

# Signature

---

*Open API for FSP Interoperability Specification*

# Signature

## Open API for FSP Interoperability Specification

### Table of Contents

---

Table of Tables .....	3
1 Preface.....	4
1.1 Conventions Used in This Document.....	5
1.2 Document Version Information.....	6
2 Introduction.....	7
2.1 Open API for FSP Interoperability Specification .....	8
3 API Signature Definition .....	9
3.1 Signature Data Model.....	10
3.2 Generating a Signature .....	11
3.3 Validating Signature.....	12
4 API Signature Examples.....	13
4.1 Generating a Signature .....	14
4.2 Validating Signature.....	18

# Signature

## Open API for FSP Interoperability Specification

### Table of Tables

Table 1 – Data model of HTTP header field FSPIOP-Signature .....	10
--	----

# Signature

## Open API for FSP Interoperability Specification

### 1 Preface

---

This section contains information about how to use this document.

# Signature

## Open API for FSP Interoperability Specification

### 1.1 Conventions Used in This Document

---

This document uses the notational conventions for BASE64URL(OCTETS), UTF8(STRING), ASCII(STRING), and || defined in RFC 7515<sup>1</sup>.

The following conventions are used in this document to identify the specified types of information

Type of Information	Convention	Example
Elements of the API, such as resources	Boldface	<b>/authorization</b>
Variables	Italics within angle brackets	< <i>ID</i> >
Glossary terms	Italics on first occurrence; defined in <i>Glossary</i>	The purpose of the API is to enable interoperable financial transactions between a <i>Payer</i> (a payer of electronic funds in a payment transaction) located in one <i>FSP</i> (an entity that provides a digital financial service to an end user) and a <i>Payee</i> (a recipient of electronic funds in a payment transaction) located in another FSP.
Library documents	Italics	User information should, in general, not be used by API deployments; the security measures detailed in <i>Signature</i> and <i>Encryption</i> should be used instead.

---

<sup>1</sup> <https://tools.ietf.org/html/rfc7515#section-1.1> – JSON Web Signature (JWS) - Notational Conventions

# Signature

## Open API for FSP Interoperability Specification

### 1.2 Document Version Information

---

Version	Date	Change Description
1.0	2019-02-21	Initial version

# Signature

## Open API for FSP Interoperability Specification

### 2 Introduction

---

This document details the security methods to be implemented for Open API for FSP Interoperability (hereafter cited as the API) to ensure *integrity* and *non-repudiation* between the API client and the API server.

In information security, *data integrity* means maintaining and assuring the accuracy and completeness of data over its entire life-cycle. For the API, data integrity means that an API message cannot be modified in an unauthorized or undetected manner by parties involved in the API communication.

In legal terms, *non-repudiation* means that a person intends to fulfill their obligations to a contract. It also means that one party in a transaction cannot deny having received the transaction, nor can the other party deny having sent the transaction. For the API, non-repudiation means that an API client cannot deny having sent an API message to a counterparty. JSON Web Signature (JWS), as defined in RFC 7515<sup>2</sup>, must be applied to the API to provide message integrity and non-repudiation for either component fields of an API payload or the full API payload. Whenever an API client sends an API message to a counterparty, the API client should sign the message using its private key. After the counterparty receives the API message, the counterparty must validate the signature with the API client's public key. Only the HTTP request message of an API message need to be signed, any HTTP response message of the APIs SHALL NOT be signed.

**Note:** The corresponding public key should either be shared in advance with the counterparty or retrieved by the counterparty (for example, the local scheme Certificate Authority).

Because intermediary fees are not supported in the current version of the API, intermediaries involved in API message-transit may not modify the API message payload. Thus, the signature at full payload level is used to protect the integrity of the full payload of an API message from end-to-end. Regardless of how many intermediaries there are in transit, the original payload cannot be modified by the intermediaries. The final recipient of the API message must validate the signature generated by the original API client based on the message payload received.

**Note:** Whether the signature needs to be validated by the intermediaries in transit is determined by the internal implementation of each intermediary or the local schema.

**Note:** In a future version of the API, intermediary fees may be supported; at that time, signature-at-field-level may also be supported. However, both features are out-of-scope for the current version of the API.

---

<sup>2</sup> <https://tools.ietf.org/html/rfc7515> - JSON Web Signature (JWS)

# Signature

## Open API for FSP Interoperability Specification

### 2.1 Open API for FSP Interoperability Specification

---

The Open API for FSP Interoperability Specification includes the following documents.

#### 2.1.1 General Documents

- *Glossary*

#### 2.1.2 Logical Documents

- *Logical Data Model*
- *Generic Transaction Patterns*
- *Use Cases*

#### 2.1.3 Asynchronous REST Binding Documents

- *API Definition*
- *JSON Binding Rules*
- *Scheme Rules*

#### 2.1.4 Data Integrity, Confidentiality, and Non-Repudiation

- *PKI Best Practices*
- *Signature*
- *Encryption*



# Signature

## Open API for FSP Interoperability Specification

### 3 API Signature Definition

---

This section introduces the technology used by the API signature, including the data exchange format for the signature of an API message and the mechanism used to generate and verify a signature.

# Signature

## Open API for FSP Interoperability Specification

### 3.1 Signature Data Model

The API uses a customized HTTP header parameter **FSPIOP-Signature** to represent the signature that is produced by the initiating API client for the API message. The data model for this parameter is described in Table 1.

**Note:** Currently the API does not support intermediaries in an API message; only the message-initiator can sign a message. If this is required in the future, there will be new customized HTTP header parameter, but this is out-of-scope for the current version of the API.

Name	Cardinality	Type	Description
protectedHeader	1	String(1..32768)	<p>This element indicates the HTTP header parameters that are protected by the signature. Its value must be BASE64URL(UTF8(JWS Protected Header)).</p> <p>According to JWS specification, the <b>alg</b> header parameter must be present to identify the cryptographic algorithm used to secure the JWS.</p> <p>A customized parameter <b>FSPIOP-URI</b> that represents the URI path and query parameters of HTTP request message of the APIs must be present.</p> <p>A customized parameter <b>FSPIOP-HTTP-Method</b> that holds the HTTP method used in the HTTP message must be present.</p> <p>A customized parameter <b>FSPIOP-Source</b> that represents the system which sent the API request must be present.</p> <p>The customized HTTP header parameter <b>FSPIOP-Destination</b> is mandatory in <b>protectedHeader</b> if it is present in the HTTP header of the request.</p>
signature	1	String(1..512)	<p>This element indicates the signature. Its value is part of JWS serialization; that is, BASE64URL(JWS Signature).</p>

Table 1 – Data model of HTTP header field FSPIOP-Signature

# Signature

## Open API for FSP Interoperability Specification

### 3.2 Generating a Signature

---

To create the signature for an API message, the following steps are performed. The order of the steps is not significant in cases where there are no dependencies between the inputs and outputs of the steps.

1. Create the content to be used as the JWS Payload. Because the signature is currently at full payload level, the full HTTP body of the API message is the JWS Payload.
2. Compute the encoded payload value `BASE64URL(JWS Payload)`.
3. Create the JSON object or objects containing the desired JWS Protected Header.
  - A. The **alg** JWS Protected Header parameter must be present. In the API, the available algorithms for the signature are **RS256**, **RS384**, **RS512**. A key of size 2048 bits or larger must be used with these algorithms.
  - B. Other parameters registered in the IANA *JSON Web Signature and Encryption Header Parameters*<sup>3</sup> are optional.
  - C. The customized parameter **FSPIOP-URI** must be included in JWS Protected Header to protect the URI path and query parameters of the APIs.
  - D. The customized parameter **FSPIOP-HTTP-Method** must be included in JWS Protected Header to protect the HTTP request operation method.
  - E. The parameter **FSPIOP-Source** must be present, and its value comes from the corresponding HTTP header parameter **FSPIOP-Source**.
  - F. The parameter **FSPIOP-Destination** must be present if it is present in the HTTP header of the request, and its value comes from the corresponding HTTP header parameter **FSPIOP-Destination**.
  - G. Other HTTP Header parameters of the APIs are recommended to be included in JWS Protected Header, but they are optional in this JWS Protected Header.
4. Compute the encoded header value `BASE64URL(UTF8(JWS Protected Header))`.
5. Compute the JWS Signature according to the JWS specification using the output of Step 2 and Step 4.
6. Compute the encoded signature value `BASE64URL(JWS Signature)`.
7. Compute the value for the HTTP header parameter **FSPIOP-Signature** as described in Section 3.1. The value for this **FSPIOP-Signature** is a JSON Object Serialization string.

**Note:** If JSON Web Encryption (JWE) is used to encrypt some fields of the payload (for more information, see *Encryption*), then the API client should first encrypt the desired fields, then replace the plain text of those fields with the encoded cipher text in the payload, and then finally sign the payload.

---

<sup>3</sup> <https://www.iana.org/assignments/jose/jose.xhtml#web-signature-encryption-header-parameters> - JSON Web Signature and Encryption Header Parameters

# Signature

## Open API for FSP Interoperability Specification

### 3.3 Validating Signature

---

When validating the signature of an API request, the following steps are performed. The order of the steps is not significant in cases where there are no dependencies between the inputs and outputs of the steps. If any of the listed steps fails, then the signature cannot be validated.

1. Parse the HTTP header parameter **FSPIOP-Signature** to get the components **protectedHeader** and **signature**.
2. Use BASE64URL to decode the encoded representation of the JWS Protected Header. Verify that the resulting octet sequence is a UTF-8-encoded representation of a completely valid JSON object conforming to JSON Data Interchange Format, defined in RFC 7159<sup>4</sup>.
3. Verify the parameters in the JWS Protected Header.
  - a) The parameter **alg** must be present and its value must be one of **RS256**, **RS384**, **RS512**.
  - b) Other parameters registered in the IANA *JSON Web Signature and Encryption Header Parameters* are optional.
  - c) The parameter **FSPIOP-URI** must be present and its value must be the same as the input URL value of the request.
  - d) The parameter **FSPIOP-HTTP-Method** must be present and its value must be same as the operation method of the request.
  - e) The parameter **FSPIOP-Source** must be present, and its value must be the same as the corresponding HTTP header parameter **FSPIOP-Source**.
  - f) The parameter **FSPIOP-Destination** must be present if it is present in the HTTP header of the request, and its value must be same as the corresponding HTTP header parameter **FSPIOP-Destination**.
  - g) If there are other HTTP header parameters present in JWS Protected Header, then their values must be validated with the corresponding HTTP header values.
4. Compute the encoded payload value BASE64URL(JWS Payload). Because the current signature is at full payload level, the full HTTP body of the API message is the JWS Payload.
5. Validate the JWS Signature against the JWS Signing Input ASCII(BASE64URL(UTF8(JWS Protected Header)) || '.' || BASE64URL(JWS Payload)) in the manner defined for the algorithm being used, which must be accurately represented by the value of the **alg** (algorithm) Header Parameter.
6. Record whether the validation succeeded.

---

<sup>4</sup> <https://tools.ietf.org/html/rfc7159> - The JavaScript Object Notation (JSON) Data Interchange Format

# Signature

## Open API for FSP Interoperability Specification

### 4 API Signature Examples

This section uses a typical quote process to explain how the API signature is implemented using JWS. The FSPs in the API can verify that their internal implementation for API signature is correct using the following case.

The case in this section uses RS256 as the signature algorithm. The RSA key used for the signature example is represented in JSON Web Key (JWK), defined in RFC 7517<sup>5</sup>, format below (with line breaks and indentation within values for display purposes only):

```
{
  "kty": "RSA",
  "n": "ofgWCuLjybRlzo0tZWJjNiuSfb4p4fAkd_wWJcyQoTbji9k0l8W26mPddx
    HmfHQp-Vaw-4qPCJrcS2mJPMEzP1Pt0Bm4d4Q1L-yRT-SFd2lZS-pCgNM5
    D1W_YpRPEwQWvG6b32690r2jZ47soMZo9wGzjb_70Mg0LOL-bSf63kpaSH
    SXndS5z5rexMdbBYUsLA9e-KXBdQOS-UTo7WTBEMa2R2CapHg665xsmtdV
    MTBQY4uDZlxb3qCo5ZwKh9kG4LT6_I5Ih1JH7aGhyxXFvUK-DWNmoudF8
    NAc09_h9iaGNj8q2ethFkMLs91kzk2PAcDTW9gb54h4FRWyuXpoQ",
  "e": "AQAB",
  "d": "Eq5xpGnNCivDflJjsRQBxHx1hdR1k6U1we2JZD50LpXyWPEAeP88vLN097I
    j1A7_GQ5sLKMgvfTeXZx9SE-7YwVol2NX0oAJe46sui395IW_GO-pWJ100
    BkTGoVeN2bKVRUCgu-GjBVaYLU6f3l9kJfFNS3E0QbVdxzubSu3Mkqzjkn
    439X0M_V51gfpRLI9JYanrC4D4qAdGcopV_0ZHHZQ1BjudU2QvXt4ehNYT
    CBr6XCLQUShb1juU01ZdiYoFaFQT5Tw8bGU1_x_jTj3ccPDVZFD9pIuhLh
    B0neufuBiB4cS98l2SR_RQyGWSeWjnczT0QU91p1DhOVRu0opznQ",
  "p": "4BzEE0tIpmVdVEZNCqS7baC4crd0pqnRH_5IB3jw3bcxGn6QLvnEtfdUdi
    YrqBdss1l58BQ3KhooKeQTa9AB0Hw_Py5PJdTJNPY8cQn7ouZ2KKDcmnPG
    BY5t7yLc1Q1Q5xHdwW1VhvKn-nXqhJTBgIPgtldC-KDV5z-y2XDwGUc",
  "q": "uQPEfgmVtjL0Uyyx88GZFF1fOunH3-7cepKmtH4pxhtCoHqpWmT8YAmZxa
    ewHgHAjLYsp1ZSe7zFYHj7C6u17TjeLQeZD_YwD66t62wDmpe_H1B-TnBA
    -njbglfIsRLtXlnDzQkv5dTltRJ11BKBBypEEF6689rjcJIDEz9RWdc",
  "dp": "BwKfV3Akq5_MFZDFZCnW-wz1-CCo83WoZvnLQwCTeDv8uzluRSnm71I3Q
    CLdhrqE2e9YkxvuxdBfpT_Pi7Yz-FOKnu1R6HsJeDCjn12Sk3vmAktV2zb
    34MCdy7cpdTh_YVr7tss2u6vneTwrA86rZtu5Mbr1C1XsmvKxHQAdYo0",
  "dq": "h_96-mK1R_7glhsum81dZxjTnYynPbZpHzizJeeHcXYsXaaMwk0l0DsWa
    7I9xXDoRwbKgB719rrmI2oKr6N3Do9U0ajaHF-NKJnwgjMd2w9cjz3_-ky
    N1xAr2v4IKhGNpmM5iIgOS1VZn0Z68m6_pblBSp3nssTdlqvd0tIiTHU",
  "qi": "IYd7DH0hrWvxkwPQsRM2tOgrjbcrfvtQJipd-DlCxyVuuM9sQLdgjV2k2o
    y26F0EmpScGLq2MowX7fhd_QJQ3ydy5cY7YIBi87w93IKLEdfnbJtoOPLU
    W0ITrJRe0go1cq9SbsxYawBgfp_gh6A5603k2-ZQwVK0JKSHuLFkuQ3U"
}
```

<sup>5</sup> <https://tools.ietf.org/html/rfc7517> - JSON Web Key (JWK)

# Signature

## Open API for FSP Interoperability Specification

### 4.1 Generating a Signature

---

The following message text is an example of **POST /quotes** without a signature sent by Payer FSP to a counterparty (line breaks and indentation within values for display purposes only).

```
POST /quotes HTTP/1.1
Accept: application/vnd.interoperability.quotes+json;version=1.0
FSPIOP-Source:1234
FSPIOP-Destination:5678
Content-Length:975
Date:Tue, 23 May 2017 21:12:31 GMT
Content-Type:application/vnd.interoperability.quotes+json;version=1.0

{
  "amount": { "amount": "150", "currency": "USD" },
  "transactionType": {
    "scenario": "TRANSFER", "initiator": "PAYER",
    "subScenario": "P2P Transfer across MM systems",
    "initiatorType": "CONSUMER"
  },
  "transactionId": "36629a51-393a-4e3c-b347-c2cb57e1e1fc",
  "quoteId": "59e331fa-345f-4554-aac8-fcd8833f7d50",
  "expiration": "2017-05-24T08:40:00.000-04:00",
  "payer": {
    "personalInfo": {
      "dateOfBirth": "1986-02-14",
      "complexName": { "middleName": "Ben",
        "lastName": "Lee", "firstName": "Bill" } },
    "name": "Bill Lee",
    "partyIdInfo": { "fspId": "1234", "partyIdType": "MSISDN",
      "partySubIdOrType": "RegisteredCustomer",
      "partyIdentifier": "16135551212" }
  },
  "payee": {
    "partyIdInfo": { "fspId": "5678",
      "partyIdType": "MSISDN",
      "partyIdentifier": "15295558888" }
  },
  "fees": { "amount": "1.5", "currency": "USD" },
  "extensionList": [
    { "value": "value1", "key": "key1" },
    { "value": "value2", "key": "key2" },
    { "value": "value3", "key": "key3" } ],
  "note": "this is a sample for POST /quotes",
  "geoCode": {
    "longitude": "125.520001", "latitude": "57.323889" },
  "amountType": "RECEIVE"
}
```

#### 4.1.1 Computing Signature Input

According to JWS specification, the signature input is `BASE64URL(UTF8(JWS Protected Header)) || '.' || BASE64URL(JWS Payload)`.

## Signature

## Open API for FSP Interoperability Specification

Assuming the HTTP header parameters **Date** and **FSPIOP-Destination** are protected by the signature, and the algorithm RS256 is used to sign the message, the JWS Protected Header in this case is as follows (line breaks and indentation within values for display purposes only):

```
{
  "alg":"RS256",
  "FSPIOP-Destination":"5678",
  "FSPIOP-URI":"/quotes",
  "FSPIOP-HTTP-Method":"POST",
  "Date":"Tue, 23 May 2017 21:12:31 GMT",
  "FSPIOP-Source":"1234"
}
```

Encoding this JWS Protected Header as BASE64URL(UTF8(JWS Protected Header)) gives this value:

eyJhbGciOiJSUzI1NiIsIkZTUElPUC1EZXXN0aw5hdGlvbI6IjU2NzgiLCJGU1BJT1AtVWJJiJo  
iL3F1b3RlcycIsIkZTUElPUC1IVFRQLU1ldGhvZCI6IlBPU1QiLCJEYXRlIjoivHVhVlCAyMyBNYX  
kgMjAxNyAyMTQxMjE1MSBHTVQilLCJGU1BJT1AtU291cmNlIjoimTIzNCJ9

In this case, JWS Payload is the HTTP Body described in Section 4.1. Encoding this JWS Payload as BASE64URL(JWS Payload) gives this value:

eyJwYXl1ZSI6eyJwYXJ0eUlkSW5mbYI6eyJwYXJ0eUlkSVHlWZSI6Ik1TTSVNETiIsInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcmFuc2Z1ciBhY3Jvc3MgTU0gc3lzdGVTcyIsImluaXRpYXRvc1R5cGUiOiJDT05TVU1FUjJ9LCJub3R1IjoidGhpcyBpcyBhIHhXbXBzZSBmb3IgeUE9TVCAvcXVvdGVzIiwiYW1vdW50IjP7ImFtb3VudCI6IjE1MCIsImN1cnJlbnN5IjoiVVNEIn0sImZlZXMiOnsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJlVU0QifSwiZXh0ZW5zaW9uTGldZCI6W3sidmFsdWUiOiJ2YXw1ZTEiLCJrZXkiOiJrZXkxIn0seyJ2YXw1ZSI6InZhbnV1MiIsImtleSI6ImtleTIifSx7InZhbnV1IjoidmFsdWUzIiwiaw2V5Ijoia2V5MyJ9XSwiZ2VvQ29kZSI6eyJsYXRpdHVkZSI6IjU3LjMyMzg4OSIsImxvbmdpdHVkZSI6IjEyNS41MjAwMDEifSwiZXhwaXJhdGlvbiI6IjIwMTctMDUtMjRUMDg6NDA6MDAuMDAwLTA0OjAwIiwicGF5ZSI6IjoiOnsicGVyc29uYXw1bnV1IjP7ImNvbXBzZXh0YW11IjP7ImZpcnN0TmFtZSI6Ik1jbGwiLCJtaWRkbGV0YVw1IjoiQmVuIiwiTGZzdE5hbWUiOiJlZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LTAYLTE0In0sInBhcnR5SWR1bnRpZm11ciI6IjE1Mjk1NTU0ODg4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB1IjoiUkVDRULWRSIsInRyYW5zYWNoaw9uVHlwZSI6eyJzY2VuYXJpbyI6I1RSQU5TRkVSIiwiaw5pdGlhdG9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6I1AyUCBUcm

### 4.1.2 Producing Signature

Use the given RSA Private Key, the JWS Protected Header and the JWS Payload to generate the signature, then encoding the signature as BASE64URL(JWS Signature) produces this value:

# Signature

## Open API for FSP Interoperability Specification

dz2ntyS0\_rDyA0pLeWluG--tBcYYr1vG99ffkXcEB-dz2ntyS0\_rDyA0pLeWluG--tBcYYr1vG9  
9ffkXcEB-uve5Qzvzyn0ZUi82J7h17RsdfHPuTnbEGvCeU9Y4Bg0nIZHGL4icswaa009T5hPPY-  
KBTzVQeHkokLmL4dXpHdr1ggSEpu3WEU3nfg0FGGAdOq355i1iGuDbhqm\_1SfVHaqdVCEhkJ2Y\_  
r2gl02QpdZrcbvsBV39derj\_PlfISBBGjdh0dIPxnFIVcZuPHiq9Ha2Ms1rBHfqwFfNeU\_xhErB  
d2PywkDQJbK01fqdkmFC9bS80fx006Mg7qdFGw-QkseJTfp0HMBH1d9e6H0cocY8xfuDNGaZp0J  
hxiYtiPLg

### 4.1.3 Re-produce API Request with Signature

As described in Section 3.1, the API signature is represented by a customized HTTP header parameter **FSPIOP-Signature**; thus the API request with the signature in this case is the following message text (line breaks and indentation within values for display purposes only).



# Signature

## Open API for FSP Interoperability Specification

```
POST /quotes HTTP/1.1
FSPIOP-Destination: 5678
Accept: application/vnd.interoperability.quotes+json;version=1.0
Content-Length: 975
Date: Tue, 23 May 2017 21:12:31 GMT
FSPIOP-Source: 1234
Content-Type: application/vnd.interoperability.quotes+json;version=1.0
FSPIOP-Signature: { "signature":
    "dz2ntyS0_rDyA0pLeWluG- -tBcYYr1vG99ffkXcEB-
    uve5Qzvzyn0ZUi82J7h17RsdfHPuTnbEGvCeU9Y4Bg0nIZHGL4icswaa009T5hPPY-
    KBTzVQeHkokLmL4dXpHdr1ggSEpu3WEU3nfgOFGGAd0q355i1iGuDbhqm_1SfVHaqdVCEh
    kJ2Y_r2gl02QpdZrcbvSBV39derj_P1fISBBGjdh0dIPxnFIVcZuPHiq9Ha2MslrBHFqWf
    fNeU_xhErBd2PywkDQJbK01fqdkmFC9bS80fx006Mg7qdFGw-
    QkseJTfp0HMBh1d9e6H0cocY8xfuDNGaZp0JhxiYtiPLg", "protectedHeader":
    "eyJhbGciOiJSUzI1NiIsIkZTUElPUC1EZXN0aW5hdGlvbiI6IjU2NzgiLCJGU1BJT1AtV
    VJJJjoiL3F1b3RlcysIkZTUElPUC1IVFRQLU1ldGhvZCI6IiBPU1QiLCJEYXRlIjoiVHV
    lLCAyMyBNYXkgMjAxNyAyMToxMjoxMSBHTVQiLCJGU1BJT1AtU291cmNlIjoiMTIzNCJ9"
  }

{
  "amount": { "amount": "150", "currency": "USD" },
  "transactionType": {
    "scenario": "TRANSFER", "initiator": "PAYER",
    "subScenario": "P2P Transfer across MM systems",
    "initiatorType": "CONSUMER" },
  "transactionId": "36629a51-393a-4e3c-b347-c2cb57e1e1fc",
  "quoteId": "59e331fa-345f-4554-aac8-fcd8833f7d50",
  "expiration": "2017-05-24T08:40:00.000-04:00",
  "payer": {
    "personalInfo": { "dateOfBirth": "1986-02-14",
      "complexName": { "middleName": "Ben",
        "lastName": "Lee", "firstName": "Bill" } },
    "name": "Bill Lee",
    "partyIdInfo": { "fspId": "1234", "partyIdType": "MSISDN",
      "partySubIdOrType": "RegisteredCustomer",
      "partyIdentifier": "16135551212" } },
  "payee": { "partyIdInfo": { "fspId": "5678",
    "partyIdType": "MSISDN",
    "partyIdentifier": "15295558888" } },
  "fees": { "amount": "1.5", "currency": "USD" },
  "extensionList": [
    { "value": "value1", "key": "key1" },
    { "value": "value2", "key": "key2" },
    { "value": "value3", "key": "key3" } ],
  "note": "this is a sample for POST /quotes",
  "geoCode": { "longitude": "125.520001", "latitude": "57.323889" },
  "amountType": "RECEIVE"
}
```

## Signature

## Open API for FSP Interoperability Specification

## 4.2 Validating Signature

After the Payee FSP receives the **POST /quotes** API message from Payer FSP, the Payee FSP must validate the signature signed by the Payer FSP.

### 4.2.1 Parse FSPIOP-Signature

1. Parse the HTTP header parameter **FSPIOP-Signature** to get the components **protectedHeader** and signature. In this case, the value of **protectedHeader** is:

eyJhbGciOiJSUzI1NiIsIkZTUElPUC1EZXN0aW5hdGlvbiI6IjU2NzgiLCJGU1BJT1AatVVJJiJjo  
iL3F1b3RlcyIsIkZTUElPUC1IVFRQLU1ldGhvZCI6IlBPU1QiLCJFYXRlIjoieiVHV1LCAyMyBNYX  
kgMjAxNyAyMT0xMj0zMSBHTVQiLCJGU1BJT1AatU291cmNlIjoieMTIzNCJ9

2. Use BASE64URL to decode the encoded representation of the JWS Protected Header. Verify that the resulting octet sequence is a UTF-8-encoded representation of a completely valid JSON object conforming to JSON Data Interchange Format, defined in RFC7159. In this case, the decoded JSON object is:

```
{
  "alg":"RS256",
  "FSPIOP-Destination":"5678",
  "FSPIOP-URI":"/quotes",
  "FSPIOP-HTTP-Method":"POST",
  "Date":"Tue, 23 May 2017 21:12:31 GMT",
  "FSPIOP-Source":"1234"
}
```

3. Verify that the **alg** parameter is valid for the API. That means it must be in the list of **RS256**, **RS384**, **RS512**. In this case, the value of **alg** is **RS256**, which is valid.
4. Verify that the value of the parameter **FSPIOP-URI** is same as the input URL of this API message.
5. Verify that the value of the parameter **FSPIOP-HTTP-Method** is same as the HTTP method of this API message.
6. Verify that the value of the HTTP header parameter **FSPIOP-Source** is the same as the corresponding value listed in this JWS Protected Header.
7. Verify that the values for the HTTP header parameter **FSPIOP-Destination** are the same as the corresponding values stated in this JWS Protected Header.
8. Verify the other protected HTTP header parameters. In this case, the **Date** parameter is protected by JWS Protected Header. If the parameters **Date** in the HTTP header of this API message and **Date** in the JWS Protected Header are equal, then the validation is successful. Both **Date** parameters in the example should be the following value:

"Tue, 23 May 2017 21:12:31 GMT"

The validation is passed.

### 4.2.2 Verify JWS Signature

1. In this case, the JWS Payload is the full HTTP body of the API message, that is (line breaks and indentation within values for display purposes only):

# Signature

## Open API for FSP Interoperability Specification

```
{
  "amount": { "amount": "150", "currency": "USD" },
  "transactionType": { "scenario": "TRANSFER", "initiator": "PAYER",
    "subScenario": "P2P Transfer across MM systems",
    "initiatorType": "CONSUMER"
  },
  "transactionId": "36629a51-393a-4e3c-b347-c2cb57e1e1fc",
  "quoteId": "59e331fa-345f-4554-aac8-fcd8833f7d50",
  "expiration": "2017-05-24T08:40:00.000-04:00",
  "payer": {
    "personalInfo": { "dateOfBirth": "1986-02-14",
      "complexName": { "middleName": "Ben",
        "lastName": "Lee", "firstName": "Bill" } },
    "name": "Bill Lee",
    "partyIdInfo": { "fspId": "1234",
      "partyIdType": "MSISDN",
      "partySubIdOrType": "RegisteredCustomer",
      "partyIdentifier": "16135551212" } },
  "payee": {
    "partyIdInfo": { "fspId": "5678",
      "partyIdType": "MSISDN",
      "partyIdentifier": "15295558888" } },
  "fees": { "amount": "1.5", "currency": "USD" },
  "extensionList": [
    { "value": "value1", "key": "key1" },
    { "value": "value2", "key": "key2" },
    { "value": "value3", "key": "key3" } ],
  "note": "this is a sample for POST /quotes",
  "geoCode": { "longitude": "125.520001", "latitude": "57.323889" },
  "amountType": "RECEIVE"
}
```

2. Compute the encoded payload value BASE64URL(JWS Payload). Get the encoded value as:

## Signature

## Open API for FSP Interoperability Specification

eYjWxYl1ZSI6eyJwYXJ0eUlkSW5mbYI6eyJwYXJ0eUlkVHlwZSI6Ik1TTSVNNETiIsInBhc nR5SWR  
lbnRpZmllciI6IjE1Mjk1NTU0dG4IiwiZnNwSWQiOiI1Njc4In19LCJhbW91bnRUeXB lIjoiUk  
VDRULWRSIsInRyYW5zZWNOaw9uVHlwZSI6eyJzY2VuYXJpbyI6IlRSQU5TRkVSiiwiaW5pdGlhd  
G9yIjoiUEFZRVIiLCJzdWJTY2VuYXJpbyI6ILAYUCBUcmFuc2ZlciBhY3Jvc3MgTU0gc3lzdGVt  
cyIsImluaXRpYXRvclR5cGUiOiJDT05TVU1FU iJ9LClub3RlIjo idGhp cyBpcyBH IHNhbXBs ZSB  
mb3IgUE9TVCAvcXVvdGVzIiwiYW1vdW50Ijp7ImFtb3VudCI6IjE1MCIsImN1cnJlbmN5IjoiVV  
NEIn0sImZlZXMiO nsiYW1vdW50IjoiMS41IiwiY3VycmVuY3kiOiJVU0QifSwiZXh0ZW5zaW9uT  
GlzdCI6W3sidmFsdWUiOiJ2YWxzTEiLCJrZXkiOiJrZXkxIn0seyJ2YWxzI6InZhbmHVlMiIs  
ImtleSI6ImtleTIifsx7InZhbmHVlIjo idmFsdWUzIiwiia2V5Ijoia2V5MyJ9XS wiZ2VvQ29kZSI  
6eyJsYXRpdHVkZSI6IjU3L jMyMzg4OSIsImxbmdpdHVkZSI6IjEyNS41MjAwMDE ifSwiZXhwaX  
JhdGlvbiI6IjIwMTctMDUtMjRUMDg6ND A6MDAuMDAwLTA00jAwIiwicGF5ZXIiOnsicGVyc29uY  
WxJbmZvIjp7ImNvbXBsZXh0YW1lIjp7ImZpcnN0TmFtZSI6Ik JpbGwiLCJtaWRkbGV0YW1lIjoi  
QmVuIiwiTG FzdE5hbWUiOiJMZWUifSwiZGF0ZU9mQmlydGgiOiIxOTg2LT AyLTE0In0sInBhc nR  
5SWRJbmZvIjp7InBhc nR5SWRUeXB lIjoiTVNJU0R0IiwiicGFydHlTdWJJZE9yVHlwZSI6IlJlZ2  
lzdGVyZWRddXN0b21lc iIsInBhc nR5SWRlbnRpZmllciI6IjE2MTM1NTUxMjEyIiwiZnNwSQiO  
iIxMjM0In0sIm5hbWUiOiJCawxsIExlZSJ9LClxdW90ZUl kIjoiNTl1MzMxZmEtMz Q1Zi00NTU0  
LWFhYzgtZmNkODgzM2Y3ZDUwIiwiidHJhbnNhY3Rpb25JZCI6IjM2NjI5YTUxLTM5M2EtNGUzYy1  
iMzQ3LWMyY2I1N2UxZTFmYyJ9

3. Validate the JWS Signature against the JWS Signing Input (that is, the JWS Protected Header, JWS Payload) with the specified algorithm **RS256** (specified in the JWS Protected Header), and the public key. Record whether the validation succeeded or not.