrer em diferentes sub-sistemas e bibliotecas utilizadas na implementação. Resumidamente, o e-certsDS é uma ferramenta projetada originalmente para ser utilizada por especialistas em sistemas GNU/Linux e linha de comando.

O principal objetivo deste trabalho é propor e implementar uma API, denominada eWebAPI, para oferecer o e-certsDS como um serviço. Acreditamos que a API irá simplificar a utilização da ferramenta e também permitir a integração com sistemas existentes nas instituições, como sistemas de gerenciamento de eventos e sistemas de gestão de projetos. Um segundo objetivo é atualizar a ferramenta e-certsDS para permitir a utilização de certificados digitais PKCS#{11,12} da infraestrutura de chaves públicas ICPEdu[2].

Como principais contribuições técnicas do trabalho podemos destacar: (a) atualização da ferramenta e-certsDS para utilização de certificados digitais PKCS#{11,12} ICPEdu ou ICPBrasil; (b) projeto da eWebAPI para disponibilização e utilização da e-certsDS como um serviço; (c) implementação de uma versão operacional da API e instanciação do e-certsDS como um serviço.

As próximas seções apresentam a proposta e implementação da eWebAPI. Por fim, apresentamos também algumas considerações finais e trabalhos futuros.

2. eWebAPI

A Figura 1 representa a visão geral da solução proposta para disponibilizar a ferramenta e-certsDS como um serviço. Resumidamente, a eWebAPI é composta por quatro módulos (**Usuários**, **Templates**, **Validação** e **Execução**) e seis *endpoints* associados aos módulos. A solução utiliza ainda um serviço externo (da ICPEdu) para validação de assinaturas e uma versão modificada da ferramenta e-certsDS. Em síntese, o usuário utiliza os *endpoints* da API para emitir um ou mais certificados eletrônicos de forma simples e automatizada.

A eWebAPI e a integração com a e-certsDS consideram um ambiente organizacional típico de uma universidade, onde há unidades como pró-reitorias, departamentos, campus e cursos. Assumimos que o administrador do sistema é responsável por adicionar e disponibilizar os certificados digitais PKCS#{11,12}, pessoais ou organizacionais, para a assinatura digital dos certificados eletrônicos. Os usuários, como o pessoal técnico das secretarias das unidades organizacionais, pode ser registrado como autorizado a emitir certificados eletrônicos utilizando os certificados digitais disponíveis no serviço.

[1] `https://unihacker-club.github.io`
[2] `https://pessoal.icpedu.rnp.br`

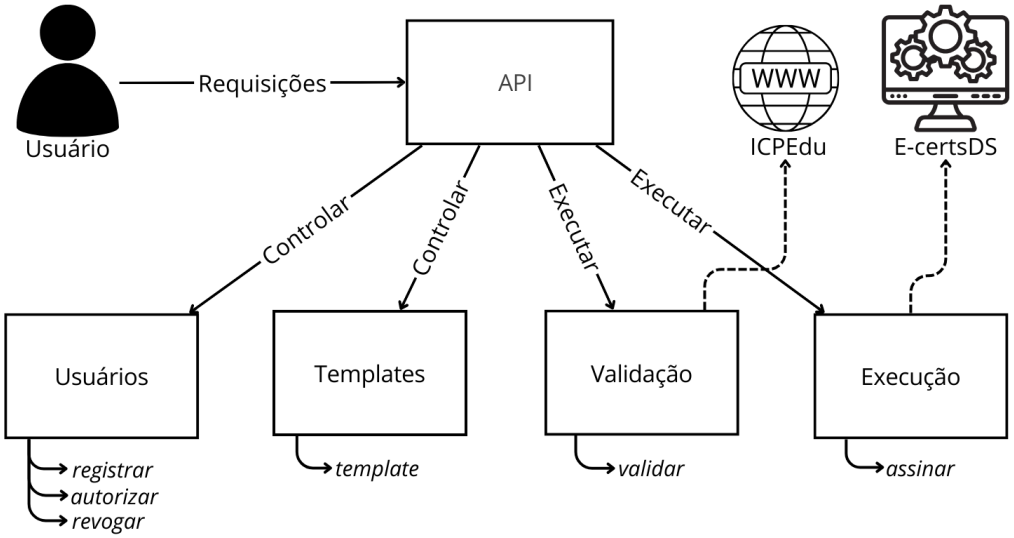

Figura 1. Visão geral da solução (API e e-certsDS)

A API é constituída pelos *endpoints* discriminados a seguir. Cada *endpoint* disponibiliza um recurso diferente para os usuários do serviço. Assumimos que os *endpoints* da API serão utilizados sobre um canal de comunicação seguro, como um canal TLS 1.3 com autenticação mútua entre o cliente e servidor.

registrar : permite aos proprietários dos certificados digitais disponíveis no serviço registrar e autorizar pessoas a emitir certificados eletrônicos. O registro ocorre através do envio da chave pública da pessoa autorizada (e.g., chave pública do certificado digital do Fulano de Tal). Parâmetros do *endpoint*: (1) *token* válido; e (2) dados (i.e., nome, email) e chave pública da pessoa que a ser adicionada à lista de autorizados.

revogar : permite aos proprietários dos certificados digitais disponíveis no serviço revogar registros previamente autorizados. Parâmetros do *endpoint*: (1) *token* válido; e (2) identificador da chave pública a ser revogada.

autorizar : permite a qualquer usuário registrado no sistema solicitar um *token* de autorização de geração de certificados eletrônicos. Parâmetros do *endpoint*: (1) identificador da chave pública; e (2) *nonce* assinado utilizando a chave privada do certificado digital.

assinar : em posse de um *token* válido, o usuário pode requisitar a emissão de um ou mais certificados eletrônicos para participantes de eventos ou projetos, por exemplo. Parâmetros do *endpoint*: (1) *token* válido; (2) lista de dados dos participantes (i.e., nome, email, tipo e horas - vide e-certsDS [Uri et al., 2021]); (3) informações sobre o evento ou atividade; e (4) identificador do template a ser utilizado na geração dos certificados.

template : usuários registrados podem também disponibilizar novos templates para emissão de certificados. Cada template submetido é validado pelo sistema de geração de certificados, retornando sucesso ou erro na adição do template ao repositório. Parâmetros do *endpoint*: (1) *token* válido; e (2) dados do novo template.

validar : é o único *endpoint* que não necessita de autorização prévia de utilização. O *endpoint* serve para validar um certificado emitido (i.e., arquivo PDF). Para validar, a API verifica se a assinatura é criptograficamente válida e se o certificado

digital utilizado para gerar a assinatura digital não foi revogado na autoridade certificadora da ICPEdu, por exemplo.

3. Implementação

Um protótipo da API foi desenvolvido utilizando a linguagem `JavaScript` e `Node.JS` (versão 18.16.1 LTS), que é um ambiente de execução `JavaScript` comumentemente utilizado por desenvolvedores de software. Para criar a API utilizamos o `Express` [Foundation OpenJS, 2023], um *framework* minimalista que permite criar rotas, *middlewares*, utilização de *cookies*, processamento e envio de cabeçalhos, requisições, entre outras funcionalidades úteis.

Uma das medidas de segurança que a implementação incorpora é a utilização de rotas protegidas para chamadas ao e-certsDS, isto é, apenas usuários autorizados poderão realizar a invocação da ferramenta. Esta proteção é implementada com o auxílio de *tokens* do tipo JWT (JSON *Web Token*) utilizando a biblioteca `jsonwebtoken` (versão 9.0.0). O *token* JWT é uma *string* que pode conter informações sobre o usuário em um formato JSON [Bray, 2017], dados sobre o tempo de validade do *token* e uma assinatura do servidor utilizando primitivas criptográficas como HMAC (*Hash-Based Message Authentication Codes*) [Krawczyk et al., 1997], SHA256[3] e uma chave privada. Resumidamente, o servidor aceitará somente *tokens* válidos nas requisições da API. O *token* gerado pelo servidor deverá ser incluído no cabeçalho HTTP (campo *Authorization*) de cada requisição enviada ao servidor.

O fluxo de operação do *endpoint* `assinar` é ilustrado na Figura 2. Ao receber a requisição, a API valida o *token* e cria os arquivos temporários contendo os dados dos participantes, para os quais os certificados serão emitidos, e as informações do template a ser utilizado na geração. Em seguida, é criada uma instância de execução do e-certsDS, para geração e publicação (ou envio por e-mail) dos certificados. A execução da ferramenta retornará sucesso ou erro. Os códigos de erro são acompanhados de detalhes que permitem ao usuário identificar a falha que ocorreu (e.g., indisponibilidade ou revogação do certificado digital escolhido).

Além da implementação do primeiro protótipo da API, adaptamos a ferramenta e-certsDS para viabilizar a utilização de assinaturas digitais com certificados PKCS. Adicionalmente, resolvemos adicionar à solução também um sub-sistema de verificação de validade dos documentos digitalmente assinados. É importante destacarmos que ferramentas de assinatura digital de PDF convencionais, não costumam incorporar métodos de validação das assinaturas digitais que verificam a revogação, ou não, dos certificados digitais na autoridade certificadora emissora [Oliveira et al., 2021].

3.1. Assinaturas digitais com certificados padrão PKCS

A primeira mudança proposta na e-certsDS foi a substituição do método de assinatura digital. Optamos por substituir o método GnuPG por assinaturas digitais embutidas nos arquivos PDF através do padrão PAdES (PDF *Advanced Electronic Signatures*). O PAdES é um padrão aceito pela ICP-Brasil, válido como prova jurídica, que permite também uma representação visual da assinatura no documento PDF. Para fins de teste e validação da

[3] `https://csrc.nist.gov/projects/hash-functions`

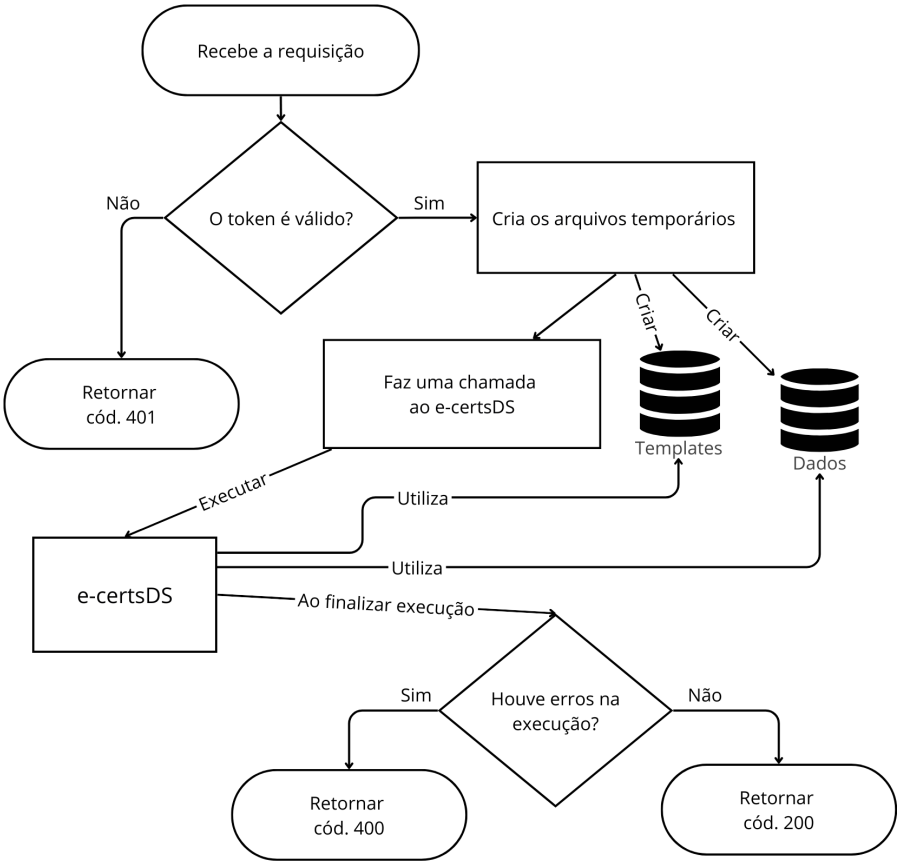

Figura 2. Fluxo de execução do *endpoint* `assinar`

implementação, utilizamos certificados digitais emitidos pela Autoridade de Certificação (AC) da ICPEdu, gerenciada pela RNP[4].

Para produzir as assinaturas no padrão PAdES, escolhemos o software pyHanko[5], desenvolvido em Python e de código aberto. O pyHanko permite adicionar estampas visuais e assinar digitalmente documentos PDF. É importante destacarmos também que ele permite a inserção de múltiplas assinaturas digitais no PDF.

Para que o gerador do e-certsDS pudesse assinar arquivos PDF utilizando o pyHanko, foi necessário a adição de um arquivo de configuração para descrever a estampa visual da assinatura e indicar o arquivo do certificado digital padrão PKCS. Além disso, o gerador necessita receber como parâmetro a senha do certificado digital que será utilizado para assinar os documentos PDF.

3.2. Verificação de validade das assinaturas digitais

Outro recurso importante é a verificação de validade da assinatura digital. Para implementar este recurso, utilizamos a ferramenta de verificação de assinatura de documentos disponível online no site da ICPEdu[6]. Incorporamos ao e-certsDS um *script* que recebe como parâmetro um arquivo PDF digitalmente assinado e envia ele para o site de verificação da

[4]`https://www.rnp.br`
[5]`https://github.com/MatthiasValvekens/pyHanko`
[6]`https://pessoal.icpedu.rnp.br/public/verificar-assinatura`

ICPEdu. A resposta HTML do site é processada para identificar se a assinatura é válida ou não. É importante destacarmos que esta verificação funciona apenas para documentos assinados com certificados digitais ICPEdu. Para certificados emitidos por outras autoridades certificadores (ACs), será necessário incorporar recursos adicionais, como um sub-sistema de verificação de listas de certificados revogados das ACs.

4. Considerações Finais

Os testes iniciais indicam que a eWebAPI consegue atingir o seu principal objetivo, ou seja, simplificar a utilização da e-certsDS, agora oferecida como um serviço. Adicionalmente, a API viabiliza a integração com sistemas existentes, como os utilizados para gerenciar projetos ou eventos. Acreditamos que futuramente a solução proposta será capaz de substituir soluções similares existentes, como a SGCE, que não possui suporte a assinaturas digitais.

Como trabalhos futuros podemos destacar: (a) evolução da API, incluindo novos *endpoints* que facilitam a integração com sistemas específicos; (b) integração com sistemas de gerenciamento de projetos existentes, como o SAP[7]; (c) análise de segurança da API e demais componentes da solução; (d) utilização de segurança assistida por hardware [Coppolino et al., 2019] para proteger em tempo de armazenamento e execução os certificados digitais disponíveis no *back-end* da solução; (e) testes de usabilidade e integração com grupos de usuários diversos.

Referências

Bray, T. (2017). The JavaScript Object Notation (JSON) Data Interchange Format. RFC 8259.

Certifier (2023). Certifier.io. `https://certifier.io/`.

Coppolino, L., D'Antonio, S., Mazzeo, G., and Romano, L. (2019). A comprehensive survey of hardware-assisted security: From the edge to the cloud. *Internet of Things*.

Doity (2022). Doity. `https://doity.com.br`.

DTIC (2014). SGCE. `https://softwarepublico.gov.br/social/sgce`.

Foundation OpenJS (2023). Express: Biblioteca para desenvolvimento de API's. Versão 4.18.2. `https://expressjs.com/pt-br/`.

Gerador de Certificados (2023). Gerador de certificados. `https://geradordecertificados.com`.

Krawczyk, D. H., Bellare, M., and Canetti, R. (1997). HMAC: Keyed-Hashing for Message Authentication. RFC 2104.

Matiê (2023). Gerar certificados. `https://gerarcertificado.com.br`.

Oliveira, S., Vargas, L., Mansilha, R., and Kreutz, D. (2021). Usabilidade de ferramentas para assinatura digital de documentos pdf. In *Anais da XIX Escola Regional de Redes de Computadores*, pages 79–84, Porto Alegre, RS, Brasil. SBC.

Uri, M., Vargas, L., and Kreutz, D. (2021). e-certsDS: Certificados eletrônicos com assinatura digital. In *Anais Estendidos do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 66–73.

[7] `https://sites.unipampa.edu.br/atendimento/manuais/sap/`