JOSE
Internet-Draft
Intended status: Standards Track
Expires: 17 September 2025

T. Reddy
Nokia
H. Tschofenig
H-BRS

16 March 2025

Enhanced JWE Security with Detached Additional Authenticated Data (AAD)
draft-reddy-jose-detached-aad-01

Abstract

This draft introduces a mechanism to support detached Additional Authenticated Data (AAD) in JWE (JSON Web Encryption), allowing the AAD to be derived from context-specific information, such as session identifiers, algorithm identifiers, and identifiers of communication endpoints, rather than being transmitted in-band. This mechanism strengthens security by mitigating risk against unknown-key-share attacks and/or other exploitation techniques that depend on some type of confusion over the role played by each party.

The document explains how to integrate this functionality into JWE, covering both JWE JSON Serialization and JWE Compact Serialization.

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 17 September 2025.

Copyright Notice

Table of Contents

1. Introduction

Encryption of payloads is performed using a symmetric content encryption key, which can be established through various key distribution mechanisms. Examples of such content key distribution mechanisms include public key encryption, such as HPKE, and ephemeral-static Diffie-Hellman. As part of this content key distribution mechanism an important design decision is what information is included about the context in which the encryption is use. This ensures that the content key material is "bound" to the context of the transaction.

There are various techniques to ensure that the established content encryption key is not used across different contexts, such as incorporating the context into the key derivation function, adding extra context information in protected JOSE headers, or including out-of-band information in the Additional Authenticated Data (AAD).

This specification uses the AAD for including context information shared out of band. This approach reduces the amount of data transmitting directly within the message. It enhances security by ensuring the cryptographic binding of context information, by minimizing the attack surface. As a consequence, the AAD must be derived by both the sender and receiver independently. A JSON-based data structure is proposed to standardize the AAD information. The described approach is applicable to both JWE JSON Serialization and JWE Compact Serialization. This mechanism is particularly useful in scenarios like OpenID for Verifiable Credentials (OID4VC), where a verifier must validate context information without depending on in-band AAD.

2. Conventions and Definitions

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.

3. Motivation and Use Cases

A common use case for detached AAD arises in protocols like OpenID for Verifiable Credentials (OID4VC), where a verifier receives a JWE response from a wallet application.

In such scenarios:

* The verifier sends a request to the wallet (typically a mobile app).

* The wallet responds with an encrypted message that includes context information (e.g., the origin and the recipient of the request) as part of the AAD.

* Without proper validation of AAD, attackers could exploit vulnerabilities by manipulating context data, potentially leading to malicious actions.

By requiring the verifier and wallet to securely agree on a method for deriving the AAD, the encrypted response can be cryptographically bound to the request’s context. This ensures the response is valid only within its intended context, reducing risks such as phishing, replay, and tampering attacks.

4. Benefits of Detached AAD

The introduction of detached AAD offers several key benefits:

* Improved Security: By deriving AAD directly from the context, the need to transmit and validate in-band AAD is eliminated. This minimizes the attack surface and reduces the risks of tampering and replay.

* Enhanced Flexibility: Applications gain greater control over AAD usage, ensuring it is transmitted only when necessary and otherwise derived out-of-band.

* Consistency Across Serializations: Supporting detached AAD in both JWE JSON Serialization and JWE Compact Serialization enables a consistent approach to context-specific AAD across various serialization formats.

5. JWE JSON Serialization

When the AAD is detached, the "detached_aad" parameter is set to true, indicating that the AAD MUST be computed by both the sender and receiver from context information.

* If the "detached_aad" parameter is true, the detached AAD MUST be derived out-of-band, following the process specified in Step 5 of Section 7.

* If the "aad" parameter contains data and the "detached_aad" parameter is not present or set to false, it is used directly as the AAD for encryption.

* If both the "detached_aad" and "aad" parameters are present, the AADs are combined and used according to the process outlined in Step 5 of Section 7.

* The derived AAD is treated as part of the encryption context but is never transmitted within the JWE structure.

Example:

A JSON Encoded JWE is shown below (with line-breaks added for readability).

```
{
  "protected": "eyJhbGciOiJIUEtFLVAyNTYtU0hBMjU2LUExMjhHQ00iLCJlbmMiOiJkaXIiLCJraWQiOiJ1cm46aWV0ZjpwYXJhbXM6b
   2F1dGg6andrLXRodW1icHJpbnQ6c2hhLTI1NjpTNkFYZmRVXzZZZnp2dTBLRERKYjBzRnV3bklXUGs2TE1URXJZaFBiMzJzIiwicHNrX2l
   kIjoib3VyLXByZS1zaGFyZWQta2V5LWlkIiwiYXV0aF9raWQiOiJ1cm46aWV0ZjpwYXJhbXM6b2F1dGg6andrLXRodW1icHJpbnQ6c2hhL
   TI1NjpTNkFYZmRVXzZZZnp2dTBLRERKYjBzRnV3bklXUGs2TE1URXJZaFBiMzJzIn0",
  "encrypted_key": "BD7QVodtG-FwYASgb36zuTzUCc80aiYwS6JOOE-6_heUGyAZt-cU0818e4oYqP7ebBuW3KTM9EQA0vM5fWp6hj0",
  "ciphertext": "ZxqtYoomgVQGctnv1I_EBVI1NIeJ7qJw2iVtqwUw3fXa8FK-",
  "aad": "8J-PtOKAjeKYoO-4jyBiZXdhcmUgdGhlIGFhZCE"
}
```


After verification:


```
{
  "protectedHeader": {
    "alg": "HPKE-0",
    "enc": "dir",
    "kid": "urn:ietf:params:oauth:jwk-thumbprint:sha-256:S6AXfdU_6Yfzvu0KDDJb0sFuwnIWPk6LMTErYhPb32s",
    "psk_id": "our-pre-shared-key-id",
    "auth_kid": "urn:ietf:params:oauth:jwk-thumbprint:sha-256:S6AXfdU_6Yfzvu0KDDJb0sFuwnIWPk6LMTErYhPb32s",
    "aad_detached": true
  },
  "plaintext": "This is plaintext!",
  "aad": "This is aad!"
}
```


In this case, the "aad_detached" parameter set to true indicates that the AAD must be derived from context information shared between the sender and receiver.

6. JWE Compact Serialization

6.1. Supporting Detached AAD in JWE Compact Serialization

To enable the use of detached Additional Authenticated Data (AAD) without introducing backward compatibility issues, this specification defines a mechanism that leverages external context-specific information and an optional protected header parameter to signal the presence of detached AAD. The existing JWE Compact Serialization format, as defined in Section 7.1 of [RFC7516], remains unchanged.

6.2. Mechanism Overview

1. External Metadata for Detached AAD: The detached AAD is not included in the JWE Compact Serialization format.

2. Optional Protected Header Parameter:

* A new optional parameter, "detached_aad", is introduced in the JWE Protected Header.

* When set to true, this parameter indicates that the AAD is detached and must be obtained through an external context.

3. Backward Compatibility: This mechanism does not alter the structure of the JWE Compact Serialization format, ensuring full compatibility with existing implementations.

The "aad_detached": true parameter is included in the JWE Protected Header. For example:

```
{
  "alg": "HPKE-0",
  "enc": "A256GCM",
  "detached_aad": true
}
```

7. Deriving detached AAD

When using detached AAD, both the sender and receiver MUST follow the same derivation process to ensure consistency. The derived AAD is never transmitted; instead, it is independently computed by both parties. The process for deriving the AAD involves the following steps:

1. Identify Context Information Elements: Applications define the context information required for AAD derivation and they are shared by both sender and receiver.

Example:

```
{
   "session_id": "sess-1234",
   "sender": "alice@example.com",
   "receiver": "bob@example.com"
}
```

1. Normalize Context Information Elements

* Each context element MUST be encoded as a UTF-8 string.

* The elements MUST be sorted in lexicographical order of their keys (as per [RFC8785]), ensuring consistent ordering during derivation.

Example:

```
Normalize Elements (Sorted):
{
   "sender": "alice@example.com",
   "session_id": "sess-1234",
   "receiver": "bob@example.com"
}
```

1. Serialize the Canonicalized JSON: Serialize the canonicalized JSON object, ensuring it has no extra spaces or newlines.

Example:

```
Concatenated String: {"sender": "alice@example.com","session_id":"sess-1234","receiver":"
bob@example.com"}
```

1. Apply Cryptographic Hashing: Apply a cryptographic hash function (e.g., SHA-256) to the serialized canonicalized JSON string to derive the AAD. The resulting hash will serve as the AAD used in the encryption operation. The use of SHA-256 is RECOMMENDED; however, other cryptographic hash functions MAY be used if they provide equivalent or stronger security properties.

Example:

```
Derived AAD: "SHA256({"session_id":"sess-1234","timestamp":"2025-01-10T12:00Z"})"
```

1. Use the Derived AAD in Encryption

* Step 15 in Section 5.1 of [RFC7516] outlines how the AAD is used in the encryption operation. In the case of detached AAD, the derived AAD is treated as part of the encryption context, even though it is never transmitted within the JWE structure.

* Compact JWE and JWE JSON Serialization without an explicit JWE AAD: For a message that lacks the JWE AAD but includes the derived AAD, the derived AAD is used in conjunction with the Content Encryption Key (CEK), JWE IV, and the message M to generate the JWE Ciphertext and JWE Authentication Tag. The Additional Authenticated Data (AAD) encryption parameter is:

ASCII(Encoded Protected Header || ’.’ ||BASE64URL(Detached AAD)).

* JWE JSON Serialization with an explicit JWE AAD: In the case of a message containing both the JWE AAD and the derived AAD, the derived AAD is used in conjunction with the Content Encryption Key (CEK), JWE IV, JWE AAD, and the message M to generate the JWE Ciphertext and JWE Authentication Tag. The Additional Authenticated Data (AAD) encryption parameter is:

ASCII(Encoded Protected Header || '.' ||BASE64URL(JWE AAD)|| '.' ||BASE64URL(Detached AAD)).

* Both the sender and receiver MUST independently compute the detached AAD to ensure consistency.

* The detached AAD is not transmitted within the JWE structure; instead, it is derived from context information elements out-of-band.

1. Error Handling: If the derived AAD does not match the expected value during decryption, the JWE MUST be treated as invalid, and the decryption process MUST fail.

8. Security Considerations

Detached AAD enhances security by ensuring that the AAD is tightly bound to the specific context shared by both the sender and recipient. However, the following considerations must be observed:

* Both the sender and recipient MUST use the same method to derive the AAD from the context to avoid mismatches, as discrepancies will lead to decryption failures.

* Applications using detached AAD must ensure that context information is validated and securely exchanged to prevent manipulation by attackers.

* Implementations should account for potential synchronization issues, such as clock drift, when deriving the AAD from time-sensitive context.

9. IANA Considerations

9.1. JSON Web Signature and Encryption Header Parameters

The following entry is requested to be added by IANA to the "JSON Web Signature and Encryption Header Parameters" registry:

9.1.1. aad_detached

* Header Parameter Name: "detached_aad"

* Header Parameter Description: Indicates that the AAD is detached and must be conveyed through an external mechanism.

* Header Parameter Usage Location(s): JWE

* Change Controller: IETF

* Specification Document(s): RFCXXXX

Acknowledgments

TODO

Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

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

[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/rfc/rfc8174>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <https://www.rfc-editor.org/rfc/rfc8785>.

Authors' Addresses

Tirumaleswar Reddy
Nokia
Bangalore
Karnataka
India
Email: kondtir@gmail.com

Hannes Tschofenig
University of Applied Sciences Bonn-Rhein-Sieg
Germany
Email: Hannes.Tschofenig@gmx.net