Network Working Group
Internet-Draft
Intended status: Informational
Expires: 11 June 2026

L. Hopkins

E. Turner

8 December 2025

Evidence Package Format Specification: Storing Evidence from Software Testing
draft-hopkins-evp-spec-09

Abstract

Taking evidence is a key part of any robust software testing process. This specification defines a format which collects evidence together and stores metadata and annotations in an organised fashion from both manual and automated testing sources.

This work is not a standard and does not enjoy community consensus.

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 11 June 2026.

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this specification is to define a format for storage of evidence produced as the result of software testing that:

* allows for basic collation of evidence;
* can store any kind of file type that might be produced;
* stores data compressed;
* stores related evidence together, but allows for dividing up by test case;
* allows test evidence to be attested with a traceable list of attestors, and;
* is built upon widely available standards.

The format does not attempt to:

* act as an captioned archiving solution for other purposes outside of software testing, even if it may be suitable for them.

1.2. Intended Audience

This specification is intended for those who might wish to write their own implementation of the evidence package format. There are a number of situations where writing an implementation may be desirable:

* in an automation tool that runs a number of operations to automatically test something, to produce an evidence package containing the results of the automated testing;
* in a manual evidence collection tool, where a user might want to collect evidence in a single, easy to manage place for later processing or sharing;
* in an analysis tool, to view, annotate, share and understand the evidence from previous testing;
* in a viewer, to view evidence that has been shared, for example from a testing team to a customer, or;
* any other situation where it may be desirable to collect test evidence and bundle it together for later.

1.3. Changes from Previous Versions

This document forms the original accepted specification.

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.

3. Specification

An evidence package is a structured ZIP archive [zip] using deflate compression. It MUST contain the file "manifest.json", and the directories "media" and "test_cases" internally within the ZIP archive. This structure does not need to be represented outside of the ZIP archive and as such the internal structure does not need to be understood by an end-user of any tool that works with evidence packages.

See Appendix A for an example of the file’s internal structure.

3.1. "manifest.json" File

The manifest.json file defines metadata relating to the entire package of evidence. It MUST be a UTF-8 encoded, LF line ended, JSON [RFC8259] file with the following elements:

Element	Condition	Type	Section	Description
$schema	Optional	String	Section 3.1.1	The $schema element MAY point to a copy of the schema for the manifest.
metadata	Mandatory	Object	Section 3.1.2	The metadata element stores package metadata.
custom_metadata	Mandatory	Object	Section 3.1.3	Custom metadata fields for test cases in this package.
media	Mandatory	Array	Section 3.1.4	The media element stores a list of media files that are stored in this evidence package.
test_cases	Mandatory	Array	Section 3.1.5	The test_cases element stores a list of test cases.

Table 1

See an example manifest.json file in Appendix B.

3.1.1. "$schema" Element

This element MAY optionally be provided to point to a JSON schema describing the structure of the file. This is typically most useful for validation, however it MUST be acceptable for it to be missing, and this specification should be seen as the primary definition of structure over anything defined in the linked schema.

The JSON schema provided at by this element may give details about any additional fields used that are not defined in this specification.

3.1.2. "metadata" Element

Element	Condition	Type	Section	Description
title	Mandatory	String		The name of the evidence package.
authors	Mandatory	Array	Section 3.1.2.1	The authors attributed to this evidence package.

Table 2

3.1.2.1. "authors" Array Element

Element	Condition	Type	Description
name	Mandatory	String	The author’s name.
email	Optional	String/ Null	The author’s email address, although format is not verified.

Table 3

3.1.3. "custom_metadata" Element

Elements within this object will become custom metadata properties for test cases in this package. Each object MUST have the following fields:

Element	Condition	Type	Section	Description
name	Mandatory	String		The name of this custom metadata field.
description	Mandatory	String	Section 3.1.2.1	The description of this custom metadata field.
primary	Mandatory	Boolean	Section 3.1.3.1	Is this custom field primary?

Table 4

3.1.3.1. "primary" Boolean

The "primary" value of custom metadata fields MAY be false for all fields, or MAY be true for exactly one field. It MUST NOT be true for more than one field.

The purpose of primary is not enforced as part of this specification, however it should be seen as suggesting that one custom metadata field is more useful than others, and as such may be used to influence the information displayed to users, for example an implementer might choose to show the primary custom metadata value for each test case alongside it.

3.1.4. "media" Array Element

Element	Condition	Type	Description
sha256_checksum	Mandatory	String	The SHA256 checksum of the associated media file.
mime_type	Mandatory	String	The Internet Media Type [RFC2046] of the associated media file.

Table 5

3.1.5. "test_cases" Array Element

Element	Condition	Type	Section	Description
id	Mandatory	String		The UUID of the test case. If present here, there MUST be an associated test case file in the "test_cases" directory of the package with the name "<UUID>.json".
attestations	Mandatory	Array of Strings	Section 3.1.5.1	An array of attestations over this test case.

Table 6

3.1.5.1. "attestations" Array

The elements within the "attestations" array MUST be JWS [RFC7515] signatures. The signature payload must be a SHA256 checksum of a copy of the test case manifest (i.e. the file "uuid.json"), having been processed into JSON canonical format as defined in [RFC8785].

In environments where it is desirable to frequently validate attestations, it is recommended to include the "x5c" X.509 certificate, and a "jku" (JWT Key URL). An implementing client is RECOMMENDED to display the URL in some fashion if the key passes validation as a way to prove the signing party, however another approach may also be desirable depending on environment.

Implementing clients SHOULD NOT use symmetric key types (although it may be acceptable for tools that are only used within a limited scope), and APIs implementing this specification MAY choose to be incompatible with symmetric types.

As a worked example, a test case may start off like this:

```
{
  "$schema":
    "https://evidenceangel-schemas.hpkns.uk/testcase.2.schema.json",
  "metadata": {
    "title": "Test Case",
    "execution_datetime": "2025-12-05T20:40:22.743821295Z",
    "passed": null
  },
  "evidence": [
    {
      "kind": "text/plain",
      "value": "plain:Hello, world!"
    }
  ]
}
```

This should then be canonicalised. This is shown below, but newlines have been added for presentation, indicated with a "\" character at the line end.

```
{"$schema":"https://evidenceangel-schemas.hpkns.uk/testcase.2.schema\
.json","evidence":[{"kind":"text/plain","value":"plain:Hello, world!\
"}],"metadata":{"execution_datetime":"2025-12-05T20:40:22.743821295Z\
","passed":null,"title":"Test Case"}}
```

A SHA256 checksum can be generated:

```
6cd9684d866d5dacd85064f20d0b3fd423e30946c6b99d1f2defae529360becc
```

This can now be signed and the original manifest can be modified:

```
{
  "id": "7928de11-8de8-4bfe-b5b7-cbf07c7066d9",
  "attestations": [
    "eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.YmI5YzdkMjczYWY2NzE5NWM1MW\
    M1N2YyNzRjMDc5NTViODZiMDA3MWE0MDU3MWFjOTIwYzE2M2UzNDQxYzUwZQ.Ktb\
    RLfAh8UmSxSWYnObpydXyjGO_IPF2acU_x-eFY6dLDBD809zJm6HaTE9jjsQlnX8\
    eGWRIOzKXTWMdgp-fXg"
  ],
  "some_other_value": "Added from somewhere other than this spec!"
}
```

3.2. "test_cases" Directory

The test cases directory stores the manifests for each test case within this evidence package.

Each test case is stored as a JSON file, with a UUIDv4 name [RFC9562]. A test case present here MUST have a valid entry in the manifest "test_cases" array defined in Section 3.1.5.

3.2.1. "<uuid>.json" File

Element	Condition	Type	Section	Description
$schema	Optional	String	Section 3.2.1.1	The $schema element MAY point to a copy of the schema for the manifest.
metadata	Mandatory	Object	Section 3.2.1.2	The metadata relating to this test case.
evidence	Mandatory	Array	Section 3.2.1.3	The evidence within this test case.

Table 7

See an example <uuid>.json file in Appendix C.

3.2.1.1. "$schema" Element

This element MAY optionally be provided to point to a JSON schema describing the structure of the file. This is typically most useful for validation, however it MUST be acceptable for it to be missing, and this specification should be seen as the primary definition of structure over anything defined in the linked schema.

The JSON schema provided at by this element may give details about any additional fields used that are not defined in this specification.

3.2.1.2. "metadata" Element

Element	Condition	Type	Description
title	Mandatory	String	The title of the test case.
execution_datetime	Mandatory	String	The ISO8601 date and time of the

			execution of this test case starting.
passed	Optional	Enumerated	The state of the test case, if present MUST be either the string "pass" or "fail", or null. If absent, it MUST be interpreted as null.
custom	Mandatory	Object	Custom metadata values.

Table 8

The "custom" field is used to add custom metadata that has been specified in the package manifest's "custom_metadata" field. If a value is specified in "custom", it MUST be present in the package manifest, but all values in the package manifest do not need to be present here. All values MUST be strings and are stored as a simple key-value map, with the custom field ID defined in the manifest as the key.

3.2.1.3. "evidence" Array Element

Element	Condition	Type	Section	Description
kind	Mandatory	String	Section 3.2.1.3.1	The Internet Media Type [RFC2046] of data stored.
value	Mandatory	String	Section 3.2.1.3.2	The data stored within this piece of evidence.
caption	Optional	String/Null		An optional caption for this piece of evidence.

original_filename	Optional	String/Null		The original filename.

Table 9

3.2.1.3.1. "kind"

The "kind" of evidence MUST be an Internet Media Type [RFC2046].

For more information about each type, see Section 5.

3.2.1.3.2. "value"

The "value" MUST be one of the following acceptable patterns:

* "plain:" followed by plain text;
* "media:" followed by a media file SHA256 hash, or;
* "base64:" followed by a base64 string of data without padding.

3.3. "media" Directory

The "media" directory stores data in files within the ZIP archive that would be otherwise impractical to store directly in the test cases.

Files stored in this directory are of abitrary type. They MUST be named by their SHA256 checksum [RFC6234] with no extension. Their SHA256 checksum and media type MUST be stored in the package manifest "media" element.

In the unlikely event that there is a checksum clash, there is currently no preferred method for resolving this. The probability of such a situation is decided to be acceptably low given the expected size and number of files stored in an evidence package, however implementers MAY choose to store the clashing file as base64 data instead of as an additional media file.

4. Handling an Evidence Package

4.1. Locking

When loading an evidence package, implemetors MUST use a lock file with the file name ".~lock." followed by the full name of the package it protects, followed by "#", for example for a package called "example.evp", the lock file MUST be called ".~lock.example.evp#". It MUST be located adjacent (in the same directory as) the evidence package. The file MUST contain the process ID of the process holding the lock.

The lock file should be considered as locking the package if it is present, regardless of contents.

If either of these is not the case, it should be assumed that the there is no current lock over the package.

4.2. Media Loading

Software implementing the evidence package format MUST NOT load files from the "media" directory into memory until it is needed for display or for extraction. implementers MUST use streams to load media files to avoid trying to load the entire file into memory as it may not fit.

5. Kinds of Evidence

Evidence packages can support any valid Internet Media Type [RFC2046] as evidence. implementers of this specification MUST be able to display the following types:

Media Type	Description
text/plain	Plain text with no formatting.
text/markdown	Text with markdown support.
text/vnd.angel.http-data	An HTTP request/response pair.
image/*	An image that should be rendered where possible.

Table 10

Common image formats SHOULD be rendered where possible, but it is not required to support every possible type of image.

Markdown SHOULD be rendered where possible, but it may be adapted for security reasons. If it is changed before display, a notice MUST be displayed to the user disclosing that it has been adjusted for security. For example, it is acceptable to strip raw HTML tags before rendering.

Other media types MUST be supported insofar as being able to extract the data from the evidence package so that they can be opened in other software.

5.1. HTTP Requests

Where text/vnd.angel.http-data is used, an HTTP request and response MUST be present in plain text, and a Record Separator character (0x1e) MUST be used to split the request and response portion. In other words, the format MUST comply with the following regular expression:

```
^(?<request>[.\r\n]*)\x1e(?<response>[.\r\n]*)$
```

For example the separator is present at _1_:

```
GET / HTTP/1.1
Host: example.com
User-Agent: HTTPie

\x1eHTTP/1.1 200 OK <1>
Cache-Control: max-age=1366
Connection: close
...
```

6. Extending Behaviours of an Evidence Package

Every JSON file within an evidence package MAY have new fields added, and as such extended behaviours MAY be implemented, however implementers MUST be able to load an evidence package without these additional fields.

When an implementer loads a file with fields it cannot understand, it MUST retain the fields on saving the file.

7. IANA Considerations

This document acts as the specification for the media type application/vnd.angel.evidence-package. Additionally, the media type text/vnd.angel.http-data is defined in Section 5.1.

8. Security Considerations

The evidence package format can store arbitrary files that may or may not be executable. implementers MUST NOT execute any file contained within and SHALL only extract the contained files if needed.

Otherwise, there are no concerns for security from the file type itself.

9. Normative References

[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>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <https://www.rfc-editor.org/info/rfc6234>.

[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>.

[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>.

[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>.

[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/info/rfc8785>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/info/rfc9562>.

[zip] PKWARE, Inc., ".ZIP File Format Specification", 1 November 2022, <https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT>.

Appendix A. Example Archive Layout

```
example.evp
 |- manifest.json
 |- media
 |   \- 203073da0b36a5921f2914e2093abcae7eb987846f405b438c25792bab1617fa
 \- test_cases
     \- eabb5d31-a958-4609-ac98-83365e14d18b.json
```

Appendix B. Example Package Manifest JSON

```
{
  "metadata": {
    "title": "Example Evidence Package",
    "authors": [
      {
        "name": "Anonymous Author"
      },
      {
        "name": "Lily Hopkins",
        "email": "lily@hpkns.uk"
      }
    ]
  },
  "custom_metadata": {
    "example": {
      "name": "Example Metadata Field",
      "description":
        "A field showing that custom fields can be added",
      "primary": true
    }
  },
  "media": [
    {
      "sha256_checksum":
  "203073da0b36a5921f2914e2093abcae7eb987846f405b438c25792bab1617fa",
      "mime_type": "text/plain"
    }
  ],
  "test_cases": [
    {
      "id": "eabb5d31-a958-4609-ac98-83365e14d18b",
      "attestations": [
        "eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.YmI5YzdkMjczYWY2NzE5NW\
        M1MWM1N2YyNzRjMDc5NTViODZiMDA3MWE0MDU3MWFjOTIwYzE2M2UzNDQxYz\
        UwZQ.KtbRLfAh8UmSxSWYnObpydXyjGO_IPF2acU_x-eFY6dLDBD809zJm6H\
        aTE9jjsQlnX8eGWRIOzKXTWMdgp-fXg"
      ]
    }
  ]
}
```

Appendix C. Example Test Case Manifest JSON

```
{
  "metadata": {
    "title": "Example Test Case",
    "execution_datetime": "2025-05-01T11:13:29+01:00",
    "passed": null,
    "custom": {
      "example": "Example custom metadata field value"
    }
  },
  "evidence": [
    {
      "kind": "text/plain",
      "value": "plain:This is some text based evidence"
    },
    {
      "kind": "text/plain",
      "value": "base64:VGhpcyBpcyBzb21lIHRleHQgYmFzZWQgYmFzZTY0IGVuY\
        29kZWQgZXZpZGVuY2U"
    },
    {
      "kind": "text/plain",
      "value": "media:203073da0b36a5921f2914e2093abcae7eb987846f405b\
        438c25792bab1617fa",
      "caption": "An example file",
      "original_filename": "example.txt"
    },
    {
      "kind":"image/png",
      "value": "media:c561967275f002e65b222b4577378f5a20a5881edd00fb\
        e648beef6b4f4971a9",
      "caption": "An example image",
      "original_filename": "image.png"
    }
  ]
}
```

Appendix D. JSON Schema for Package Manifest

In this snippet, some newlines are added for presentation. These are denoted with a "\" at the end of the preceding line and should be considered excluded from the JSON.

```
{
  "$id":
    "https://evidenceangel-schemas.hpkns.uk/manifest.2.schema.json",
  "$schema": "http://json-schema.org/draft-07/schema",
  "type": "object",
  "description":
```

```
   "The metadata file ‘metadata.json‘ as part of an evidence package.",
     "properties": {
       "metadata": {
         "type": "object",
         "properties": {
           "title": {
             "type": "string",
             "description": "The name of the evidence package.",
             "minLength": 1,
             "maxLength": 30
           },
           "authors": {
             "type": "array",
             "description":
               "The authors attributed to this evidence package.",
             "items": {
               "type": "object",
               "properties": {
                 "name": {
                   "type": "string",
                   "description": "The author’s name."
                 },
                 "email": {
                   "type": "string",
                   "description":
         "The author’s email address, although format is not verified."
                 }
               },
               "required": ["name"]
             }
           },
           "description": {
             "type": "string",
             "description": "An optional description of the package."
           }
         },
         "required": ["title", "authors"]
       },
       "custom_metadata": {
         "type": "object",
         "description": "Custom metadata fields for test cases",
         "patternProperties": {
           ".+": {
             "type": "object",
             "description": "A custom metadata field",
             "properties": {
               "name": {
                 "type": "string",
```

```
                "description":
                  "A user-friendly name for this custom property."
              },
              "description": {
                "type": "string",
                "description":
                  "A description for this custom property."
              },
              "primary": {
                "type": "boolean",
                "description": "Is this custom property the main one \
                  in this package? This may influence how it is \
                  displayed in editors."
              }
            },
            "required": ["name", "description", "primary"]
          }
        }
      },
      "media": {
        "type": "array",
        "items": {
          "type": "object",
          "description": "A media entry. When an entry is present in \
            this manifest, it MUST also be present in the ‘media‘\
            directory of the package.",
          "properties": {
            "sha256_checksum": {
              "type": "string",
              "description": "The SHA256 checksum of the media file. \
                This MUST also match identically the name of the file \
                with no extension in the ‘media‘ directory.",
              "pattern": "^[0-9a-f]{64}$"
            },
            "mime_type": {
              "type": "string",
              "description": "The MIME type of the media file."
            }
          },
          "required": ["sha256_checksum", "mime_type"]
        }
      },
      "test_cases": {
        "type": "array",
        "items": {
          "type": "object",
          "properties": {
            "id": {
```

```
              "type": "string",
              "format": "uuid",
              "description": "The UUID of the test case. If present \
                here, there MUST be an associated test case file in \
                the ‘testcases‘ directory of the package with the name\
                ‘<UUID>.json‘."
            },
            "attestations": {
              "type": "array",
              "description":
                "An array of attestations over this test case.",
              "items": {
                "type": "string",
                "description": "The elements within the \
                  \"attestations\" array **MUST** be JWS [RFC7515] \
                  signatures. The signature payload must be a SHA256 \
                  checksum of a copy of the test case manifest (i.e. \
                  the file \"uuid.json\"), having been processed into \
                  JSON canonical format as defined in [RFC8785].",
                "pattern": "^[A-z0-9_-]+\\.[A-z0-9_-]+\\.[A-z0-9_-]+$"
              }
            }
          },
          "required": ["id", "attestations"]
        }
      }
    },
    "required": ["metadata", "media", "test_cases"]
  }
```

Appendix E. JSON Schema for Test Case Manifest

In this snippet, some newlines are added for presentation. These are denoted with a "\" at the end of the preceding line and should be considered excluded from the JSON.

```
  {
    "$id":
      "https://evidenceangel-schemas.hpkns.uk/testcase.2.schema.json",
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "description": "A test case file ‘testcases/<UUID>.json‘ as part \
      of an evidence package.",
    "properties": {
      "metadata": {
        "type": "object",
        "properties": {
          "title": {
```

```
            "type": "string",
            "description": "The title of the test case",
            "minLength": 1,
            "maxLength": 30
          },
          "execution_datetime": {
            "type": "string",
            "format": "date-time",
            "description": "The date and time of the execution of \
              this test case starting."
          },
          "passed": {
            "type": ["string", "null"],
            "description": "The state of the test case",
            "enum": [
              "pass",
              "fail",
              null
            ]
          },
          "custom": {
            "type": "object",
            "description": "Custom metadata values",
            "patternProperties": {
              ".+": {
                "type": "string"
              }
            }
          }
        },
        "required": ["title", "execution_datetime"]
      },
      "evidence": {
        "type": "array",
        "items": {
          "type": "object",
          "description":
            "A piece of evidence as part of this test case.",
          "properties": {
            "kind": {
              "type": "string",
              "description":
                "The Internet Media Type of the data stored.",
              "pattern": "^(\\w*)\\/([\\w\\.-]*)(\\+([\\w\\.-]*))?\
                (;((.+)=(.*);)*(.+)=(.*))?$"
            },
            "value": {
              "type": "string",
```

```
              "description": "Either ‘plain:‘ followed by plain text, \
                ‘media:‘ followed by a media SHA256 hash, or ‘base64:‘\
                followed by a base64 string of data without padding.",
              "pattern":
              "^(plain:.*)|(media:[0-9a-f]{64})|(base64:[A-z0-9+/]*)$"
            },
            "caption": {
              "type": "string",
              "description":
                "An optional caption for this piece of evidence."
            },
            "original_filename": {
              "type": "string",
              "description": "The original filename for File evidence"
            }
          },
          "required": ["kind", "value"]
        }
      }
    },
    "required": ["metadata", "evidence"]
  }
```

Authors’ Addresses

Lily Hopkins
Email: lily@hpkns.uk
URI: https://github.com/lilopkins

Eden Turner
Email: somebirb7190@gmail.com
URI: https://github.com/Some-Birb7190