

ADD WG  
Internet-Draft  
Intended status: Standards Track  
Expires: April 10, 2022

T. Reddy  
Akamai  
D. Wing  
Citrix  
M. Richardson  
Sandelman Software Works  
M. Boucadair  
Orange  
October 7, 2021

DNS Server Selection: DNS Server Information with Assertion Token  
draft-reddy-add-server-policy-selection-09

## Abstract

The document defines a mechanism that is meant to communicate DNS resolver information to DNS clients for use as a criteria for server selection decisions. Such an information that is cryptographically signed to attest its authenticity is used for the selection of DNS resolvers. Typically, evaluating the resolver information and the signatory, DNS clients with minimal or no human intervention can select the DNS servers for resolving domain names.

This assertion is useful for encrypted DNS (e.g., DNS-over-TLS, DNS-over-HTTPS, or DNS-over-QUIC) servers that are either public resolvers or discovered in a local network.

## 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 April 10, 2022.

---

Internet-Draft      DNS Server Info with Assertion Token      October 2021

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">2.</a>	Terminology . . . . .	<a href="#">3</a>
<a href="#">3.</a>	Resolver Assertion Token (REAT): Overview . . . . .	<a href="#">4</a>
<a href="#">4.</a>	REAT Header . . . . .	<a href="#">6</a>
<a href="#">4.1.</a>	'typ' (Type) Header Parameter . . . . .	<a href="#">6</a>
<a href="#">4.2.</a>	'alg' (Algorithm) Header Parameter . . . . .	<a href="#">6</a>
<a href="#">4.3.</a>	'x5u' (X.509 URL) Header Parameter . . . . .	<a href="#">6</a>
<a href="#">4.4.</a>	An Example of REAT Header . . . . .	<a href="#">7</a>
<a href="#">5.</a>	REAT Payload . . . . .	<a href="#">7</a>
<a href="#">5.1.</a>	JWT Defined Claims . . . . .	<a href="#">7</a>
<a href="#">5.1.1.</a>	'iat' - Issued At Claim . . . . .	<a href="#">7</a>
<a href="#">5.1.2.</a>	'exp' - Expiration Time Claim . . . . .	<a href="#">7</a>
<a href="#">5.2.</a>	REAT Specific Claims . . . . .	<a href="#">8</a>
<a href="#">5.2.1.</a>	DNS Server Identity Claims . . . . .	<a href="#">8</a>
<a href="#">5.2.2.</a>	'resinfo' (Resolver Information) Claim . . . . .	<a href="#">8</a>
<a href="#">5.2.3.</a>	An Example . . . . .	<a href="#">9</a>
<a href="#">6.</a>	REAT Signature . . . . .	<a href="#">9</a>
<a href="#">7.</a>	Extending REAT . . . . .	<a href="#">10</a>
<a href="#">8.</a>	Deterministic JSON Serialization . . . . .	<a href="#">10</a>
<a href="#">8.1.</a>	Example REAT Deterministic JSON Form . . . . .	<a href="#">11</a>
<a href="#">9.</a>	Using RESINFO Responses . . . . .	<a href="#">11</a>
<a href="#">10.</a>	Security Considerations . . . . .	<a href="#">12</a>
<a href="#">11.</a>	IANA Considerations . . . . .	<a href="#">12</a>
<a href="#">11.1.</a>	Media Type Registration . . . . .	<a href="#">12</a>
<a href="#">11.1.1.</a>	Media Type Registry Contents Additions Requested . . . . .	<a href="#">12</a>
<a href="#">11.2.</a>	JSON Web Token Claims Registration . . . . .	<a href="#">13</a>

<a href="#">11.2.1.</a>	Registry Contents Additions Requested . . . . .	<a href="#">13</a>
<a href="#">11.3.</a>	DNS Resolver Information Registration . . . . .	<a href="#">14</a>
<a href="#">12.</a>	Acknowledgments . . . . .	<a href="#">14</a>
<a href="#">13.</a>	References . . . . .	<a href="#">14</a>
<a href="#">13.1.</a>	Normative References . . . . .	<a href="#">14</a>

<a href="#">13.2.</a>	Informative References . . . . .	<a href="#">16</a>
<a href="#">Appendix A.</a>	Example of ES256-based REAT JWS Serialization and Signature . . . . .	<a href="#">18</a>
A.1.	X.509 Private Key in PKCS#8 Format for ES256 Example** .	<a href="#">20</a>
<a href="#">A.2.</a>	X.509 Public Key for ES256 Example** . . . . .	<a href="#">20</a>
<a href="#">Appendix B.</a>	Complete JWS JSON Serialization Representation with multiple Signatures . . . . .	<a href="#">20</a>
<a href="#">B.1.</a>	X.509 Private Key in PKCS#8 format for E384 Example** . .	<a href="#">21</a>
<a href="#">B.2.</a>	X.509 Public Key for ES384 Example** . . . . .	<a href="#">21</a>
Authors' Addresses	. . . . .	<a href="#">21</a>

## [1.](#) Introduction

[RFC7626] discusses DNS privacy considerations in both "on the wire" ([Section 2.4 of \[RFC7626\]](#)) and "in the server" ([Section 2.5 of \[RFC7626\]](#)) contexts. Examples of protocols that provide encrypted channels between DNS clients and servers are DNS-over-HTTPS (DoH) [[RFC8484](#)], DNS-over-TLS (DoT) [[RFC7858](#)], and DNS-over-QUIC (DoQ) [[I-D.ietf-dprive-dnsquic](#)].

DNS clients can discover and authenticate encrypted DNS servers provided by a local network, for example using the techniques proposed in [[I-D.ietf-add-dnr](#)] and [[I-D.ietf-add-ddr](#)]. If the mechanism used to discover the encrypted DNS server is insecure, the DNS client needs evidence about the encrypted server to assess its trustworthiness and a way to appraise such evidence. The mechanism specified in this document can be used by the DNS client to cryptographically identify if it is connecting to an encrypted DNS server hosted by a specific organization (e.g., ISP or Enterprise). This strengthens the protection as clients can detect and reject connections to encrypted DNS servers hosted by attackers.

## [2.](#) Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" 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.

This document makes use of the terms defined in [[RFC8499](#)] and [[I-D.ietf-dnsop-terminology-ter](#)].

'Encrypted DNS' refers to a DNS protocol that provides an encrypted channel between a DNS client and server (e.g., DoT, DoH, or DoQ).

The terms 'Evidence', 'Verifier', 'Background Check', 'Relying Party', 'Appraisal Policy', and 'Attestation Results' are defined in [[I-D.ietf-rats-architecture](#)].

### [3.](#) Resolver Assertion Token (REAT): Overview

The mechanism used in this specification resembles the Background-Check Model discussed in Sections [5.2](#) and [5.3](#) of Remote attestation procedure (RATS) Architecture [[I-D.ietf-rats-architecture](#)]. RATS enables a relying party to establish a level of confidence in the trustworthiness of a remote peer through the creation of Evidence to assess the peer's trustworthiness, and an Appraisal Policy for such Evidence.

In this document, the Relying Party is the DNS client and the Attester is the encrypted DNS server. The Encrypted DNS servers MAY use "Domain Validation" (DV) certificates for certificate-based server authentication in TLS connections.

The DNS server's resolver information needs to be validated and signed. This signature is called an Attestation Result [[I-D.ietf-rats-architecture](#)]. This validation can be performed by the DNS operator itself (signed by the DNS operator's certificate) acting as a verifier or performed by an external Verifier (signed by that external Verifier). The signing certificate can to be an Extended Validation (EV) certificate issued by a public CA in specific scenarios listed below. An EV certificate is issued by the public CA after a thorough Background Check to verify the requesting organization's legal identity. If the signing certificate is a EV

certificate, it leaves the client with a better audit trail of the organization hosting the DNS server in comparison with the DV certificate.

The use of EV certificate is needed in the following scenarios:

- o It helps the client to avoid sending DNS queries to an Encrypted DNS server hosted by an attacker discovered insecurely (e.g., using DHCP/RA or DNS). For example, an attacker can get a domain name, domain-validated public certificate from a CA and host a Encrypted DNS server. Furthermore, an attacker can use a public IP address, get an 'IP address'-validated public certificate from a CA and host a Encrypted DNS server.
- o It can be used by the client to identify the Encrypted DNS server is hosted by a legal organization.

The use of EV certificate is not required in the following scenarios:

- o If the Encrypted DNS server can only be discovered securely (e.g., using IKEv2 [[I-D.btw-add-ipsecme-ike](#)]), the signing certificate need not be an EV certificate.
- o Secure Zero Touch Provisioning [[RFC8572](#)] defines a bootstrapping strategy for enabling a networking device to securely obtain the required configuration information with no user input. If the encrypted DNS server is insecurely discovered and not preconfigured in the networking device, the DNS client on the networking device can validate the Resolver Assertion Token signature using the owner certificate as per [Section 3.2 of \[RFC8572\]](#).

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 accommodate asserted resolver information of the Encrypted DNS server. 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. The next sections define the header and claims that MUST be minimally used with JWT and JWS for resolver assertion token.

The REsolver Assertion Token (REAT) specifically uses this token format and defines claims that convey the resolver information of Encrypted DNS server.

The client can retrieve the REAT object using the RESINFO RRtype defined in [[I-D.reddy-add-resolver-info](#)] and QNAME of the domain name that is used to authenticate the DNS server (referred to as ADN in [[RFC8310](#)]). If the special use domain name "resolver.arpa" defined in [[I-D.ietf-add-ddr](#)] is used to discover the Encrypted DNS server, the client can retrieve the REAT object using the RESINFO RRtype and QNAME of the special use domain name.

The signature of REAT object MUST be validated by the DNS client. If signature is invalid, the REAT object is rejected. If signature is valid and signer is trusted, the DNS client can use that encrypted DNS server.

## [4.](#) REAT Header

The JWS token header is a JOSE header ([Section 4 of \[RFC7515\]](#)) that defines the type and encryption algorithm used in the token.

The REAT header MUST include, at a minimum, the header parameters defined in [Sections 4.1](#), [4.2](#), and [4.3](#).

### [4.1.](#) 'typ' (Type) Header Parameter

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

For REAT Token the 'typ' header MUST be the string 'rat'. This represents that the encoded token is a JWT of type rat.

## [4.2.](#) 'alg' (Algorithm) Header Parameter

The 'alg' (Algorithm) Header Parameter is defined in [Section 4.1.1 of \[RFC7515\]](#). It specifies the JWS signature cryptographic algorithm. 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 REAT tokens and their digital signatures, implementations MUST support ES256 as defined in [Section 3.4 of \[RFC7518\]](#). Implementations MAY support other algorithms registered in the JSON Web Signature and Encryption Algorithms registry created by [\[RFC7518\]](#). The content 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 REAT tokens may likewise be updated in the future.

Implementations of REAT digital signatures using ES256 as defined above SHOULD use deterministic ECDSA when supported for the reasons stated in [\[RFC6979\]](#).

## [4.3.](#) 'x5u' (X.509 URL) Header Parameter

As defined in [Section 4.1.5 of \[RFC7515\]](#), 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 [Section 4.1.5 of \[RFC7515\]](#) this corresponds to an HTTPS or DNSSEC resource using integrity protection.

## [4.4.](#) An Example of REAT Header

An example of the REAT header is shown in Figure 1. It includes the specified REAT type, ES256 algorithm, and an URI referencing the network location of the certificate needed to validate the REAT signature.

```
{
```

```
"typ":"rat",  
"alg":"ES256",  
"x5u":"https://cert.example.com/rat.cer"  
}
```

Figure 1: A REAT Header Example

## 5. REAT Payload

The token claims consist of the resolver information of the DNS server that needs to be verified at the DNS client. These claims follow the definition of a JWT claim ([Section 4 of \[RFC7519\]](#)) and are encoded as defined by the JWS Payload ([Section 3 of \[RFC7515\]](#)).

REAT defines the use of a standard JWT-defined claim as well as custom claims corresponding to the DoT or DoH servers.

Claim names MUST use the US-ASCII character set. Claim values MAY contain characters that are outside the ASCII range, however they MUST follow the default JSON serialization defined in [Section 7 of \[RFC7519\]](#).

### 5.1. JWT Defined Claims

#### 5.1.1. 'iat' - Issued At Claim

The JSON claim MUST include the 'iat' ([Section 4.1.6 of \[RFC7519\]](#)) defined claim "Issued At". The 'iat' should be set to the date and time of issuance of the JWT. The time value should be of the format (NumericDate) defined in [Section 2 of \[RFC7519\]](#).

#### 5.1.2. 'exp' - Expiration Time Claim

The JSON claim MUST include the 'exp' ([Section 4.1.4 of \[RFC7519\]](#)) defined "claim Expiration Time". The 'exp' should be set to specify the expiration time on or after which the JWT is not accepted for processing. The REAT object should expire after a reasonable duration. A short expiration time for the REAT object periodically reaffirms the resolver information of the DNS server to the DNS client and ensures the DNS client does not use outdated resolver

information. If the DNS client knows the REAT object has expired, it

should make another request to get the new REAT object from the DNS server.

## [5.2.](#) REAT Specific Claims

### [5.2.1.](#) DNS Server Identity Claims

The DNS server identity is represented by a claim that is required for REAT: the 'server' claim. The 'server' MUST contain claim values that are identity claim JSON objects where the child claim name represents an identity type and the claim value is the identity string, both defined in subsequent subsections.

These identities can be represented as either authentication domain name (ADN) (defined in [[RFC8310](#)]) or Uniform Resource Indicators (URI).

The DNS client constructs a reference identifier for the DNS server based on the ADN or the domain portion in the URI of the DNS server identity. The domain name in the DNS-ID identifier type within subjectAltName entry in the DNS server certificate conveyed in the TLS handshake is matched with the reference identifier. If the match is not successful, the client MUST not accept the REAT for further processing.

#### [5.2.1.1.](#) 'adn' - Authentication Domain Name Identity

If the DNS server identity is an ADN, the claim name representing the identity MUST be 'adn'. The claim value for the 'adn' claim is the ADN.

#### [5.2.1.2.](#) 'uri' - URI Identity

If the DNS server identity is of the form URI Template, as defined in [[RFC6570](#)], the claim name representing the identity MUST be 'uri' and the claim value is the URI Template form of the DNS server identity.

As a reminder, if DoH is supported by the DNS server, the DNS client uses the URI Template ([Section 3 of \[RFC8484\]](#)).

### [5.2.2.](#) 'resinfo' (Resolver Information) Claim

The 'resinfo' claim contains the resolver information of the DNS server defined in Section 5 of [[I-D.reddy-add-resolver-info](#)].

### [5.2.3.](#) An Example

Figure 2 shows an example of resolver information.

```
{
  "server":{
    "adn":"example.com"
  },
  "iat":1443208345,
  "exp":1443640345,
  "resinfo": {
    "qnameminimization":false,
  }
}
```

Figure 2: An Example of Resolver Information

## [6.](#) REAT Signature

The signature of the REAT is created as specified in [Section 5.1 of \[RFC7515\]](#) (Steps 1 through 6). REAT MUST use the JWS Protected Header.

For the JWS Payload and the JWS Protected Header, the lexicographic ordering and white space rules described in [Section 4](#) and [Section 5](#), and JSON serialization rules in [Section 8](#) MUST be followed.

The REAT is cryptographically signed by the domain hosting the DNS server and optionally by a third party who performed privacy and security audit of the DNS server.

The resolver information is attested using "Extended Validation" (EV) certificate to avoid bad actors taking advantage of this mechanism to advertise encrypted DNS servers for illegitimate and fraudulent purposes meant to trick DNS clients into believing that they are using a legitimate encrypted DNS server hosted to provide privacy for DNS transactions.

Alternatively, a DNS client has to be configured to trust the leaf of the signer of the REAT object. That is, trust of the signer MUST NOT be determined by validating the signer via the OS or the browser trust chain because that would allow any arbitrary entity to operate a DNS server and assert any sort of resolver information.

[Appendix A](#) provides an example of how to follow the steps to create the JWS Signature.

JWS JSON serialization (Step 7 in [Section 5.1 of \[RFC7515\]](#)) is supported for REAT to enable multiple signatures to be applied to the REAT object. For example, the REAT object can be cryptographically signed by the domain hosting the DNS server and by a third party who performed privacy and security audit of the DNS server.

[Appendix B](#) includes an example of the full JWS JSON serialization representation with multiple signatures.

[Section 5.1 of \[RFC7515\]](#) (Step 8) describes the method to create the final JWS Compact Serialization form of the REAT Token.

## [7.](#) Extending REAT

REAT includes the minimum set of claims needed to securely assert the resolver information of the DNS server. JWT supports a mechanism to add additional asserted or signed information by simply adding new claims. REAT can be extended beyond the defined base set of claims to represent other DNS server information requiring assertion or validation. Specifying new claims follows the baseline JWT procedures ([Section 10.1 of \[RFC7519\]](#)). Understanding new claims on the DNS client is optional. The creator of a REAT object cannot assume that the DNS client will understand the new claims.

## [8.](#) Deterministic JSON Serialization

JSON objects 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 following rules. These rules are based on the thumbprint of a JSON Web Key (JWK) as defined in [Section 3 of \[RFC7638\]](#) (Step 1).

1. The JSON object MUST contain no whitespace or line breaks before or after any syntactic elements.

2. JSON objects MUST have the keys ordered lexicographically by the Unicode [[UNICODE](#)] code points of the member names.
3. JSON value literals MUST be lowercase.
4. JSON numbers are to be encoded as integers unless the field is defined to be encoded otherwise.

5. Encoding rules MUST be applied recursively to member values and array values.

#### [8.1](#). Example REAT Deterministic JSON Form

This section demonstrates the deterministic JSON serialization for the example REAT Payload shown in [Section 5.2.3](#).

The initial JSON object is shown in Figure 3.

```
{
  "server":{
    "adn":"example.com"
  },
  "iat":1443208345,
  "exp":1443640345,
  "resinfo": {
    "qnameminimization":false,
  }
}
```

Figure 3: Initial JSON Object

The parent members of the JSON object are as follows, in lexicographic order: "exp", "iat", "resinfo", "server".

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

```
{"exp":1443640345,"iat":1443208345,
"resinfo":{"qnameminimization":false},
```

```
"server":{"adn":"example.com"}}}
```

Figure 4: Deterministic JSON Form

## [9.](#) Using RESINFO Responses

This document defines the following entry for the IANA DNS Resolver Information Registry that is defined in [I-D.reddy-add-resolver-info].

- o The "attested-resinfo" name contains the full REAT object. The REAT header, REAT payload, and REAT signature components comprise a full REAT object. If the "attested-resinfo" name is conveyed to the client, the server need not convey the attributes "resinfourl", "identityurl", "extendeddnserver" and

"qnameminimization" attributes separately as that resolver information will be extracted by the client from the REAT payload.

## [10.](#) Security Considerations

The use of REAT object 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 the resolver information of the DNS server being asserted. Bad actors can host encrypted DNS servers to invade the privacy of the user. Bad actor can get a domain name, host encrypted DNS servers, and get the DNS server certificate signed by a CA. The resolver information will have to be attested using EV certificates or a REAT object signer trusted by the DNS client to prevent the attack.

The CA that issued the EV certificate does not attest the resolver information. The organization hosting the DNS server attests the resolver information using the EV certificate and the client uses the EV certificate to identify the organization (e.g., ISP or Enterprise) hosting the DNS server.

If the REAT object is asserted by a third party, it can do a "time of check" but the DNS server is susceptible of "time of use" attack. For example, changes to the DNS server can cause a disagreement

between the auditor and the DNS server operation, hence the REAT object needs to be also asserted by the domain hosting the DNS server. In addition, the REAT object needs to have a short expiration time (e.g., 7 days) to ensure the DNS server's domain re-asserts the resolver information and limits the damage from change in behaviour and mis-issuance.

## [11.](#) IANA Considerations

### [11.1.](#) Media Type Registration

#### [11.1.1.](#) Media Type Registry Contents Additions Requested

This section registers the 'application/rat' 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 REAT defined JWT.

- o Type name: application
- o Subtype name: rat
- o Required parameters: n/a

- o Optional parameters: n/a
- o Encoding considerations: 8bit; application/rat values are encoded as a series of base64url-encoded values (some of which may be the empty string) separated by period ('.') characters..
- o Security considerations: See the Security Considerations Section of [[RFC7515](#)].
- o Interoperability considerations: n/a
- o Published specification: [THIS\_DOCUMENT]
- o Applications that use this media type: DNS
- o Fragment identifier considerations: n/a
- o Additional information:

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

- o Person & email address to contact for further information: Tirumaleswar Reddy, kondtir@gmail.com
- o Intended usage: COMMON
- o Restrictions on usage: none
- o Author: Tirumaleswar Reddy, kondtir@gmail.com
- o Change Controller: IESG
- o Provisional registration? No

## [11.2.](#) JSON Web Token Claims Registration

### [11.2.1.](#) Registry Contents Additions Requested

IANA is requested to assign the following claims in the registry maintained in: <https://www.iana.org/assignments/jwt/jwt.xhtml>.

- o Claim Name: 'server'
- o Claim Description: DNS server identity
- o Change Controller: IESG

- o Specification Document(s): [Section 5.2.1](#) of [THIS\_DOCUMENT]
- o Claim Name: 'resinfo'
- o Claim Description: Resolver information of DNS server.
- o Change Controller: IESG
- o Specification Document(s): [Section 5.2.2](#) of [THIS\_DOCUMENT]

## [11.3.](#) DNS Resolver Information Registration

IANA will add the name "attested-resinfo" to the DNS Resolver Information registry defined in [Section 7.2](#) of [I-D.reddy-add-resolver-info].

## [12.](#) Acknowledgments

This specification leverages some of the work that has been done in [\[RFC8225\]](#). Thanks to Tommy Jensen, Ted Lemon, Paul Wouters, Neil Cook, Vittorio Bertola, Vinny Parla, Chris Box, Ben Schwartz and Shashank Jain for the discussion and comments.

## [13.](#) References

### [13.1.](#) Normative References

[I-D.ietf-add-ddr]

Pauly, T., Kinnear, E., Wood, C. A., McManus, P., and T. Jensen, "Discovery of Designated Resolvers", [draft-ietf-add-ddr-03](#) (work in progress), October 2021.

[I-D.ietf-add-dnr]

Boucadair, M., Reddy, T., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for the Discovery of Network-designated Resolvers (DNR)", [draft-ietf-add-dnr-02](#) (work in progress), May 2021.

[I-D.reddy-add-resolver-info]

Reddy, T. and M. Boucadair, "DNS Resolver Information", [draft-reddy-add-resolver-info-03](#) (work in progress), April 2021.

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

[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>>.

- [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, <<https://www.rfc-editor.org/info/rfc3986>>.
- [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, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC6570] Gregorio, J., Fielding, R., Hadley, M., Nottingham, M., and D. Orchard, "URI Template", [RFC 6570](#), DOI 10.17487/RFC6570, March 2012, <<https://www.rfc-editor.org/info/rfc6570>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", [BCP 13](#), [RFC 6838](#), DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.
- [RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", [RFC 6979](#), DOI 10.17487/RFC6979, August 2013, <<https://www.rfc-editor.org/info/rfc6979>>.
- [RFC7493] Bray, T., Ed., "The I-JSON Message Format", [RFC 7493](#), DOI 10.17487/RFC7493, March 2015, <<https://www.rfc-editor.org/info/rfc7493>>.
- [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>>.
- [RFC7518] Jones, M., "JSON Web Algorithms (JWA)", [RFC 7518](#), DOI 10.17487/RFC7518, May 2015, <<https://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, <<https://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, <<https://www.rfc-editor.org/info/rfc7638>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", [RFC 7858](#), DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", [RFC 8484](#), DOI 10.17487/RFC8484, October 2018, <<https://www.rfc-editor.org/info/rfc8484>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", [BCP 219](#), [RFC 8499](#), DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.

### [13.2](#). Informative References

- [I-D.btw-add-home]  
Boucadair, M., Reddy, T., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for Encrypted DNS Discovery", [draft-btw-add-home-12](#) (work in progress), January 2021.
- [I-D.btw-add-ipsecme-ike]  
Boucadair, M., Reddy, T., Wing, D., and V. Smyslov, "Internet Key Exchange Protocol Version 2 (IKEv2) Configuration for Encrypted DNS", [draft-btw-add-ipsecme-ike-03](#) (work in progress), May 2021.
- [I-D.ietf-dnsop-terminology-ter]  
Hoffman, P., "Terminology for DNS Transports and Location", [draft-ietf-dnsop-terminology-ter-02](#) (work in progress), August 2020.
- [I-D.ietf-dprive-dnsoquic]  
Huitema, C., Dickinson, S., and A. Mankin, "Specification of DNS over Dedicated QUIC Connections", [draft-ietf-dprive-dnsoquic-04](#) (work in progress), September 2021.

Internet-Draft

DNS Server Info with Assertion Token

October 2021

## [I-D.ietf-rats-architecture]

Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote Attestation Procedures Architecture", [draft-ietf-rats-architecture-12](#) (work in progress), April 2021.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", [RFC 7159](#), DOI 10.17487/RFC7159, March 2014, <<https://www.rfc-editor.org/info/rfc7159>>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", [RFC 7626](#), DOI 10.17487/RFC7626, August 2015, <<https://www.rfc-editor.org/info/rfc7626>>.

[RFC7816] Bortzmeyer, S., "DNS Query Name Minimisation to Improve Privacy", [RFC 7816](#), DOI 10.17487/RFC7816, March 2016, <<https://www.rfc-editor.org/info/rfc7816>>.

[RFC8225] Wendt, C. and J. Peterson, "PASSport: Personal Assertion Token", [RFC 8225](#), DOI 10.17487/RFC8225, February 2018, <<https://www.rfc-editor.org/info/rfc8225>>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, [RFC 8259](#), DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.

[RFC8310] Dickinson, S., Gillmor, D., and T. Reddy, "Usage Profiles for DNS over TLS and DNS over DTLS", [RFC 8310](#), DOI 10.17487/RFC8310, March 2018, <<https://www.rfc-editor.org/info/rfc8310>>.

[RFC8572] Watsen, K., Farrer, I., and M. Abrahamsson, "Secure Zero Touch Provisioning (SZTP)", [RFC 8572](#), DOI 10.17487/RFC8572, April 2019, <<https://www.rfc-editor.org/info/rfc8572>>.

[RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", [RFC 8914](#), DOI 10.17487/RFC8914, October 2020,

<<https://www.rfc-editor.org/info/rfc8914>>.

[UNICODE] The Unicode Consortium, "The Unicode Standard", June 2016,  
<<http://www.unicode.org/versions/latest/>>.

Reddy, et al.

Expires April 10, 2022

[Page 17]

---

Internet-Draft

DNS Server Info with Assertion Token

October 2021

## Appendix A. Example of ES256-based REAT JWS Serialization and Signature

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

- o 'protected', with the value BASE64URL(UTF8(JWS Protected Header))
- o 'payload', with the value BASE64URL (JWS Payload)
- o '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 REAT.

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

```
{
  "server":{
    "adn":"example.com"
  },
  "iat":1443208345,
  "exp":1443640345,
  "resinfo": {
    "qnameminimization":false
  }
}
```

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

```
{"exp":1443640345,"iat":1443208345,"resinfo":{"qnameminimization":false},"server":{"adn":"example.com"}}
```

eyJleHAiOjE0NDM2NDZlNDU5ImVhdCI6MTQ0MzIwODM0NSwicmVzaW5mbyI6eyJxYmFtZWl1bmVtaXphdGlvbiI6ZmFsc2V9LCJzZXJ2ZXIiOi0nsiYWRuIjoieXhhbXBsZS5jb20ifX0

Reddy, et al. Expires April 10, 2022 [Page 18]

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

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

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpzZW50b3VhdC5lbnQ9YXQuY2VyIn0=

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

d1g7szj0roHsWe8psCzYVl4QdN2b7pQnq8EJhc4j3G0Jj2NE6M9Em6aidtycnFJ5  
mRj3ojiUfVF6rK5RksD0rg

Step 8 describes how to create the final REAT 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):

eyJhbGciOiJFUzI1NiIsInR5cCI6InJhdCI6Ing1dSI6Imh0dHBzOi8vY2VydC5l  
eGFtcGxllmNvbS9yYXQuY2VyIn0  
.

eyJhbGciOiJFUzI1NiIsInR5cCI6InJhdCI6Ing1dSI6Imh0dHBzOi8vY2VydC5l  
eGFtcGxllmNvbS9yYXQuY2VyIn0  
.

d1g7szj0roHsWe8psCzYVl4QdN2b7pQnq8EJhc4j3G0Jj2NE6M9Em6aidtycnFJ5  
mRj3ojiUfVF6rK5RksD0rg

#### [A.1.](#) X.509 Private Key in PKCS#8 Format for ES256 Example\*\*

```
-----BEGIN PRIVATE KEY-----  
MIGHAgEAMBMGBByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQgeVZzL1gdAFr88hb2  
OF/2NxApJCzGCEDdfSp6VQ030hyhRANCAAQRWz+jn65BtOMvdyHKcvjBeBSDZH2r  
1RTwjmYSi9R/zpBnuQ4EiMnCqfMPWiZqB4QdbAd0E7oH50VpuZ1P087G  
-----END PRIVATE KEY-----
```

#### [A.2.](#) X.509 Public Key for ES256 Example\*\*

```
-----BEGIN PUBLIC KEY-----  
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEVVs/o5+uQbTjL3chynL4wXgUg2R9  
q9UU8I5mEovUf86QZ7k0BIjJwqnzDlomagEHwwHdB06B+dFabmdT9P0xg==  
-----END PUBLIC KEY-----
```

### [Appendix B.](#) Complete JWS JSON Serialization Representation with multiple Signatures

The JWS payload used in this example as follows.

```
{
  "server":{
    "adn":"example.com"
  },
  "iat":1443208345,
  "exp":1443640345,
  "resinfo": {
    "qnameminimization":false
  }
}
```

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

```
{"exp":1443640345,"iat":1443208345,"resinfo":{"qnameminimization":false},"server":{"adn":"example.com"}}
```

The JWS protected Header value used for the first signature is same as that used in the example in [Appendix A](#). The X.509 private key used for generating the first signature is same as that used in the example in [Appendix A.1](#).

The JWS Protected Header value used for the second signature is:

```
{
  "alg":"ES384",
  "typ":"rat",
  "x5u":"https://cert.audit-example.com/rat.cer"
}
```

The complete JWS JSON Serialization for these values is as follows (with line breaks within values for display purposes only):

```
{
  "payload":
    "eyJhbGciOiJIJFUiI1NiIsInR5cCI6IkpzZW50dHBzOi8vY2VydC5l"
```

```

    eGFtcGxllMnVbS9yYXQuY2VyIn0",
    "signatures":[
      {"protected":"eyJhbGciOiJFUzI1NiIsInR5cCI6InJhdCI6Ing1dSI6Imh0dHBz
        Oi8vY2VydC5leGFtcGxllMnVbS9yYXQuY2VyIn0",
        "signature":"d1g7szj0roHsWe8psCzYVl4QdN2b7pQnq8EJhc4j3G0Jj2NE6M9E
        m6aidtycnFJ5mRj3ojiUfVF6rK5RksD0rg"}},
      {"protected":"eyJhbGciOiJFUzM4NCIsInR5cCI6InJhdCI6Ing1dSI6Imh0dHBz
        zOi8vY2VydC5hdWRpdC1leGFtcGxllMnVbS9yYXQuY2VyIn0",
        "signature":"GnKuEEFql_Y8HdZl_mqd027DlziGRXFHvjMoY_ukX-M0k5v2jSL
        vsQAY0GdKFnt3JY6t938HfBV1onsWerNhgceMJpx5hAsl-xus3fmNY8K1g6QK39
        hn2Dhbleeeyp0f"}]}
  ]
}

```

#### [B.1.](#) X.509 Private Key in PKCS#8 format for E384 Example\*\*

```

-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGBByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQgeVZzL1gdAFr88hb2
OF/2NxApJCzGCEDdfSp6VQ030hyhRANCAAQRWz+jn65BtOMvdyHKcvjBeBSDZH2r
1RTwjmYSi9R/zpBnuQ4EiMnCqfMPWiZqB4QdbAd0E7oH50VpuZ1P087G
-----END PRIVATE KEY-----

```

#### [B.2.](#) X.509 Public Key for ES384 Example\*\*

```

-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEEVs/o5+uQbTjL3chynL4wXgUg2R9
q9UU8I5mEovUf86QZ7k0BIjJwqnzD1omageEHWwHdB06B+dFabmdT9P0xg==
-----END PUBLIC KEY-----

```

Authors' Addresses

Reddy, et al.

Expires April 10, 2022

[Page 21]

Internet-Draft

DNS Server Info with Assertion Token

October 2021

Tirumaleswar Reddy  
 Akamai  
 Embassy Golf Link Business Park  
 Bangalore, Karnataka 560071  
 India

Email: kondtir@gmail.com

Dan Wing  
Citrix Systems, Inc.  
USA

Email: dwing-ietf@fuggles.com

Michael C. Richardson  
Sandelman Software Works  
USA

Email: mcr+ietf@sandelman.ca

Mohamed Boucadair  
Orange  
Rennes 35000  
France

Email: mohamed.boucadair@orange.com