KPAPI

ADR-HTTP Message and payload signing with JAdES

KPAPI Guide
Draft 07 juli 2023

This version:
https://geonovum.github.io/KPAPI/

Latest published version:
https://publicatie.centrumvoorstandaarden.nl/dk/template/

Latest editor's draft:
https://geonovum.github.io/KPAPI/

Editor:
Peter Haasnoot (Logius)

Author:
KPAPI (KPAPI)

Participate:
GitHub Geonovum/KPAPI
File an issue
Commit history
Pull requests

Abstract

This ADR Module contains the requirements for ADR-HTTP Message and payload signing with JAdES

- JAdES [JAdES]

This module is based on the *ISA² IPS REST API Profile v1.0 section 5.2.2 Message And Payload Level Security*

Status of This Document

This is a draft that could be altered, removed or replaced by other documents. It is not a recommendation approved by de werkgroep.

Table of Contents

§ 1. ADR-HTTP Message and payload signing with JAdES

> NOTE: Status
>
> This module is under development

§ 1.1 Introduction

This module specifies the use of JAdES signatures for HTTP message and payload siging. The module is directly based on the *ISA² IPS REST API Profile v1.0* (which was a result of the REST API Pilot project for eDelivery)

§ 1.2 JWS detached signatures

This module enforces the use of JWS detached signatures following the HttpHeaders Mechanism of the ETSI ESI JAdES specification [ETSI-JADES].

This structure is enforced for the following reasons:

- JWS, being a simple JSON Structure, can be supported by clients in a light context, while specifications like the ETSI ESI ASIC containers are more difficult to do.
- JWS in detached form does not change the payload structure, meaning that a client not supporting the validation of signature can continue to operate as if there was no signature applied.
- JWS Detached can be transported using an HTTP header, making its presence unintrusive and easily transportable.

§ 1.3 Cryptographic Algorithms

Following ENISA's Good Practises in Cryptography – Primitives and Schemes [ENISA-CRYPTO-2020], the following algorithms found in [RFC7518] are selected for this profile, to be used in the following form:

- The ECDSA Algorithm with SHA-256 and P-256 Curve *MUST* be supported, with a key length of at least 256 bits. The value "ES256" for the alg parameter *MUST* be used in this case as defined in [RFC7518].
- The EdDSA Algorithm [RFC8032] using one of the curves defined in [RFC7748] *SHOULD* be supported and is *RECOMMENDED* for use, with a key length of at least 256 bits. The value "EdDSA" for the alg parameter *MUST* be used in this case and the curve shall be encoded in the crv parameter as defined in [RFC8037].

§ 1.4 Payload signing

Payload signing ensures the integrity and authenticity of the payload part of the message. When payload signing is considered, the Detached JSON Web Signatures following the JAdES specification [ETSI-JADES] *MUST* be applied with the following restrictions:

- The JWS content (Data to be Signed) *MUST* be detached from the signatures as defined in [RFC7515] Appendix F.
- The signed SigD parameter object *MUST* be present in the JWS headers, denoting the use of the JAdES detached header profile.
- The value of the mId parameter *MUST* be set to "http://uri.etsi.org/19182/HttpHeaders".
- The pars array of the SigD *MUST* contain only the element "digest", denoting that for the calculation of the signature only the digest of the HTTP payload must be taken into account, according to [[RFC3230].
- The alg parameter *MUST* be set to the correct value depending on the algorithm used (see above).
- If the alg parameter is set to "EdDSA", the crv parameter *MUST* be set to the correct value (see above).

The JWS structure shall be carried in HTTP header field named **nlgov-adr-payload-sig**. The header field can be used in both requests and responses. The header field *MUST* not appear more than once in a message; if a message contains multiple nlgov-adr-payload-sig header fields, the receiver *MUST* consider the signature invalid.

§ 1.5 HTTP Message signing

The Introduction section of [[DRAFT-IETF-HTTPSBIS-MSG-SIGS] details why message integrity and authenticity are critical to the secure operation of many HTTP/REST applications. When Message-Level Security is considered, the HttpHeaders Mechanism of the JAdES Specification [[ETSI-JADES] *MUST* be used, with the following restrictions applied:

- The JWS content (Data to be Signed) *MUST* be detached from the signatures as defined in [[RFC7515] Appendix F.
- The signed SigD parameter object *MUST* be present in the JWS headers, denoting the use of the JAdES detached header profile.
- The value of the mId parameter *MUST* be set to "http://uri.etsi.org/19182/HttpHeaders".
- The pars array of the SigD *MUST* contain at least the following elements:
    - the element "(request-target)", for containing the HTTP Request URI
    - the element "host", for containing the host the message was submitted to, if present
    - the element "origin", for containing the scheme, hostname, and port from which the request was initiated, if present

  - the element "content-encoding", if present
  - the element "content-type", if present
  - the element "content-length", if present
  - the element "digest", for taking into account the Digest header that contains the hash value of the HTTP payload.
- The alg parameter *MUST* be set to the correct value depending on the algorithm used (see above).
- If the alg parameter is set to "EdDSA", the crv parameter *MUST* be set to the correct value (see above).

Implementations that make use of the HTTP Header fields for data representation *SHOULD* also include these header fields in the pars array. The JWS structure *MUST* be carried in HTTP header field named **nlgov-adr-message-sig**. The header field can be used in both requests and responses. The header field *MUST* not appear more than once in a message; if a message contains multiple nlgov-adr-message-sig header fields, the receiver *MUST* consider the signature invalid.

§ 1.6 Signature Representations

The folowing example is strictly informative !

```
openapi: 3.1.0
info:
    title: JAdES Signatures
    summary: An example showcasing JAdES signatures
    description: An example showcasing JAdES signatures as JWS detached signatur
    termsOfService: https://domain.server.io/terms-of-service
    license:
        name: EUPL-1.2 or later
        url: https://eupl.eu/1.2/en/
    version: 1.0.0
    x-edelivery:
        lifecycle:
            maturity: supported
        publisher:
            name: ACME Publisher
            URL: https://www.acme-publisher.org/
externalDocs:
   description: The ISA² IPS REST API Core Profile
   url: https://joinup.ec.europa.eu/collection/api4dt/document/isa2-ips-rest-api
servers:
- url: https://domain.server.io/v2
```

```
tags:
- name: DetachedPayloadSignature
  description: Operations using payload security
- name: DetachedMessageSignature
  description: Operations using message-level security
paths:
    /openapi:
        get:
            summary: Returns the OpenAPI Document for the API
        ...
        responses:
        200:
            description: ...
            content: {
               $ref: 'https://spec.openapis.org/oas/3.1/schema/2021-05-20'
               ...
            }
    /certificate:
    get:
      tags:
      - DetachedMessageSignature
      summary: Get a Certificate
      securitySchemes:
        OAuth2:
           type: oauth2
        flows:
           authorizationCode:
              authorizationUrl: https://example.com/api/oauth/dialog
              scopes:
                 send:message: send a message
       ...
      responses:
      200:
         headers:
            nlgov-adr-message-sig:
               $ref: '#/components/headers/nlgov-adr-message-sig'
          description: List of Certificates
          content: { ... }
components:
   headers:
      nlgov-adr-payload-sig:
         schema:
             $ref: '#/components/schemas/JwsCompactDetached'
      nlgov-adr-message-sig:
         schema:
             $ref: '#/components/schemas/JwsCompactDetached'
```

```
schemas:
  JwsCompactDetached:
    title: The format for the message-level and payload signature
    description: Defines the string pattern as a regular expression that
    MUST be followed to represent detached JWS compact tokens
    "$id": https://raw.githubusercontent.com/isa2-api4ips/rest-api-profile/m
    "$schema": https://json-schema.org/draft/2020-12/schema
    type: string
    format: jws-compact-detached
    pattern: ^[A-Za-z0-9_-]+(?:(\\.\\.)[A-Za-z0-9_-]+){1}
```

§ 2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words *MUST*, *RECOMMENDED*, and *SHOULD* in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

§ A. References

§ A.1 Normative references

[ENISA-CRYPTO-2020]
ENISA Good Practises in Cryptography – Primitives and Schemes, December 2020. (Limited availability). URL: https://www.enisa.europa.eu/topics/cryptography

[ETSI-JADES]
JAdES digital signatures. URL: https://www.etsi.org/deliver/etsi_ts/119100_119199/11918201/01.01.01_60/ts_11918201v010101p.pdf

[RFC2119]
Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc2119

[RFC7515]
JSON Web Signature (JWS). M. Jones; J. Bradley; N. Sakimura. IETF. May 2015. Proposed Standard. URL: https://www.rfc-editor.org/rfc/rfc7515

[RFC7518]
JSON Web Algorithms (JWA). M. Jones. IETF. May 2015. Proposed Standard. URL: https://www.rfc-editor.org/rfc/rfc7518

[RFC7748]
Elliptic Curves for Security. A. Langley; M. Hamburg; S. Turner. IETF. January 2016. Informational. URL: https://www.rfc-editor.org/rfc/rfc7748

[RFC8032]
Edwards-Curve Digital Signature Algorithm (EdDSA). S. Josefsson; I. Liusvaara. IETF. January 2017. Informational. URL: https://www.rfc-editor.org/rfc/rfc8032

[RFC8037]
CFRG Elliptic Curve Diffie-Hellman (ECDH) and Signatures in JSON Object Signing and Encryption (JOSE). I. Liusvaara. IETF. January 2017. Proposed Standard. URL: https://www.rfc-editor.org/rfc/rfc8037

[RFC8174]
Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc8174

§ A.2 Informative references

[JAdES]
JAdES digital signatures. URL: https://www.etsi.org/deliver/etsi_ts/119100_119199/11918201/01.01.01_60/ts_11918201v010101p.pdf

↑