STIR
Internet-Draft
Intended status: Standards Track
Expires: March 30, 2017

C. Wendt
Comcast
J. Peterson
Neustar Inc.

September 26, 2016

Personal Assertion Token (PASSporT)
draft-ietf-stir-passport-08

Abstract

This document defines a method for creating and validating a token that cryptographically verifies an originating identity, or more generally a URI or telephone number representing the originator of personal communications. The PASSporT token is cryptographically signed to protect the integrity of the identity the originator and to verify the assertion of the identity information at the destination. The cryptographic signature is defined with the intention that it can confidently verify the originating persona even when the signature is sent to the destination party over an insecure channel. PASSporT is particularly useful for many personal communications applications over IP networks and other multi-hop interconnection scenarios where the originating and destination parties may not have a direct trusted relationship.

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 http://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 March 30, 2017.

Table of Contents

1. Introduction

In today's IP-enabled telecommunications world, there is a growing concern about the ability to trust incoming invitations for communications sessions, including video, voice and messaging [RFC7340]. As an example, modern telephone networks provide the ability to spoof the calling party telephone number for many legitimate purposes including providing network features and services on the behalf of a legitimate telephone number. However, as we have seen, bad actors have taken advantage of this ability for illegitimate and fraudulent purposes meant to trick telephone users to believe they are someone they are not. This problem can be extended to many emerging forms of personal communications.

This document defines a method for creating and validating a token that cryptographically verifies an originating identity, or more generally a URI or telephone number representing the originator of personal communications. Through extensions defined in this document, in Section 7, other information relevant to the personal communications can also be added to the token. The goal of PASSporT is to provide a common framework for signing originating identity related information in an extensible way. Additionally, this functionality is independent of any specific personal communications signaling call logic, so that the assertion of originating identity related information can be implemented in a flexible way and can be used in applications including end-to-end applications that require different signaling protocols or gateways between different communications systems. It is anticipated that signaling protocol specific guidance will be provided in other related documents and specifications to specify how to use and transport PASSporT tokens, however this is intentionally out of scope for this document.

[I-D.ietf-stir-rfc4474bis] provides details of the use of PASSporT within SIP [RFC3261] signaling protocol for the signing and verification of telephone numbers and SIP URIs.

2. PASSporT Token Overview

JSON Web Token (JWT) [RFC7519] and JSON Web Signature (JWS) [RFC7515] and related specifications define a standard token format that can be used as a way of encapsulating claimed or asserted information with

an associated digital signature using X.509 based certificates. JWT provides a set of claims in JSON format that can conveniently accommodate asserted originating identity information and is easily extensible for extension mechanisms defined below. Additionally, JWS provides a path for updating methods and cryptographic algorithms used for the associated digital signatures.

JWS defines the use of JSON data structures in a specified canonical format for signing data corresponding to JOSE header, JWS Payload, and JWS Signature. JWT defines a set of claims that are represented by specified key value pairs which can be extended with custom keys for specific applications. The next sections define the header and claims that MUST be minimally used with JWT and JWS for PASSporT.

3. PASSporT Header

The JWS token header is a JOSE header, [RFC7515] Section 4, that defines the type and encryption algorithm used in the token.

PASSporT header should include, at a minimum, the header parameters defined in the next three subsections.

3.1. "typ" (Type) Header Parameter

The "typ" (Type) Header Parameter is defined in JWS [RFC7515] Section 4.1.9. to declare the media type of the complete JWS.

For PASSporT Token the "typ" header MUST be the string "passport". This represents that the encoded token is a JWT of type passport.

3.2. "alg" (Algorithm) Header Parameter

The "alg" (Algorithm) Header Parameter is defined in JWS [RFC7515] Section 4.1.1. This definition includes the ability to specify the use of a cryptographic algorithm for the signature part of the JWS. It also refers to a list of defined "alg" values as part of a registry established by JSON Web Algorithms (JWA) [RFC7518] Section 3.1.

For the creation and verification of PASSporT tokens and their digital signatures, implementations MUST support ES256 as defined in JWA [RFC7518] Section 3.4. Implementations MAY support other algorithms registered in the JSON Web Signature and Encryption Algorithms registry created by [RFC7518]. The contents of that registry may be updated in the future depending on cryptographic strength requirements guided by current security best practice. The mandatory-to-support algorithm for PASSporT tokens may likewise be updated in future updates to this document.

3.3. "x5u" (X.509 URL) Header Parameter

As defined in JWS [RFC7515] Section 4.1.5., the "x5u" header parameter defines a URI [RFC3986] referring to the resource for the X.509 public key certificate or certificate chain [RFC5280] corresponding to the key used to digitally sign the JWS. Generally, as defined in JWS [RFC7515] section 4.1.5, this would correspond to an HTTPS or DNSSEC resource using integrity protection.

3.4. Example PASSporT header

An example of the header, would be the following, including the specified passport type, ES256 algorithm, and a URI referencing the network location of the certificate needed to validate the PASSporT signature.

```
{
    "typ":"passport",
    "alg":"ES256",
    "x5u":"https://cert.example.org/passport.cer"
}
```

4. PASSporT Payload

The token claims consist of the information which needs to be verified at the destination party. These claims follow the definition of a JWT claim [RFC7519] Section 4 and are encoded as defined by the JWS Payload [RFC7515] Section 3.

PASSporT defines the use of a standard JWT defined claim as well as custom claims corresponding to the two parties associated with personal communications, the originator and destination as detailed below.

Any claim key values outside the US-ASCII range should be encoded using percent encoding as described in Section 2.1 of [RFC3986], case normalized as described in 6.2.2.1 of [RFC3986].

4.1. JWT defined claims

4.1.1. "iat" - Issued At claim

The JSON claim MUST include the "iat" [RFC7519] Section 4.1.6 defined claim Issued At. As defined the "iat" should be set to the date and time of issuance of the JWT and MUST the origination of the personal communications. The time value should be of the format defined in [RFC7519] Section 2 NumericDate. This is included for securing the

token against replay and cut and paste attacks, as explained further in the security considerations in Section 9.

4.2. PASSporT specific claims

4.2.1. Originating and Destination Identity Claims

PASSporT defines claims that convey the identity of the origination and destination of personal communications. Origination in the context of PASSporT and for a given application's use of PASSporT is the point in the network that has the authority to assert the callers identity. This authority is represented in PASSporT by the certificate holder and is signed at the applications choice of authoritative point(s) in the network, for example, at a device that has authenticated with a user, or at a network entity with an authenticated trust relationship with that device and it's user. Destination represents the intended destination of the personal communications, i.e. the identity(s) being called by the caller. The destination point(s) determined by the application need to have the capability to verify the PASSporT token and the digital signature. The PASSporT associated certificate is used to validate the authority of the originating signer, generally via a certificate chain to the trust anchor for that application.

The origination and destination identities are represented by two claims that are required for PASSporT, the "orig" and "dest" claims. Both "orig" and "dest" MUST have claims where the key represents an identity type and the value is the identity string, both defined in subsequent subsections. Currently, these identities can be represented as either telephone numbers or Uniform Resource Indicators (URIs).

The "orig" JSON object MUST only have one key value pair representing the asserted identity of any type (currently either "tn" or "uri") of the originator of the personal communications signaling.

The "dest" JSON object MUST have at least have one key value pair, but could have multiple identity types (i.e. "tn" and/or "uri") but only one of each. If both "tn" and "uri" are included, the JSON object should list the "tn" array first and the "uri" array second. Within the "tn" and "uri" arrays, the identity strings should be put in lexicographical order including the scheme-specific portion of the URI characters. Additionally, in the case of "dest" only, the identity type key value MUST be an array signaled by standard JSON brackets, even when there is a single identity value in the identity type key value.

4.2.1.1. "tn" - Telephone Number identity

If the originating or destination identity is a telephone number, the key representing the identity MUST be "tn".

Telephone Number strings for "tn" MUST be canonicalized according to the procedures specified in [I-D.ietf-stir-rfc4474bis] Section 8.3.

4.2.1.2. "uri" - URI identity

If any of the originating or destination identities is of the form URI, as defined in [RFC3986], the key representing the identity MUST be "uri" URI form of the identity.

4.2.1.3. Future identity forms

We recognize that in the future there may be other standard mechanisms for representing identities. The "orig" and "dest" claims currently support "tn" and "uri" but could be extended in the future to allow for other identity types with new IANA registered unique types to represent these forms.

4.2.1.4. Examples

Single Originator, with telephone number identity +12155551212, to Single Destination, with URI identity 'sip:alice@example.com', example:

```
{
    "dest":{"uri":["sip:alice@example.com"]},
    "iat":"1443208345",
    "orig":{"tn":"12155551212"}
}
```

Single Originator, with telephone number identity +12155551212, to Multiple Destination Identities, with telephone number identity +12125551212 and two URI identities, sip:alice@example.com and sip:bob@example.com, example:

```
{
    "dest":{
    "tn":["12125551212"],
        "uri":["sip:alice@example.com",
            "sip:bob@example.net"]
    },
    "iat":"1443208345",
    "orig":{"tn":"12155551212"}
}
```

4.2.2. "mky" - Media Key claim

Some protocols that use PASSporT may also want to protect media security keys delivered within their signaling in order to bind those keys to the identities established in the signaling layers. The "mky" is an optional PASSporT claim defining the assertion of media key fingerprints carried in SDP [RFC4566] via the "a=fingerprint" attribute [RFC4572] Section 5. This claim can support either a single or multiple fingerprints appearing in a single SDP body corresponding to one or more media streams offered. The "mky" claim MUST be formatted in a JSON form including the "alg" and "dig" keys with the corresponding algorithm and hexadecimal values. If there is more than one fingerprint value associated with different media streams in SDP, the fingerprint values MUST be constructed as a JSON array denoted by bracket characters. For the "dig" key value, the hash value MUST be the hexadecimal value without any colons. The "mky" array MUST order the JSON objects containing both "alg" and "dig" key values in lexicographic order of the "alg" string first followed by the corresponding lexicographic order of the "dig" string values. Within each of those objects the JSON keys MUST have "alg" first and "dig" second.

An example claim with "mky" claim is as follows:

For an SDP offer that includes the following fingerprint values,

```
a=fingerprint:sha-256 4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:
5D:49:6B:19:E5:7C:AB:3E:4B:65:2E:7D:46:3F:54:42:CD:54:F1
a=fingerprint:sha-256 02:1A:CC:54:27:AB:EB:9C:53:3F:3E:4B:65
:2E:7D:46:3F:54:42:CD:54:F1:7A:03:A2:7D:F9:B0:7F:46:19:B2
```

the PASSporT Payload object would be:

```
{
    "dest":{"uri":["sip:alice@example.com"]},
    "iat":"1443208345",
    "mky":[
        {
           "alg":"sha-256",
           "dig":"021ACC5427ABEB9C533F3E4B652E7D463F5442CD54
                F17A03A27DF9B07F4619B2"
        },
        {
           "alg":"sha-256",
           "dig":"4AADB9B13F82183B540212DF3E5D496B19E57C
                AB3E4B652E7D463F5442CD54F1"
        }
      ],
    "orig":{"tn":"12155551212"}
}
```


5. PASSporT Signature

The signature of the PASSporT is created as specified by JWS [RFC7515] Section 5.1 Steps 1 through 6. PASSporT MUST use the JWS Protected Header. For the JWS Payload and the JWS Protected Header, the lexicographic ordering and white space rules described above, and JSON serialization rules in Section 8 of this document MUST be followed.

Appendix A of this document has a detailed example of how to follow the steps to create the JWS Signature.

JWS [RFC7515] Section 5.1 Step 7 JWS JSON serialization is not supported for PASSporT.

JWS [RFC7515] Section 5.1 Step 8 describes the method to create the final JWS Compact Serialization form of the PASSporT Token.

6. Compact form of PASSporT

For a using protocol of PASSporT, the PASSporT Claims as well as the PASSporT Header may include redundant or default information that could be reconstructed at the destination based on information provided in the signaling protocol transporting the PASSporT object. In this case, it may be advantageous to have a more compact form of PASSporT to save the transmission of the bytes needed to represent the header and claims.

We define the compact form of the PASSporT token, in the spirit of form defined in [RFC7515] Appendix F, with the use of '..', two

periods to represent the header and claim objects being removed, followed by PASSporT signature as defined in Section 5, and the need for the destination to reconstruct the header and claim objects in order to verify the signature.

In order to construct the Compact form of the PASSporT string, the procedure described in Section 5 with the exception of Step 8 described in JWS [RFC7515] Section 5.1. This step would be replaced by the following construction of the compact form of PASSporT, '..' || BASE64URL(JWS Signature).

The using protocol of the compact form of PASSporT MUST be accompanied by a specification for how the header and claims objects can be reconstructed from information in the signaling protocol being used.

6.1. Example Compact form PASSporT Token

The compact form of the following example token (with line breaks between period used for readability purposes only)


```
eyJhbGciOiJFUzI1NiIsInR5cCI6InBhc3Nwb3J0IiwieDV1IjoiaHR0cHM6Ly9j
ZXJ0LmV4YW1wbGUub3JnL3Bhc3Nwb3J0LmNlciJ9
.
eyJkZXN0Ijp7InVyaSI6WyJzaXA6YWxpY2VAZXhhbXBsZS5jb20iXX0sImlhdCI
6IjE0NDMyMDgzNDUiLCJvcmlnIjp7InRuIjoiMTIxNTU1NTEyMTIifX0
.
rq3pjT1hoRwakEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjpjlk-cpFYpFYsojN
CpTzO3QfPOlckGaS6hEck7w
```


would be as follows (with line breaks between period used for readability purposes only)


```
..rq3pjT1hoRwakEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjpjlk-cpFYpFYsojN
CpTzO3QfPOlckGaS6hEck7w
```


7. Extending PASSporT

PASSporT includes the bare minimum set of claims needed to securely assert the originating identity and support the secure properties discussed in various parts of this document. JWT supports a straight forward way to add additional claims by simply adding new claim key pairs. PASSporT can be extended beyond the defined base set of claims to represent other information requiring assertion or validation beyond the originating identity itself as needed.

7.1. "ppt" (PASSporT) header parameter

Any using protocol can extend the payload of PASSporT with additional JWT claims. JWT claims are managed by an existing IANA registry as defined in [RFC7519] Section 10.1. Implementations of PASSporT MUST support the baseline claims defined in Section 4.2, and MAY support extended claims. If it is necessary for an extension to PASSporT to require that a relying party support a particular extended claim or set of claims in the PASSporT object, it can do so by specifying a "ppt" element for the PASSporT JOSE header. All values of "ppt" need to be defined in a specification which associates the new value of the "ppt" element with the required claims and behaviors. Relying parties MUST fail to validate PASSporT objects containing an unsupported "ppt".

Using protocols that carry the compact form of PASSporT, defined in Section 6, instead of the full form MUST use only mandatory extensions signaled with "ppt" - if a using protocol were to add additional optional claims to a PASSporT object it carried in compact form, relying parties would have no way to reconstruct the token. Moreover, using protocols that support the compact form of PASSporT MUST have some field to signal "ppt" to relying parties, as the compact form of PASSporT omits the JOSE header.

7.2. Example extended PASSporT header

An example header with a PASSporT extension type of "foo" is as follows:

```
{
    "alg":"ES256",
    "ppt":"foo",
    "typ":"passport",
    "x5u":"https://tel.example.org/passport.cer"
}
```

7.3. Extended PASSporT Claims

Specifications that define extensions to the PASSporT mechanism MUST explicitly specify what claims they include beyond the base set of claims from this document, the order in which they will appear, and any further information necessary to implement the extension. All extensions MUST include the baseline PASSporT claim elements specified in Section 4; claims may only be appended to the claims object specified; they can never be removed or re-ordered. Specifying new claims follows the baseline JWT procedures ([RFC7519] Section 10.1). Understanding an extension or new claims defined by the extension on the destination verification of the PASSporT token

is optional. The creator of a PASSporT object cannot assume that destination systems will understand any given extension. Verification of PASSporT tokens by destination systems that do support an extension may then trigger appropriate application-level behavior in the presence of an extension; authors of extensions should provide appropriate extension-specific guidance to application developers on this point.

An example set of extended claims, extending the first example in Section 4.2.1.4 using "bar" as the newly defined claim would be as follows:

```
{
    "bar":"beyond all recognition"
    "dest":{"uri":["sip:alice@example.com"]},
    "iat":"1443208345",
    "orig":{"tn":"12155551212"}
}
```

8. Deterministic JSON Serialization

JSON, as a canonical format, can include spaces and line breaks, and key value pairs can occur in any order. It is therefore a non-deterministic string format. In order to make the digital signature verification work deterministically, the JSON representation of the JWS Protected Header object and JWS Payload object MUST be computed as follows.

The JSON object MUST follow the rules for the construction of the thumbprint of a JSON Web Key (JWK) as defined in [RFC7638] Section 3 Step 1 only. Step 2 should not be performed; as noted in JWK this is still a legal JWK object.

The PASSporT header and claim direct members MUST follow the lexicographical ordering rules. Any top level JSON members that contain JSON objects or arrays, such as "dest" or "mky" MUST follow their own lexicographical ordering and whitespace and line break rules for the sub-elements. This includes any header or claims defined in future specifications using PASSporT.

8.1. Example PASSport deterministic JSON form

This section demonstrate the deterministic JSON serialization for the example PASSporT Payload shown in Section 4.2.1.4.

The initial JSON object is shown here:

```
{
    "dest":{"uri":["sip:alice@example.com"]},
    "orig":{"tn":"12155551212"}
    "iat":"1443208345",
    "mky":[
    {
       "alg":"sha-256",
       "dig":"021ACC5427ABEB9C533F3E4B652E7D463F5442CD54
        F17A03A27DF9B07F4619B2"
    },
    {
       "alg":"sha-256",
       "dig":"4AADB9B13F82183B540212DF3E5D496B19E57C
        AB3E4B652E7D463F5442CD54F1"
    }
  ],
}
```


The parent members of the JSON object are as follows:

- "dest"
- "orig"
- "iat"
- "mky"

Their lexicographic order is:

- "dest"
- "iat"
- "mky"
- "orig"

The final constructed deterministic JSON serialization representation, with whitespace and line breaks removed, (with line breaks used for display purposes only) is:


```
{"dest":{"uri":["sip:alice@example.com"],"iat":1443208345,"mky":
  [{"alg":"sha-256","dig":"021ACC5427ABEB9C533F3E4B652E7D463F5442CD5
  4F17A03A27DF9B07F4619B2"},{"alg":"sha-256","dig":"4AADB9B13F82183B5
  40212DF3E5D496B19E57CAB3E4B652E7D463F5442CD54F1"}],
  "orig":{"tn":"12155551212"}}
```

9. Security Considerations

9.1. Avoidance of replay and cut and paste attacks

There are a number of security considerations for use of the token for avoidance of replay and cut and paste attacks. PASSporT tokens SHOULD only be sent with application level protocol information (e.g. for SIP an INVITE as defined in [RFC3261]) corresponding to the required fields in the token. A uniqueness of the set of token claims and token signature is constructed using the originating identity being asserted with the 'orig' claim along with the the following two claims:

- 'iat' claim should correspond to a date/time the message was originated. It should also be within a relative time that is reasonable for clock drift and transmission time characteristics associated with the application using the PASSporT token. Therefore, validation of the token should consider date and time correlation, which could be influenced by signaling protocol specific use and network time differences.

- 'dest' claim is included to prevent the valid re-use of a previously originated message to send to another destination party.

9.2. Solution Considerations

The use of PASSporT tokens based on the validation of the digital signature and the associated certificate requires consideration of the authentication and authority or reputation of the signer to attest to the identity being asserted. The following considerations should be recognized when using PASSporT: * The use of this token should not, in it's own right, be considered a full solution for absolute non-repudiation of the identity being asserted. * In many applications, the end user the asserted identity represents and signer may not be one in the same. For example, when a service provider signs and validates the token on the behalf of the user consuming the service, the provider MUST have an authenticated and secure relationship with the end user or the device initiating and terminating the communications signaling. * Applications that use PASSporT should ensure the verification of the signature includes the means of verifying the signer is authoritative through the use of an application or service specific set of common trust anchors for the application.

10. IANA Considerations

10.1. Media Type Registration

10.1.1. Media Type Registry Contents Additions Requested

This section registers the "application/passport" media type [RFC2046] in the "Media Types" registry in the manner described in [RFC6838], which can be used to indicate that the content is a PASSporT defined JWT and JWS.

- Type name: application

- Subtype name: passport

- Required parameters: n/a

- Optional parameters: n/a

- Encoding considerations: 8bit; application/passport values outside the US-ASCII range are encoded using percent encoding as described in Section 2.1 of [RFC3986] (some values may be the empty string), each separated from the next by a single period ('.') character.

- Security considerations: See the Security Considerations Section of [RFC7515].

- Interoperability considerations: n/a

- Published specification: [RFCThis]

- Applications that use this media type: STIR and other applications that require identity related assertion

- Fragment identifier considerations: n/a

- Additional information:

  Magic number(s): n/a File extension(s): n/a Macintosh file type code(s): n/a

- Person & email address to contact for further information: Chris Wendt, chris-ietf@chriswendt.net

- Intended usage: COMMON

- Restrictions on usage: none

- o Author: Chris Wendt, chris-ietf@chriswendt.net
- o Change Controller: IESG
- o Provisional registration? No

10.2. JSON Web Token Claims Registration

10.2.1. Registry Contents Additions Requested

- o Claim Name: "orig"
- o Claim Description: Originating Identity String
- o Change Controller: IESG
- o Specification Document(s): Section 4.2.1 of [RFCThis]
- o Claim Name: "dest"
- o Claim Description: Destination Identity String
- o Change Controller: IESG
- o Specification Document(s): Section 4.2.1 of [RFCThis]
- o Claim Name: "mky"
- o Claim Description: Media Key Fingerprint String
- o Change Controller: IESG
- o Specification Document(s): Section 4.2.2 of [RFCThis]

10.3. JSON Web Signature and Encryption Header Parameter Registry

10.3.1. Registry Contents Additions Requested

Header Parameter Name: "ppt"

- o Header Parameter Description: PASSporT extension identifier
- o Header Parameter Usage Location(s): JWS
- o Change Controller: IESG
- o Specification Document(s): Section 7.1 of [RFCThis]

11. Acknowledgements

Particular thanks to members of the ATIS and SIP Forum NNI Task Group including Jim McEchern, Martin Dolly, Richard Shockey, John Barnhill, Christer Holmberg, Victor Pascual Avila, Mary Barnes, Eric Burger for their review, ideas, and contributions also thanks to Henning Schulzrinne, Russ Housley, Alan Johnston, Richard Barnes, Mark Miller, Ted Hardie, Dave Crocker, Robert Sparks for valuable feedback on the technical and security aspects of the document. Additional thanks to Harsha Bellur for assistance in coding the example tokens.

12. References

12.1. Normative References

[I-D.ietf-stir-rfc4474bis]
Peterson, J., Jennings, C., Rescorla, E., and C. Wendt, "Authenticated Identity Management in the Session Initiation Protocol (SIP)", draft-ietf-stir-rfc4474bis-12 (work in progress), September 2016.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <http://www.rfc-editor.org/info/rfc2046>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <http://www.rfc-editor.org/info/rfc4566>.

[RFC4572] Lennox, J., "Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)", RFC 4572, DOI 10.17487/RFC4572, July 2006, <http://www.rfc-editor.org/info/rfc4572>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

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

[RFC7518] Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, DOI 10.17487/RFC7518, May 2015, <http://www.rfc-editor.org/info/rfc7518>.

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

[RFC7638] Jones, M. and N. Sakimura, "JSON Web Key (JWK) Thumbprint", RFC 7638, DOI 10.17487/RFC7638, September 2015, <http://www.rfc-editor.org/info/rfc7638>.

12.2. Informative References

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC7340] Peterson, J., Schulzrinne, H., and H. Tschofenig, "Secure Telephone Identity Problem Statement and Requirements", RFC 7340, DOI 10.17487/RFC7340, September 2014, <http://www.rfc-editor.org/info/rfc7340>.

Appendix A. Example ES256 based PASSporT JWS Serialization and Signature

For PASSporT, there will always be a JWS with the following members:

- "protected", with the value BASE64URL(UTF8(JWS Protected Header))

- "payload", with the value BASE64URL (JWS Payload)

- "signature", with the value BASE64URL(JWS Signature)

This example will follow the steps in JWS [RFC7515] Section 5.1, steps 1-6 and 8 and incorporates the additional serialization steps required for PASSporT.

Step 1 for JWS references the JWS Payload, an example PASSporT Payload is as follows:

```
{
    "dest":{"uri":["sip:alice@example.com"]}
    "iat":"1443208345",
    "orig":{"tn":"12155551212"}
}
```

This would be serialized to the form (with line break used for display purposes only):

```
{"dest":{"uri":["sip:alice@example.com"]},"iat":"1443208345",
  "orig":{"tn":"12155551212"}}
```

Step 2 Computes the BASE64URL(JWS Payload) producing this value (with line break used for display purposes only):

eyJkZXN0Ijp7InVyaSI6WyJzaXA6YWxpY2VAZXhhbXBsZS5jb20iXX0sImlhdCI
6IjE0NDMyMDgzNDUiLCJvcmlnIjp7InRuIjoiMTIxNTU1NTEyMTIifX0

For Step 3, an example PASSporT Protected Header comprising the JOSE Header is as follows:

```
{
    "alg":"ES256",
    "typ":"passport",
    "x5u":"https://cert.example.org/passport.cer"
}
```

This would be serialized to the form (with line break used for display purposes only):

```
{"alg":"ES256","typ":"passport","x5u":"https://cert.example.org
    /passport.cer"}
```

Step 4 Performs the BASE64URL(UTF8(JWS Protected Header)) operation and encoding produces this value (with line break used for display purposes only):

eyJhbGciOiJFUzI1NiIsInR5cCI6InBhc3Nwb3J0IiwieDV1IjoiaHR0cHM6Ly9j
ZXJ0LmV4YW1wbGUub3JnL3Bhc3Nwb3J0LmNlciJ9

Step 5 and Step 6 performs the computation of the digital signature of the PASSporT Signing Input ASCII(BASE64URL(UTF8(JWS Protected Header)) || '.' || BASE64URL(JWS Payload)) using ES256 as the algorithm and the BASE64URL(JWS Signature).

rq3pjT1hoRwakEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjpjlk-cpFYpFYsojN
CpTzO3QfPOlckGaS6hEck7w

Step 8 describes how to create the final PASSporT token, concatenating the values in the order Header.Payload.Signature with period ('.') characters. For the above example values this would produce the following (with line breaks between period used for readability purposes only):

eyJhbGciOiJFUzI1NiIsInR5cCI6InBhc3Nwb3J0IiwieDV1IjoiaHR0cHM6Ly9j
ZXJ0LmV4YW1wbGUub3JnL3Bhc3Nwb3J0LmNlciJ9
.
eyJkZXN0Ijp7InVyaSI6WyJzaXA6YWxpY2VAZXhhbXBsZS5jb20iXX0sImlhdCI
6IjE0NDMyMDgzNDUiLCJvcmlnIjp7InRuIjoiMTIxNTU1NTEyMTIifX0
.
rq3pjT1hoRwakEGjHCnWSwUnshd0-zJ6F1VOgFWSjHBr8Qjpjlk-cpFYpFYsojN
CpTzO3QfPOlckGaS6hEck7w

A.1. X.509 Private Key for ES256 Example**

-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFeZ1R208QCvcu5GuYyMfG4W7sH4m99/7eHSDLpdYllFoAoGCCqGSM49
AwEHoUQDQgAE8HNbQd/TmvCKwPKHkMF9fScavGeH78YTU8qLS8I5HLHSSmlATLcs
lQMhNC/OhlWBYC626nIlo7XeebYS7Sb37g==
-----END EC PRIVATE KEY-----

A.2. X.509 Public Key for ES256 Example**

-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8HNbQd/TmvCKwPKHkMF9fScavGeH
78YTU8qLS8I5HLHSSmlATLcslQMhNC/OhlWBYC626nIlo7XeebYS7Sb37g==
-----END PUBLIC KEY-----

Authors' Addresses

Chris Wendt
Comcast
One Comcast Center
Philadelphia, PA 19103
USA

Email: chris-ietf@chriswendt.net

Jon Peterson
Neustar Inc.
1800 Sutter St Suite 570
Concord, CA 94520
US

Email: jon.peterson@neustar.biz