Eidas Sign API

Table of Contents

Chapter 1. Overview

The eIdas signature API is a PoC API that allows you to sign objects using X509 / eidas compliant signatures.

1.1. Version information

Version : 0.1

1.2. Contact information

Contact : Sphereon
Contact Email : dev@sphereon.com

1.3. License information

License : Apache License Version 2.0
License URL : https://www.apache.org/licenses/LICENSE-2.0
Terms of service : https://docs.sphereon.com/api/eidas-sign-poc/0.1/html

1.4. URI scheme

Host : gw.api.cloud.sphereon.com
BasePath : /crypto/keys/0.9
Schemes : HTTPS

1.5. Tags

- Certificates : Certificates and signing
- Signatures : Signature verification
- eidas-controller : Eidas Controller

Chapter 2. Introduction

The eIdas Sign API is a Proof of Concept to sign input data using X509 certificates using a Cades signature.

Warning:
This is Proof of Concept code. Sphereon is building an Open-Source and Commercial product based upon the lessons learned. We provide this code for people that are interested in experimenting or learning from it. Do not use in production settings!

This API typically will be used with the eIDAS-VC-Bridge-POC, which allows you to sign Verifiable Credentials using eIdas compliant Cades signatures with the help of this API.

2.1. API - Swagger docs

This document can be found whilst accessing the URL at which the application runs and supplying the path /docs. By default, this will be http://localhost:21762/docs

We suggest reading the documentation from there, as it will include request/responses and links to all the class documentation.

The OpenAPI/Swagger JSON document can be found by using the path /v2/api-docs, by default this will be http://localhost:21762/v2/api-docs

2.2. Relation to crypto-keys API - Keys, Secret and Certificates

This API is seperated from our bigger Crypto-Keys-API, which enables you to store and use cryptographic keys with support for multiple key types and algorithms. It allows you to remotely and securely store secrets in Hardware Security Modules. Certificates are built on top of these keys and secrets. They are the combination of public and private keys with a secret. The crypto keys allows you to encrypt, decrypt, sign and verify data, as well as store protected keys and certificates.

Chapter 3. SDKs

Our API's are based on the OpenAPI specification (formerly known as Swagger specification). This means you can pickup our REST API definition file and generate classes for your favorite programming language.

3.1. Provided SDKs and generated SDKs

3.1.1. Provided SDKs

Since this is a PoC, Sphereon does not provide supported SDKs.

3.1.2. Generate your own SDK

Please feel free to generate an SDK for your own programming language using our Swagger file. Please note that we do not officially support your SDK, but unofficially we are here to help of course.
You can use Swagger codegen for this (https://swagger.io/swagger-codegen). Swagger codegen support almost 100 programming languages.

Please use version 2.X of swagger codegen with our current API's

3.2. Version compatibility between SDK and API

Al of our REST API's follow the versioning scheme below

`XX.YY`

When the major number (XX) changes this means we completely redesign an API.
Minor number (YY) changes means smaller backwards comptible breaks within the API. An API can be changed forward compatible within the same minor number.

Our SDK follow the below versioning scheme

`XX.YY.ZZ`

The major (XX) and minor (YY) number always map directly to the accompanying REST API version. The micro number (ZZ) is used if we add forward compatible changes to our REST API, or when bugs are encountered in a specific SDK.

Chapter 4. Privacy and data storage

This API needs a mongo database to run. Certificates are stored in the database. Please be aware that this PoC has no integration with Hardware Security Modules. That will be part of future Open-Source and Commercial products, based upon lessons learned from this PoC.

Chapter 5. Resources

5.1. Certificates

Certificates and signing

5.1.1. Import certificate

```
POST /eidas/1.0/certificates
```

Description

Import a X509 certificate in base65 form

Parameters

Type	Name	Description	Schema
Body	**certificateImportRequest** *required*	certificateImportRequest	EidasCertificateImportRequest

Responses

HTTP Code	Description	Schema
200	OK	ResponseEntity

Consumes

- `application/json`

Produces

- `/`

Security

Type	Name	Scopes
oauth2	**oauth2schema**	global

5.1.2. Sign input data

```
POST /eidas/1.0/certificates/{name}/sign
```

Description

Create a signature using the named certificate

Parameters

Type	Name	Description	Schema
Path	**name** *required*	name	string
Body	**cadesSignatureRequest** *required*	cadesSignatureRequest	CadesSignatureRequest

Responses

HTTP Code	Description	Schema
200	OK	CadesSignatureResponse

Consumes

- `application/json`

Produces

- `/`

Security

Type	Name	Scopes
oauth2	**oauth2schema**	global

5.2. Signatures

Signature verification

5.2.1. Verify input data

```
POST /eidas/1.0/signatures
```

Description

Verifies a signature

Parameters

Type	Name	Description	Schema
Body	**cadesSignatureVerifyRequest** *required*	cadesSignatureVerifyRequest	CadesSignatureVerifyRequest

Responses

HTTP Code	Description	Schema
200	OK	CadesSignatureVerifyResponse

Consumes

- `application/json`

Produces

- `/`

Security

Type	Name	Scopes
oauth2	**oauth2schema**	global

5.3. Eidas-controller

Eidas Controller

Chapter 6. Security

6.1. oauth2schema

Type : oauth2
Flow : application
Token URL : https://gw.api.cloud.sphereon.com/token

Name	Description
global	accessEverything

Chapter 7. Definitions

7.1. CadesSignatureRequest

Name	Schema
content *optional*	string
password *optional*	string

7.2. CadesSignatureResponse

Name	Schema
signature *optional*	string

7.3. CadesSignatureVerifyRequest

Name	Schema
signature *optional*	string

7.4. CadesSignatureVerifyResponse

Name	Schema
originalData *optional*	string
simpleReport *optional*	XmlSimpleReport
verified *optional*	boolean

7.5. EidasCertificateImportRequest

Name	Schema
base64Certificate *optional*	string
name *optional*	string

7.6. ResponseEntity

Name	Schema
body *optional*	object
statusCode *optional*	enum (ACCEPTED, ALREADY_REPORTED, BAD_GATEWAY, BAD_REQUEST, BANDWIDTH_LIMIT_EXCEEDED, CHECKPOINT, CONFLICT, CONTINUE, CREATED, DESTINATION_LOCKED, EXPECTATION_FAILED, FAILED_DEPENDENCY, FORBIDDEN, FOUND, GATEWAY_TIMEOUT, GONE, HTTP_VERSION_NOT_SUPPORTED, IM_USED, INSUFFICIENT_SPACE_ON_RESOURCE, INSUFFICIENT_STORAGE, INTERNAL_SERVER_ERROR, I_AM_A_TEAPOT, LENGTH_REQUIRED, LOCKED, LOOP_DETECTED, METHOD_FAILURE, METHOD_NOT_ALLOWED, MOVED_PERMANENTLY, MOVED_TEMPORARILY, MULTIPLE_CHOICES, MULTI_STATUS, NETWORK_AUTHENTICATION_REQUIRED, NON_AUTHORITATIVE_INFORMATION, NOT_ACCEPTABLE, NOT_EXTENDED, NOT_FOUND, NOT_IMPLEMENTED, NOT_MODIFIED, NO_CONTENT, OK, PARTIAL_CONTENT, PAYLOAD_TOO_LARGE, PAYMENT_REQUIRED, PERMANENT_REDIRECT, PRECONDITION_FAILED, PRECONDITION_REQUIRED, PROCESSING, PROXY_AUTHENTICATION_REQUIRED, REQUESTED_RANGE_NOT_SATISFIABLE, REQUEST_ENTITY_TOO_LARGE, REQUEST_HEADER_FIELDS_TOO_LARGE, REQUEST_TIMEOUT, REQUEST_URI_TOO_LONG, RESET_CONTENT, SEE_OTHER, SERVICE_UNAVAILABLE, SWITCHING_PROTOCOLS, TEMPORARY_REDIRECT, TOO_EARLY, TOO_MANY_REQUESTS, UNAUTHORIZED, UNAVAILABLE_FOR_LEGAL_REASONS, UNPROCESSABLE_ENTITY, UNSUPPORTED_MEDIA_TYPE, UPGRADE_REQUIRED, URI_TOO_LONG, USE_PROXY, VARIANT_ALSO_NEGOTIATES)
statusCodeValue *optional*	integer (int32)

7.7. XmlCertificate

Name	Schema
id *optional*	string
qualifiedName *optional*	string

7.8. XmlCertificateChain

Name	Schema
certificate *optional*	< XmlCertificate > array

7.9. XmlSemantic

Name	Schema
key *optional*	string
value *optional*	string

7.10. XmlSimpleReport

Name	Schema
containerType *optional*	enum (ASiC_E, ASiC_S)
documentName *optional*	string
semantic *optional*	< XmlSemantic > array
signatureOrTimestamp *optional*	< XmlToken > array
signaturesCount *optional*	integer (int32)
validSignaturesCount *optional*	integer (int32)
validationPolicy *optional*	XmlValidationPolicy

Name	Schema
validationTime *optional*	string (date-time)

7.11. XmlToken

Name	Schema
certificateChain *optional*	XmlCertificateChain
errors *optional*	< string > array
filename *optional*	string
id *optional*	string
indication *optional*	enum (FAILED, INDETERMINATE, NO_SIGNATURE_FOUND, PASSED, TOTAL_FAILED, TOTAL_PASSED)
infos *optional*	< string > array
subIndication *optional*	enum (CERTIFICATE_CHAIN_GENERAL_FAILURE, CHAIN_CONSTRAINTS_FAILURE, CRYPTO_CONSTRAINTS_FAILURE, CRYPTO_CONSTRAINTS_FAILURE_NO_POE, EXPIRED, FORMAT_FAILURE, HASH_FAILURE, NOT_YET_VALID, NO_CERTIFICATE_CHAIN_FOUND, NO_POE, NO_SIGNING_CERTIFICATE_FOUND, OUT_OF_BOUNDS_NOT_REVOKED, OUT_OF_BOUNDS_NO_POE, POLICY_PROCESSING_ERROR, REVOKED, REVOKED_CA_NO_POE, REVOKED_NO_POE, SIGNATURE_POLICY_NOT_AVAILABLE, SIGNED_DATA_NOT_FOUND, SIG_CONSTRAINTS_FAILURE, SIG_CRYPTO_FAILURE, TIMESTAMP_ORDER_FAILURE, TRY_LATER)
warnings *optional*	< string > array

7.12. XmlValidationPolicy

Name	Schema
policyDescription *optional*	string
policyName *optional*	string