Workgroup: Network Working Group
Internet-Draft: draft-hardt-gnap-jose-01
Published: 4 July 2020
Intended Status: Standards Track
Expires: 5 January 2021
Authors: D. Hardt, Ed.
SignIn.Org

# JOSE Authentication

## Abstract

TBD

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 5 January 2021.



## Table of Contents

# 1. Introduction

TBD

**Terminology**

This document uses the following terms defined in [GNAP]:

*Authorization URI (AZ URI)

*Client

*Client Handle

*Registered Client

*Dynamic Client

*Grant

*Grant Server (GS)

*GS URI

*NumericDate

*Resource Server (RS)

### Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [RFC2119].

Certain security-related terms are to be understood in the sense defined in [RFC4949]. These terms include, but are not limited to, "attack", "authentication", "authorization", "certificate", "confidentiality", "credential", "encryption", "identity", "sign", "signature", "trust", "validate", and "verify".

Unless otherwise noted, all the protocol parameter names and values are case sensitive.

## 2. JOSE Authentication

How the Client authenticates to the GS and RS are independent of each other. One mechanism can be used to authenticate to the GS, and a different mechanism to authenticate to the RS.

Other documents that specify other Client authentication mechanisms will replace this section.

In the JOSE Authentication Mechanism, the Client authenticates by using its private key to sign a JSON document with JWS per [RFC7515] which results in a token using JOSE compact serialization.

[Editor: are there advantages to using JSON serialization in the body?]

Different instances of a Registered Client MAY have different private keys, but each instance has a certificate to bind its private key to to a public key the GS has for the Client ID. An instance of a Client will use the same private key for all signing operations.

The Client and the GS MUST both use HTTP/2 ([RFC7540]) or later, and TLS 1.3 ([RFC8446]) or later, when communicating with each other.

[Editor: too aggressive to mandate HTTP/2 and TLS 1.3?]

The token may be included in an HTTP header, or as the HTTP message body.

The following sections specify how the Client uses JOSE to authenticate to the GS and RS.

### 2.1. Grant Server Access

The Client authenticates to the GS by passing either a signed header parameter, or a signed message body. The following table shows the method, uri and token location for each Client request to the GS:

| request | http method | uri | token in |
|---|---|---|---|
| Create Grant | POST | GS URI | body |
| Verify Grant | PATCH | Grant URI | body |
| Read Grant | GET | Grant URI | header |
| Update Grant | PUT | Grant URI | body |
| Delete Grant | DELETE | Grant URI | header |
| Read AuthZ | GET | AZ URI | header |
| Update AuthZ | PUT | AZ URI | body |
| Delete AuthZ | DELETE | AZ URI | header |
| GS Options | OPTIONS | GS URI | header |
| Grant Options | OPTIONS | Grant URI | header |
| AuthZ Options | OPTIONS | AZ URI | header |

Table 1

#### 2.1.1. Authorization Header

For requests with the token in the header, the JWS payload MUST contain the following attributes:

**iat** - the time the token was created as a NumericDate.

**jti** - a unique identifier for the token per [RFC7519] section 4.1.7.

**uri** - the value of the URI being called (GS URI, Grant URI, or AZ URI).

**method** - the HTTP method being used in the call ("GET", "DELETE", "OPTIONS")

The HTTP authorization header is set to the "jose" parameter followed by one or more white space characters, followed by the resulting token.

A non-normative example of a JWS payload and the HTTP request follows:

```
{
    "iat"       : 15790460234,
    "jti"       : "f6d72254-4f23-417f-b55e-14ad323b1dc1",
    "uri"       : "https://as.example/endpoint/grant/example6",
    "method"    : "GET"
}
```

```
GET /endpoint/example.grant HTTP/2
Host: as.example
Authorization: jose eyJhbGciOiJIUzI1NiIsIn ...
```

[Editor: make a real example token]

**GS Verification**

The GS MUST verify the token by:

- *TBD

#### 2.1.2. Signed Body

For requests with the token in the body, the Client uses the Request JSON as the payload in a JWS. The resulting token is sent with the content-type set to "application/jose".

A non-normative example (line breaks added to the body for readability):

```
POST /endpoint HTTP/2
Host: as.example
Content-Type: application/jose
Content-Length: 155

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyzdWIiOiIxMjM0NTY3ODkwIiwibmF
tZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMe
Jf36POk6yJV_adQssw5c
```

[Editor: make a real example token]

**GS Verification**

The GS MUST verify the token by:

- *TBD

#### 2.1.3. Public Key Resolution

- ***Registered Clients** MAY use any of the JWS header values to direct the GS to resolve the public key matching the private key linked

to the Client ID. The GS MAY restrict which JWS headers a Client can use.

[Editor: would examples help here so that implementors understand the full range of options, and how an instance can have its own asymetric key pair]

A non-normative example of a JOSE header for a Registered Client with a key identifier of "12":

```
{
    "alg"   : "ES256",
    "typ"   : "JOSE",
    "kid"   : "12"
}
```

* **Dynamic Clients** MUST include their public key in the "jwk" JWS header in a GNAP Create Grant request, unless they have a Client Handle and include it in the GNAP Request JSON "client" object.

A non-normative example of a JOSE header for a Dynamic Client:

```
{
    "alg"   : "ES256",
    "typ"   : "JOSE",
    "jwk"   : {
        "kty"   : "EC",
        "crv"   : "P-256",
        "x"     : "Kgl5DJSgLyV-G32osmLhFKxJ97FoMW0dZVEqDG-Cwo4",
        "y"     : "GsL4mOM4x2e6iON8BHvRDQ6AgXAPnw0m0SfdlREV7i4"
    }
}
```

### 2.2. Resource Server Access

In the "jose" mechanism Section 2.2.2, all Client requests to the RS include a proof-of-possession token in the HTTP authorization header. In the "jose+body" mechanism Section 2.2.3, the Client signs the JSON document in the request if the POST or PUT methods are used, otherwise it is the same as the "jose" mechanism.

#### 2.2.1. JOSE header

The GS provides the Client one or more JWS header parameters and values for a a certificate, or a reference to a certificate or certificate chain, that the RS can use to resolve the public key matching the private key being used by the Client.

A non-normative examples JOSE header:

```
{
    "alg"   : "ES256",
    "typ"   : "JOSE",
    "x5u"   : "https://as.example/cert/example2"
}
```

[Editor: this enables Dynamic Clients to make proof-of-possession API calls the same as Registered Clients.]

#### 2.2.2. "jose" Mechanism

The JWS payload MUST contain the following attributes:

**iat** - the time the token was created as a NumericDate.

**jti** - a unique identifier for the token per [RFC7519] section 4.1.7.

**uri** - the value of the RS URI being called.

**method** - the HTTP method being used in the call

**token** - the access token provided by the GS to the Client

The HTTP authorization header is set to the "jose" parameter followed by one or more white space characters, followed by the resulting token.

A non-normative example of a JWS payload and the HTTP request follows:

```
{
    "iat"       : 15790460234,
    "jti"       : "f6d72254-4f23-417f-b55e-14ad323b1dc1",
    "uri"       : "https://calendar.example/calendar",
    "method"    : "GET",
    "token"     : "eyJJ2D6.example.access.token.mZf9pTSpA"
}
```

```
GET /calendar HTTP/2
Host: calendar.example
Authorization: jose eyJhbG.example.jose.token.adks
```

[Editor: make a real example token]

**RS Verification**

The RS MUST verify the token by:

*verify access token is bound to the public key - include key fingerprint in access token?

*TBD

### 2.2.3. "jose+body" Mechanism

The "jose+body" mechanism can only be used if the content being sent to the RS is a JSON document.

Any requests not sending a message body will use the "jose" mechanism Section 2.2.2.

Requests sending a message body MUST have the following JWS payload:

**iat** - the time the token was created as a NumericDate.

**jti** - a unique identifier for the token per [RFC7519] section 4.1.7.

**uri** - the value of the RS URI being called.

**method** - the HTTP method being used in the call

**token** - the access token provided by the GS to the Client

**body** - the message body being sent to the RS

A non-normative example of a JWS payload and the HTTP request follows:

```
{
    "iat"   : 15790460234,
    "jti"   : "f6d72254-4f23-417f-b55e-14ad323b1dc1",
    "uri"   : "https://calendar.example/calendar",
    "method": "POST",
    "token" : "eyJJ2D6.example.access.token.mZf9pTSpA",
    "payload" : {
        "event" : {
            "title"             : "meeting with joe",
            "start_date_utc"    : "2020-02-21 11:00:00",
            "end_date_utc"      : "2020-02-21 11:00:00"
        }
    }
}
```

```
POST /calendar HTTP/2
Host: calendar.example
Content-Type: application/jose
Content-Length: 155

eyJhbGciOi.example.jose+body.adasdQssw5c
```

[Editor: make a real example token]

**RS Verification**

The RS MUST verify the token by:

- TBD

#### 2.2.4. Public Key Resolution

The RS has a public key for the GS that it uses to verify the certificate or certificate chain the Client includes in the JWS header.

### 2.3. Request Encryption

[Editor: to be fleshed out]

The Client encrypts a request when ??? using the GS public key returned as the ??? attribute in GS Options.

### 2.4. Response Signing

[Editor: to be fleshed out]

The Client verifies a signed response ??? using the GS public key returned as the ??? attribute in GS Options.

### 2.5. Response Encryption

[Editor: to be fleshed out]

The Client decrypts a response when ??? using the private key matching the public key included in the request as the ??? attribute in [GNAP] Grant Request JSON.

## 3. Acknowledgments

TBD

## 4. IANA Considerations

TBD

## 5. Security Considerations

TBD

## 6. Normative References

**[RFC2119]** Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/

RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

**[RFC3966]** Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, DOI 10.17487/RFC3966, December 2004, <https://www.rfc-editor.org/info/rfc3966>.

**[RFC5322]** Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

**[RFC4949]** Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <https://www.rfc-editor.org/info/rfc4949>.

**[RFC7515]** Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

**[RFC7516]** Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

**[RFC7519]** Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

**[RFC7540]** Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/info/rfc7540>.

**[RFC8446]** Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

**[GNAP]** Hardt, D., "The Grant Negotiation and Authorization Protocol", June 2020, <https://tools.ietf.org/html/draft-hardt-xauth-protocol>.

## Appendix A. Document History

### A.1. draft-hardt-gnap-jose-00

*Initial version

### A.2. draft-hardt-gnap-jose-01

*renamed HTTP verb to method

## Author's Address

Dick Hardt (editor)
SignIn.Org
United States

Email: dick.hardt@gmail.com