Firmas solidarias: Tecnología criptográfica para los derechos humanos

Solidarity Signatures: Cryptographic Technology for Human Rights

Erick Isaac Lascano Otañez - A00836571
Pedro Soto Juárez - A00837560
Luis Fernando Alcázar Díaz - A00836287
Leonardo De Regil Cárdenas - A00837118
Alexei Carrillo Acosta - A01285424

Profesor Encargado: Luis Miguel Méndez Díaz

16 de marzo de 2025

Índice

1. Resumen

El proyecto “Firmas solidarias: Tecnología criptográfica para los derechos humanos” busca desarrollar una solución tecnológica para que Casa Monarca firme digitalmente documentos y proteja su acceso con autenticación segura. Se propone crear una aplicación en Python integrada con Microsoft PowerApps para gestionar firmas digitales y autenticación de usuarios mediante Microsoft Entra ID. Se usarán bibliotecas criptográficas como PyCryptodome (RSA y ECDSA) y se considerarán opciones económicas de almacenamiento en la nube (AWS S3, Google Cloud Storage) y servidores locales. El objetivo es ofrecer a Casa Monarca una herramienta accesible que garantice autenticidad, integridad y confidencialidad de documentos, protegiendo los derechos humanos de las personas atendidas.

The project “Solidarity Signatures: Cryptographic Technology for Human Rights” aims to develop a technological solution to enable Casa Monarca to digitally sign documents and protect their access through secure authentication. The proposal involves creating a Python-based application integrated with Microsoft PowerApps to manage digital signatures and user authentication using Microsoft Entra ID. Cryptographic libraries such as PyCryptodome (RSA and ECDSA) will be utilized, and cost-effective cloud storage options (AWS S3, Google Cloud Storage) as well as local servers will be considered. The goal is to provide Casa Monarca with an accessible tool that ensures the authenticity, integrity, and confidentiality of documents, safeguarding the human rights of the individuals they serve.

2. Introducción

Garantizar la autenticidad, integridad y disponibilidad de documentos digitales es fundamental para cualquier organización. Casa Monarca, es una organización que acoge, protege, promueve e integra a migrantes internos, deportados, con necesidad de protección internacional en tránsito con situación migratoria irregular (CasaMonarca, 2023), la cual necesita una solución tecnológica que permita firmar digitalmente documentos, así como permitir que solo personal autorizado pueda acceder a ellos, sin que puedan ser alterados.

Para abordar este reto, se propone el desarrollo de una aplicación que permita asignar firmas digitales a documentos, además de establecer mecanismos de autenticación para proteger el acceso y contenido de los mismos. Una firma digital es un sello de autenticación electrónico cifrado en información digital, como documentos electrónicos. La firma constata que la información proviene del firmante y que no ha sido modificada (Microsoft, 2025).

Se busca implementar este sistema mediante un script base en Python que gestione tanto la firma digital como la autenticación de los documentos. La solución se integrará en un entorno de Microsoft PowerApps, permitiendo una visualización clara y facilitando su adopción dentro del entorno de la organización. Además, se analizarán los costos asociados a su implementación, incluyendo la viabilidad de utilizar un sistema en la nube.

Sin duda, el desarrollo de este proyecto representa un desafío técnico, ético y social, al crear una solución que permita firmar digitalmente la información de personas en condiciones de vulnerabilidad, garantizando su autenticidad y asegurando su derecho a la privacidad y acceso a la tecnología.

3. Marco referencial

3.1. Revisión de soluciones existentes.

OpenXPKI es una plataforma de código abierto enfocada en la gestión de infraestructura de clave pública y certificados digitales. Se distingue por su capacidad de automatizar la emisión, renovación y revocación de certificados para simplificar la administración de identidades

digitales. Es compatible con protocolos estándar como SCEP, EST y ACME, además de contar con una API REST. Emplea OpenSSL, lo que garantiza un almacenamiento seguro. Tiene una interfaz web basada en Ember.js que facilita su uso. Además, cuenta con sistemas de autenticación como LDAP, OAuth y SAML, lo que lo convierte en una opción para organizaciones que buscan mejorar su seguridad y confianza en la documentación digital.

OpenXPKI es una solución robusta y flexible que permite la administración eficiente de firmas digitales, garantizando la autenticidad e integridad de los documentos críticos.(OpenXPKI Project, 2024)

SIGNply es una solución de firma digital diseñada para dispositivos móviles, permitiendo la autenticación segura de documentos por medio de firma biométrica. Cumple con la normativa europea de eIDAS, así como con el ISO 19794/7, lo que garantiza la legalidad y seguridad de los documentos firmados. Una de sus principales ventajas es la incorporación de biometría avanzada, ya que registra parámetros como presión, aceleración y velocidad del trazo de la firma, lo que lo hace infalible ante falsificaciones. (SIGNply, 2024)

PyCryptodome es una biblioteca de Python diseñada para ofrecer funciones criptográficas. Genera y gestiona claves criptográficas esenciales para autenticar documentos y usa algoritmos de cifrado asimétrico como RSA y DSA. Esta biblioteca crea pares de claves públicas y privadas, lo que garantiza que los documentos firmados digitalmente no puedan ser alterados.

Usa hash criptográfico, que permite generar resúmenes de datos únicos y verificar la integridad de documentos firmados. PyCryptodome es una solución clave para la implementación de firmas digitales en Python, lo cual puede ser pieza clave en la elaboración del reto.(GeeksforGeeks, 2024)

3.2. Revisión de estándares existentes.

Dentro de la protección de documentos y la autenticación de firmas digitales existen varios estándares, aqui mostramos los principales: Las firmas digitales están reguladas por normativas internacionales que aseguran su validez legal y técnica. Estas normativas establecen requisitos sobre la autenticación, integridad y verificabilidad de documentos firmados electrónicamente. Aqui están las principales normativas:

- ETSI EN 319 411

ETSI EN 319 411 regula los requisitos para las Autoridades de Certificación (CAs), estableciendo cómo deben operar para garantizar la confiabilidad de los certificados.

- ETSI EN 319 412

ETSI EN 319 412 define los formatos específicos de certificados electrónicos utilizados en firmas digitales avanzadas y cualificadas.

- XAdES (XML Advanced Electronic Signatures) – ETSI TS 101 903

Permite firmar documentos XML con distintos niveles de seguridad, asegurando autenticidad e integridad, incluyendo marcas de tiempo para verificación a largo plazo.

Tambien por otro lado, es fundamental poder asegurar que solo usuarios autorizados puedan acceder a los documentos. Para esto se emplean protocolos de autenticación y control de acceso como:

- LDAP (Lightweight Directory Access Protocol)

LDAP es un protocolo estándar para la gestión de identidades y autenticación en sistemas de gestión documental empresariales.

- SAML (Security Assertion Markup Language)

Permite la autenticación federada entre distintas organizaciones, facilitando el acceso seguro a documentos digitales.

3.3. Uso de recursos disponibles:

Se planea desarrollar un sistema basado en Python para gestionar la firma digital y la autenticación de documentos, integrado en Microsoft PowerApps para facilitar su adopción en la organización. Se considerará la infraestructura tecnológica existente para maximizar la eficiencia y minimizar costos. Además, se evaluarán los recursos necesarios, como bases de datos, almacenamiento en la nube y dispositivos tecnológicos disponibles, así como los costos asociados a licencias, distribución y operación.

Fase 1: Desarrollo de Infraestructura de Almacenamiento y Procesamiento

- Base de Datos

Para la gestión de documentos firmados y metadatos, se recomienda utilizar bases de datos relacionales como PostgreSQL o MySQL, los cuales se especializan en datos estructurados y consultas complejas. En dado caso que se requiere una estructura más flexible, se puede optar por bases de datos NoSQL como MongoDB, que ofrecen mayor adaptabilidad en la estructura de los datos.

- Almacenamiento de Respaldo

Para prevenir la pérdida de datos, se implementa un sistema de almacenamiento de respaldo. Las opciones más económicas incluyen el uso de AWS S3, que ofrece un costo que ronda entre 0.02205 y 0.02415 USD por GB almacenado. También se puede considerar Google Cloud Storage, específicamente el Archive Storage, cuyo costo es de 0.0027 USD por GB al mes, por la cual se peude optar para almacenamiento a largo plazo. Como alternativa, se pueden realizar copias físicas utilizando memorias USB o discos duros externos para tener un respaldo offline disponible.

- Servidor Offline (Alternativa económica)

Como alternativa económica al uso de almacenamiento en la nube, se puede implementar un servidor offline utilizando un equipo reacondicionado, como un Lenovo ThinkCenter o similar, configurado con RAID para mayor seguridad. La inversión inicial estimada para este servidor es de entre 500 y 1,000 USD, mientras que cada disco duro de 2 a 4 TB tiene un costo de entre 100 y 200 USD. Es imporante tomar tambien en cuenta los costos operativos anuales incluyen el consumo eléctrico, estimado entre 120 y 180 USD, y el mantenimiento, con un costo aproximado de 20 a 50 USD anuales. Considerando una amortización de la inversión inicial en 5 años, el costo total anual se estima entre 150 y 250 USD.

Fase 2: Seguridad y Autenticación

- Gestión de Claves Criptográficas

Se recomiendan dos opciones principales. La primera es AWS KMS, que ofrece un nivel gratuito y costos adicionales economicos, con un precio de 0.03 USD por cada 10,000 solicitudes. La segunda opción es HashiCorp Vault, el cual es conveniente para ambientes de desarrollo, con un costo aproximado de 0.62 USD por clúster por hora.

- Bibliotecas de Firma Digital en Python

Respecto a la implementación de firmas digitales utilizando Python, se consideran bibliotecas eficientes como extttcryptography, que soporta algoritmos RSA y ECDSA; extttpyHanko, especializada en la firma de documentos PDF; y extttsignxml, ideal para la firma de documentos XML.

- Autenticación de Usuarios

Para la autenticación de usuarios, se recomienda la integración con Microsoft Entra ID, la cual es gratuita si Casa Monarca ya cuenta con licencias de Microsoft 365. Además, se puede implementar autenticación multifactor (MFA) utilizando herramientas como Google Authenticator o Microsoft Authenticator para mejorar la seguridad.

Fase 3: Desarrollo e Integración

- Lenguaje de Programación

Se utilizará Python como el principal lenguaje de desarrollo debido a su versatilidad, amplia disponibilidad de bibliotecas de criptografía y autenticación, y facilidad de integración con otros sistemas.

- Integración con PowerApps

El uso de Microsoft PowerApps permitirá diseñar y desplegar la interfaz de usuario, asegurando una integración nativa con Microsoft Entra ID para la autenticación de usuarios. Este enfoque también facilita la adopción de la solución dentro de la organización.

- Fase 4: Hardware y Dispositivos

Se emplearán computadoras con acceso a PowerApps y a sistemas de autenticación para desarrollar e implementar la solución. Según las necesidades del proyecto, se considerará el uso de servidores locales o máquinas virtuales para garantizar un procesamiento eficiente.

Fase 5: Costos y Licencias

- Costo de Almacenamiento

Las opciones de almacenamiento más rentables incluyen AWS S3 con un costo entre 0.02205 y 0.02415 USD por GB y Google Cloud Storage (Archive Storage) con un costo de aproximadamente 0.0027 USD por GB al mes.

- Licencias para PowerApps

Respecto a licencias, se puede optar por el plan gratuito para desarrolladores o adquirir PowerApps Premium, cuyo costo varía entre 12 y 20 USD por usuario al mes, dependiendo del número de usuarios.

3.4. Bibliotecas disponibles para implementar esquemas de clave pública

Existen diferentes bibliotecas que facilitan la implementacion de esquemas de clave publica en lenguajes como Java y Python, cada una con caracteristicas especificas que se adaptan a diferentes necesidades. A continuacion se presentaran las mas relevantes de ambos lenguajes, dando un mayor enfoque a las bibliotecas de Python.

En Java, la biblioteca Bouncy Castle, es reconocida por su flexibilidad y amplia cobertura de algoritmos criptograficos como RSA, ECC y DSA. Esta biblioteca se integra perfectamente con la arquitectura de seguridad de Java, conocida como Java Cryptography Architecture (JCA), lo que permite desarrollar soluciones robustas y seguras. Ademas, proporciona herrameintas avanzadas para la generacion de certificados, firmas digitales y manejo de claves. (Bouncy Castle, nd)

Mientras que para Python, la biblioteca PyCryptodome se presenta como una opcion poderosa y versatil. Esta ofrece una amplia variedad de algoritmos de clave publica, incluyendo RSA, DSA y ECC, facilitando tanto el cifrado de datos como la implementacion de firmas digitales. Su sintaxis clara y su integracion sencilla con proyectos Python la hacen especialmente util para proteger datos confidenciales (PyCryptodome, nd). Aunque PyCryptodome no cuenta con soporte nativo directo para entornos de ejecuciuon seguros, puede complementarse con herramientas adicionales como tpm2-pytss, una biblioteca que permite interactuar con modulos TPM 2.0, ideal para almacenar claves de forma segura (tpm2-pytss, nd). Asimismo, es posible integrar PyCryptodome con el Intel SGX SDK mediante bindings en Python, logrando asi una proteccion avanzada mediante enclaves seguros. (Intel, nd)

En el caso del desarrollo de aplicaciones que integren Intel SGX, tanto el Intel SGX SDK como el Open Enclave SDK proporcionan un entorno robusto para implementar enclaves seguros, protegiendo asi datos sensibles incluso en entornos potencialmente comprometidos. Ambas APIs pueden ser utilizadas en proyectos escritos en Python mediante bindings especializados que permiten interactuar con estas herramientas de forma eficiente. El Intel SGX SDK se integra mediante extensiones en Python que permiten invocar enclaves seguros desde codigo Python, mientras que el Open Enclave SDK proporciona una API unificada que facilita el desarrollo de aplicaciones que protejan datos criticos durante su procesamiento, ya que sea en Intel SGX, ARM TrustZone u otros entornos seguros. Esto resulta especialmente util para porteger datos cinfidenciales en sistemas distribuidos o en la nube (Microsoft, nd).

3.5. ¿Cuáles son los esquemas criptográficos más adecuados?

.

Los esquemas criptográficos más adecuados para el reto deben ser aquellos que garanticen la integridad, autenticidad, confidencialidad y el no repudio en los documentos firmados digitalmente, mientras también se cumplen con los requisitos legales necesarios y que se adapte a los sistemas operacionales de la organización socio formadora.

Por ejemplo, se puede considerar el algoritmo RSA, que es un sistema criptográfico de clave pública que utiliza un par de claves para asegurar las comunicaciones digitales y las transacciones a través de redes no seguras, como Internet. La criptografía de clave pública, también conocida como criptografía asimétrica, emplea dos claves distintas pero matemáticamente relacionadas: una pública y una privada. La clave pública puede compartirse con cualquier persona, mientras que la clave privada debe mantenerse en secreto (Yasar and Cobb, 2025).

Por otro lado, el ECDSA(Elliptic Curve Digital Signature Algorithm) se basa en la criptografía de curvas elípticas (ECC) y se utiliza para generar claves, autenticar, firmar y verificar mensajes. Es el algoritmo criptográfico empleado para crear claves y generar y verificar firmas utilizadas en la autenticación (Nightingale, 2024). Sin duda, ofrece una alternativa al RSA, aunque su implementación puede ser más técnica.

Además, es relevante explorar opciones que involucren la tecnología Blockchain y las firmas

digitales a través de Ethereum, que utiliza el ya mencionado ECDSA. Este algoritmo no solo se utiliza para transacciones dentro de la blockchain, sino también en la firma digital de documentos. Lo interesante de las firmas en Ethereum es su integración con contratos inteligentes (smart contracts), que permiten verificar automáticamente las firmas ECDSA a través de Solidity. Esta funcionalidad no solo es clave para enviar transacciones, sino también para autenticar documentos de manera segura y eficiente. A diferencia de RSA, que es más lento, ECDSA ofrece mayor eficiencia y seguridad, especialmente en el contexto de blockchain (Zuidhoorn, 2025).

4. Referencias

Bouncy Castle (n.d.). The Legion of the Bouncy Castle Java Cryptography APIs.

CasaMonarca (2023). Misión - Casa Monarca.

GeeksforGeeks (2024). What is pycryptodome in python?

Intel (n.d.). Intel® Software Guard Extensions (Intel® SGX) SDK Documentation.

Microsoft (2025). Firmas digitales y certificados - Soporte técnico de Microsoft.

Microsoft (n.d.). Open Enclave SDK Documentation.

Nightingale, C. (2024). What is Elliptic Curve Digital Signature Algorithm? - ECDSA.

OpenXPKI Project (2024). OpenXPKI - Enterprise-Grade PKI Software.

PyCryptodome (n.d.). PyCryptodome Documentation.

SIGNply (2024). SIGNply: Firma Digital para Empresas.

tpm2-pytss (n.d.). TPM 2.0 Python TSS Documentation.

Yasar, K. and Cobb, M. (2025). What is the RSA algorithm?

Zuidhoorn, M. (2025). The Magic of Digital Signatures on Ethereum.