Content Delivery Networks Interconnection
Internet-Draft
Intended status: Standards Track
Expires: 8 January 2026

B. Rosenblum
Vecima
O. Ramadan
Blockcast
K. Seward
Lumen

7 July 2025

CDNI Logging Extensions
draft-ietf-cdni-logging-extensions-02

Abstract

This document defines a set of extensions to CDNI for supporting transmission of transaction logs in both push and pull operational modes, new log container formats and log record formats, new logging fields, and metadata for describing the transformation, obfuscation, and encryption of log record fields.

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 8 January 2026.

Copyright Notice

Table of Contents

1. Introduction

[TODO: Add ASCII art diagram describing Pull Operational Mode]

Figure 1: Logging Component Overview, Pull Operational Mode

[TODO: Add ASCII art diagram describing Push Operational Mode]

Figure 2: Logging Component Overview, Push Operational Mode

The CDNI Logging Interface [RFC7937] defines requirements for the exchange of log data between CDN participants, including a Logging Reference Model, a Logging File Structure, and Logging Records, along with an Atom-based index for retrieving the generated log files. Open Caching expands on the functionality defined in the CDNI Logging Interface [RFC7937] to provide clear definitions and an expanded set of log record fields, additional mechanisms for transporting logs, new log container formats, log archive files, logging metadata sidecar files, new log record formats, configured inclusion of OPTIONAL request and response header fields, and methods for advertising and configuring transformations of individual log fields.

In the future, this specification will be expanded to cover other Open Caching System (OCS) component logs (e.g., request router), custom log container formats, and log record formats, as well as uCDN-controlled configuration for aggregation, filtering, and sampling of log records.

The following subsections summarize the importance of logging in the context of content acquisition and its delivery by surrogates on behalf of the CDN.

1.1. Log Generation

As described in section 2.2.1 of the CDNI Logging Interface [RFC7937], logging is generated by the downstream CDN (dCDN), which applies to the OCS in this context, to record all significant task completions, events, and failures. It is expected that all devices acting as caching nodes SHALL generate logging information, including devices performing request routing and other control functions. It

is to be noted that this does not preclude open caching nodes (OCNs) from generating and reporting transactions that are in progress and not fully completed, such as progressive downloads that may run for a longer duration and are therefore considered good candidates for partial reporting prior to fully completing a transaction. The individual implementations are, however, expected to make a determination as to when it is appropriate to generate reports for partial transactions.

1.2. Log Retention

The OCN SHALL retain logs for a specified duration. The specific duration is up to individual implementations to determine. Presumably, in the absence of logs being uploaded to or acquired by the CDN, it is possible the size of logging information would grow exponentially, therefore, it is also RECOMMENDED to define a shorter duration for upload/acquisition, such as a period of a few hours.

1.3. Accounting

Logging information is essential for accounting. The logging information provided by the OCS enables the CDNs to compute the total amount of traffic delivered by every OCN for a given content service provider (CSP), as well allow the CDNs to determine associated bandwidth usage (peak or percentile) and key performance indicators, such as maximum (or minimum) number of simultaneous sessions over a given period of time.

1.4. Analytics And Reporting

The goals of analytics include gathering all relevant information in order to be able to develop statistics on content download, analyze user behavior, and monitor the performance and quality of content delivery. For instance, logging information enables CDN providers to report on content consumption (e.g., delivered sessions per content) in a specific geographic area.

The goal of reporting is to gather all relevant information to monitor the performance and quality of content delivery, and allow detection of delivery issues. For instance, reporting could track the average delivery throughput experienced by end users in a given region for a specific CSP or content set over a period of time.

1.5. Content Protection

The goal of content protection is to monitor and prevent unauthorized access, misuse, modification, and denial of access to content. A set of information is logged in a CDN (by an OCN) for security purposes. In particular, a record of access to content is usually collected to permit the CSP to detect infringements of content delivery policies and other abnormal end-user behaviors.

1.6. Private Data Protection

As described in section 7.4 (Section 6.4), the goal of logging transforms is to modify the outputs of fields that may include private information to comply with the confidentiality requirements of both the uCDCN and dCDN. The uCDN configurations are bundled within the transport configuration to support configurations that may be specific to the jurisdictions of individual reporting endpoints and service footprints. Additionally, the dCDN can advertise field transformations that have been applied in the footprint and capabilities interface.

2. Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Container File Formats

For non-streaming logging formats, the container/file format describes the structure of the file that contains the individual log records.

The following container/file formats are supported:

* CDNI Logging File described in section 3 of [RFC7937], inspired by W3C Extended Log File Format (cdni_v1)

* JSON (json_v1)

* Newline delimited records:

  - \n (linefeed_v1)

  - \n\r (carriage_return_v1)

* Compressed Archive containers (tar.gz) that support inclusion of multiple files in other container formats (tar_v1)

* protobuf containers, according to the record type definitions in section 10 (Section 8)

3.1. JSON

The JSON container/file format consists of a single root object with file metadata, fields common to all records, and an array of log records.

Where the JSON format MUST be referenced from other configurations (e.g., in the container-format property of MI.LoggingTransport), it MAY be referred to by the string: "json_v1"

Property: shortname

* Description: A unique identifier for the dCDN.

* Format: String

* Mandatory-to-Specify: No

Property: timestamp-start-ns

* Description: The starting timestamp for the time range for which the log file was generated. This can be earlier than the timestamp of the file’s first log record.

* Format: Integer time in nanoseconds since Unix epoch

* Mandatory-to-Specify: No

Property: timestamp-start-iso8601

* Description: The starting timestamp for the time range for which the log file was generated. This can be earlier than the timestamp of the file’s first log record.

* Type: ISO-8601-1:2019

* Mandatory-to-Specify: No

Property: timestamp-end-ns

* Description: The ending timestamp for the time range for which the log file was generated. This can be later than the timestamp of the file’s last log record.

* Type: Integer time in nanoseconds since Unix epoch

* Mandatory-to-Specify: No

Property: timestamp-end-iso8601

* Description: The ending timestamp for the time range for which the log file was generated. This can be later than the timestamp of the file's last log record.

* Type: ISO-8601-1:2019

* Mandatory-to-Specify: No

Property: account-id

* Description: Account ID as defined by the account-id property of MI.LoggingMetadata.

* Type: String

* Mandatory-to-Specify: No

Property: metadata

* Description: Metadata associated with configuration of the logging records, e.g., associated record-type.

* Type: Object. Valid properties:

  - record-type: String. The record type string defining the format of included logging records. See section 6. (Section 5)

  - transforms: Object. The MI.LoggingTransforms object applied to the record set. See section 7.4. (Section 6.4)

  - secret-stores: Array. Any MI.SecretStore objects referenced from MI.LoggingTransforms.

  - include-headers: Array. Any optionally included request or response header fields. See section 10.1 (Section 6.1)

* Mandatory-to-Specify: Yes

Property: records

* Description: The log records consisting of an array wherein all properties are valid log record fields as described in Section 8. (Section 4)

* Type: Array of objects

* Mandatory-to-Specify: Yes

The following is an example of the JSON container format:

```
{
  "shortname": "exampleCDN",
  "timestamp-start-iso8601": "2023-01-15T14:00:00Z",
  "timestamp-end-iso8601": "2023-01-15T14:59:59Z",
  "account-id": "foobarMedia",
  "metadata": {
    "record-type": "opencaching_standard_json_v1",
    "transforms": [
      {
        "record-fields": [
                 "c-ip"
             ],
        "operations": [
                 {
            "type": "MI.LoggingTransformMaskIp",
            "value": {
              "mask-lsb-v4": 4,
              "mask-lsb-v6": 16
            }
          }
             ]
      }
    ]
  },
  "records": [
    {
      "timestamp-ns": 1672938865000000000,
      "cs-uri": "https://cdn.example.com/assets/b874bc5b/index.m3u8",
      "sc-status": "200",
      "sc-total-bytes": 1784,
      "s-upstream-bytes": 1784
    }
  ]
}
```

Figure 3

3.1.1. JSON Metadata Sidecar

A metadata sidecar file MAY be used as a companion to other record container files, allowing detailed metadata as specified in the JSON container format.

Metadata sidecars are identical to the full JSON container format except that the records property is NOT included.

Sidecar files are identified by a file naming convention; the corresponding sidecar for a particular log file has the same name as the log file with an appended suffix of ".metadata.json".

For example, a log file named "2023-01-01-12_00_00-region1.csv" would have a corresponding metadata sidecar file named "2023-01-01_12_00_00-region1.csv.metadata.json".

dCDN support for metadata sidecars is indicated with the allow-metadata-sidecar property of FCI.LoggingPush Use of metadata sidecars for a particular configuration is indicated by the include-metadata-sidecar boolean property in MI.LoggingTransport and FCI.LoggingPullTransport See section 7.3 (Section 6.3) and section 8.3 (Section 7.3)

3.2. Newline Delimited Records

format-type	Description
linefeed_v1	Log records separated by linefeed (\n).
carriage_return_v1	Log records separated by linefeed, carriage return (\n\r).

Table 1

3.3. Archive Containers

File-based transport modes MAY bundle a set of log files together in a gzipped tar archive. When utilizing tar_v1 as the container-type, the properties of the MI.LoggingContainerMetadata object refer to the generation of the archive file, and the archive-metadata property of that object contains another MI.LoggingContainerMetadata object that specifies the configuration for the individual files within that archive.

For example, if the archive container is specified with an interval of 86400, and the inner archive-metadata has an interval of 3600, a tar.gz archive will be produced once a day containing 24 individual log files, each containing one hour of log records.

3.4. Protobuf Containers

Each log MAY be serialized using protobuf protocol buffers. This document provides the protobuf definition languages (.proto files) for different record types in Appendix section 13.1 (Section 8.1)

4. Record Fields

The following table describes the log record fields that are included in each of the standard log record formats. Y - Yes, N - No, O - Optional.

Field	Minimal	Standard	Extended
timestamp-ns	Y	Y	Y
timestamp-iso8601	N	N	Y
s-time-total-ms	N	Y	Y
c-groupid	N	N	Y
cs-method	N	N	Y
cs-version	N	N	Y
cs-uri	Y	Y	Y
cs-uri-transformed	N	N	Y
sc-status	Y	Y	Y
sc-total-bytes	Y	Y	Y
ccid	N	N	Y
s-sid	N	Y	Y
s-cached	N	Y	Y
cs-hdr-Referer	N	N	Y
cs-hdr-Range	N	N	Y
cs-hdr-User-Agent	N	Y	Y
sc-hdr-Content-Type	N	N	Y

sc-header-size-bytes	N	N	Y
c-ip	N	Y	Y
s-shortname	N	Y	Y
s-id	N	N	Y
s-time-first-ms	N	N	Y
s-sdur-ms	N	N	Y
s-time-upstream-ms	N	N	Y
s-upstream-bytes	Y	Y	Y
sc-hdr-*, cs-hdr-*	O	O	O

Table 2

4.1. Fields

4.1.1. timestamp-ns

* Description: The timestamp of the first byte read by the OCN application to a best effort of accuracy.

* Format: Integer time in nanoseconds since Unix epoch

* Record Types: Minimal, Standard, Extended

4.1.2. timestamp-iso8601

* Description: The timestamp of the first byte read by the OCN application to a best effort of accuracy. It is RECOMMENDED that timestamps use Coordinated Universal Time (UTC).

* Format: Time in ISO-8601/RFC3339 format, milliseconds OPTIONAL (e.g., 1985-04-12T23:20:50.52Z)

* Record Types: Extended

4.1.3. s-time-total-ms

* Description: The elapsed time between reading the first byte of the request and sending the last byte of the response to a best effort of accuracy.
* Format: Duration, in milliseconds
* Record Types: Standard, Extended

4.1.4. c-groupid

* Description: An opaque identifier for an aggregate set of clients, derived from the client IPv4 or IPv6 address in the request received by the surrogate and/or other network-level identifying information. See [RFC7937] section 3.4.1.
* Format: String
* Record Types: Minimal, Standard, Extended

4.1.5. cs-method

* Description: The HTTP request method.
* Format: String
* Record Types: Extended

4.1.6. cs-version

* Description: The protocol version (e.g., HTTP/1.1).
* Format: String
* Record Types: Extended

4.1.7. cs-uri

* Description: Request URI prior to any transformation from MI.ProcessingStages. It is RECOMMENDED to limit the maximum length of this field with an applied MI.LoggingTransformTruncate, allowing at least 2,048 bytes.
* Format: String
* Record Types: Minimal, Standard, Extended

4.1.8. cs-uri-transformed

* Description: Full request URI, including query parameters. If this field is identical to cs-uri, the field SHOULD be truncated to an empty string.
* Format: String
* Record Types: Extended

4.1.9. sc-status

* Description: The response status code.
* Format: String
* Record Types: Minimal, Standard, Extended

4.1.10. sc-total-bytes

* Description: The response size total bytes, including headers.
* Format: Integer
* Record Types: Minimal, Standard, Extended

4.1.11. ccid

* Description: The ccid assigned via MI.Grouping or MI.GroupingExtended configuration.
* Format: String
* Record Types: Minimal, Standard, Extended

4.1.12. s-sid

* Description: The Session ID assigned by the dCDN. This can be derived from Common Media Client Data (CMCD) or another mechanism, but it is defined by the dCDN, and uCDN allocated session identifiers are outside the scope of this field.
* Format: String
* Record Types: Standard, Extended

4.1.13. s-cached

* Description: Indicates a cache hit or cache miss. For this field, a cache hit is defined as a request that does not result in content retrieval from the upstream source.

* Format: Boolean; a value of "True" indicates a cache hit.

* Record Types: Standard, Extended

4.1.14. cs-hdr-*

* Description: The request headers as available from the dCDN and specified in FCI.Logging In the standard record type, only the cs-hdr-User-Agent header is included, if available. The extended record type additionally includes cs-hdr-Range and cs-hdr-Referer See the Record Fields Table (Section 4) for reference.

* Format: String or null if the field does not exist (or is not supported by record encoding)

  - Special care needs to be taken while composing the value of these fields as the requested HTTP header name MAY be present multiple times in the request. When such cases are encountered, implementations are expected to follow the semantics defined for combining HTTP field values in [RFC9110] sections 5.2 and 5.3, in order to compose the final value to be logged for this field in the record.

* Record Types: Standard, Extended

4.1.15. sc-hdr-*

* Description: The response headers as available from the dCDN and specified in FCI.Logging In the extended type, sc-hdr-Content-Type is always included. See the Record Fields Table (Section 4) for reference.

* Format: String or null if the field does not exist (or is not supported by record encoding)

  - Special care needs to be taken while composing the value of these fields as the requested HTTP header name MAY be present multiple times in the response. When such cases are encountered, implementations are expected to follow the semantics defined for combining HTTP field values in [RFC9110] sections 5.2 and 5.3, in order to compose the final value to be logged for this field in the record.

* Record Types: Standard, Extended

4.1.16. sc-header-size-bytes

* Description: The response header size bytes.

* Format: Integer

* Record Types: Extended

4.1.17. c-ip

* Description: The client IP address.

* Format: String

* Record Types: Standard, Extended

4.1.18. s-shortname

* Description: A short name for the dCDN.

* Format: String

* Record Types: Standard, Extended

4.1.19. s-id

* Description: The server instance ID assigned by the dCDN. This value SHOULD be unique to a particular instance.

* Format: String

* Record Types: Extended

4.1.20. s-time-first-ms

* Description: The elapsed time from the first request byte read until the first response byte sent.

* Format: Integer duration, in milliseconds

* Record Types: Extended

4.1.21. s-sdur-ms

* Description: The session duration as determined by the dCDN.

* Format: Integer duration, in milliseconds

* Record Types: Extended

4.1.22. s-time-upstream-ms

* Description: The time spent accessing the upstream source.

* Format: Integer duration, in milliseconds. If no upstream is accessed, the duration is 0.

* Record Types: Extended

4.1.23. s-upstream-bytes

* Description: The number of bytes transferred from the upstream source, including headers.

* Format: Integer, size in bytes. If no upstream is accessed, the size is 0.

* Record Types: Minimal, Standard, Extended

5. Record Types

Standard record types consist of a field set, a format, and a version that together form the record type, identified by a string like: "opencaching_minimal_json_v1" or "opencaching_standard_csv_v1" or "opencaching_extended_whitespace_v1".

The identifier is then referenced in the supported-record-types field of FCI.Logging and the record-type field of MI.LoggingTransport.

For this draft, we do not support custom record types, but future drafts will provide a way to define custom record types with their own associated identifiers.

5.1. Log Record Formats

5.1.1. JSON

The JSON log record format consists of a JSON object wherein each defined field of the record type is a property of the JSON object; the field name is the property name and the field value is the property value. If a field is not present in the log data, the property MAY be omitted.

The following is an example record in the JSON record format:

```
{
  "timestamp-ns": 1672938865000000000,
  "cs-uri": "https://cdn.example.com/assets/b874bc5b/index.m3u8",
  "sc-status": "200",
  "sc-total-bytes": 1784,
  "s-upstream-bytes": 1784
}
```

Figure 4

5.1.2. CSV

The comma-separated value record format follows the definition in [RFC4180] with the addition of a special value to handle null values.

If a field value is null, it MUST be represented by the following unquoted literal value: $NULL$

5.1.3. Whitespace

The whitespace delimited record format consists of field values enclosed in QUOTATION MARK (U+0022: ") characters, with characters separated by any number of SPACE (U+0020) or TAB (U+0009) characters.

If a field value is null, it must be represented by the following unquoted literal value: $NULL$

5.1.4. Protobuf

Appendix section 10.1 (Section 8.1) provides the protobuf definitions in the proto file format for the protobuf record types defined.

5.2. Predefined Log Record Types

* A predefined log record type is a set of fields in a defined textual or binary format (JSON, CSV, whitespace delimited, or protobuf). This specification includes three predefined record types:
  - A ‘minimal’ type optimized for data size, including only the fields necessary to perform billing functions.
  - A ‘standard’ type with a set of fields likely to be useful for operational monitoring.
  - An ‘extended’ type with an exhaustive field set likely to cover most operational and troubleshooting use cases.

5.2.1. Minimal

Order	Field
1	timestamp-ns
4	c-groupid
7	cs-uri
9	sc-status
10	sc-total-bytes
11	ccid
20	s-upstream-bytes
24	sc-hdr-*, cs-hdr-*

Table 3

The Minimal log record type field set consists of the fields denoted as 'Minimal' in section 8. (Section 4)

* JSON format: 'opencaching_minimal_json_v1'
* CSV format: 'opencaching_minimal_csv_v1'
* Whitespace delimited format:'opencaching_minimal_whitespace_v1'
* Protobuf format as defined in section 10 (Section 8)

5.2.2. Standard

Order	Field
1	timestamp-ns
2	s-time-total-ms
4	c-groupid
7	cs-uri

9	sc-status
10	sc-total-bytes
11	ccid
12	s-sid
13	s-cached
15	s-shortname
20	s-upstream-bytes
24	cs-hdr-User-Agent
25+	sc-hdr-*, cs-hdr-*

Table 4

The Standard log record type field set consists of the fields denoted as 'Standard' in section 8. (Section 4)

* JSON format: 'opencaching_standard_json_v1'

* CSV format: 'opencaching_standard_csv_v1'

* Whitespace delimited format: 'opencaching_standard_whitespace_v1'

* Protobuf format as defined in section 10 (Section 8)

5.2.3. Extended

Order	Field
1	timestamp-ns
2	timestamp-iso8601
3	s-time-total-ms
4	c-groupid
5	cs-method
6	cs-version

7	cs-uri
8	cs-uri-transformed
9	sc-status
10	sc-total-bytes
11	ccid
12	s-sid
13	s-cached
14	c-ip
15	s-shortname
16	s-id
17	s-time-first-ms
18	s-sdur-ms
19	s-time-upstream-ms
20	s-upstream-bytes
21	sc-header-size-bytes
22	cs-hdr-Referer
23	cs-hdr-Range
24	cs-hdr-User-Agent
25	sc-hdr-Content-Type
26+	sc-hdr-*, cs-hdr-*

Table 5

The Extended log record type field set consists of the fields denoted as ’Extended’ in section 8. (Section 4)

* JSON format: ’opencaching_extended_json_v1’

* CSV format: ’opencaching_extended_csv_v1’

* Whitespace delimited format: ’opencaching_extended_whitespace_v1’

* Protobuf format as defined in section 10 (Section 8)

5.2.4. Legacy

For backwards compatibility, the Legacy type covers the record type as defined in the Open Caching Logging Integration Functional Specification (2017) [OC-LIFS] based on the Squid proxy log format.

The Legacy record type, if used, MUST be referenced as: ’opencaching_legacy_2017’.

5.2.5. uCDN Configured Header Inclusion

If the dCDN advertises the allow-include-headers property in FCI.LoggingPush (see section 11.1 (Section 7.1)), a uCDN can configure additional request/response headers as custom fields to be included with the MI.LoggingTransport property include-headers (see section 10. (Section 6.3)3 (Section 6.3)). Included headers are added to the log records in the order defined, with names cs-hdr-* and sc-hdr-*, to indicate request and response headers respectively.

The following is an example of MI.LoggingMetadata that includes headers:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transports": [{
                    "type": "MI.LoggingTransportKafka",
                    "record-type": "opencaching_standard_json_v1",
                    "include-headers": ["cs-hdr-Range", "sc-hdr-Content-Length"],
                    "account-id": "dCDN1",
                    "endpoint": {
                            "bootstrap-servers": ["kafka.ucdn.example.com:9092"],
                            "topic": "dcdn-logs"
                    }
            }]
    }
}
```

Figure 5

5.3. Custom Record Types

It is permitted with the out-of-band agreement of the participants to use a record type not defined in this specification. In this case, the format of the record type is entirely dependent on cooperation between the uCDN and dCDN outside the scope of the Open Caching API. A unique identifier SHOULD be assigned for every custom record type, and it MAY be utilized in place of record types defined in this specification wherever a record-type value is required.

6. Logging Metadata

The MI.LoggingMetadata object provides configuration that instructs the dCDN how to transmit log files or streams of log records to the uCDN.

MI.LoggingTransport objects are enclosed in the MI.LoggingMetadata object described below.

The dCDN specifies supported logging transports in the FCI.LoggingSource advertisement as specified in Section 8.2. (Section 7.2)

6.1. MI.LoggingMetadata

Property: transports

* Description: An array of MI.LoggingTransport objects.
* Format: Array
* Record Types: No

Property: pull-transport-metadata

* Description: An object that can be used to configure certain properties of dCDN pull transports.
* Type: MI.LoggingPullTransportMetadata
* Mandatory-to-Specify: No

6.2. MI.LoggingPullTransportMetadata

Property: sftp-public-keys

* Description: An array of SSH public keys that the dCDN SHOULD authorize on SFTP logging endpoints.

* Type: Array of strings

* Mandatory-to-Specify: No

6.3. MI.LoggingTransport

MI.LoggingTransport objects are enclosed in the MI.LoggingMetadata object and instruct the dCDN how to transmit log files or streams to the uCDN.

Property: transport-type

* Description: The type discriminator for the endpoint property. The following values are supported:

  - MI.LoggingTransportKafka

  - MI.LoggingTransportSFTP

  - MI.LoggingTransportS3

* Type: String

* Mandatory-to-Specify: Yes

Property: record-type

* Description: The format of the individual log records. This MUST be a valid value as defined in Section 9 (Section 5)

* Type: String

* Mandatory-to-Specify: Yes

Property: container-metadata

* Description: The configuration for the generation of log files when operating in file-based modes.

* Type: MI.LoggingContainerMetadata

* Mandatory-to-Specify:

  - File or object-based transports: Yes

  - Streaming transports: No

Property: include-headers

* Description: Additional header fields to include, dependent on dCDN support via the allow-include-headers flag of MI.LoggingMetadata.

* Type: Ordered array of header fields in sc-hdr-* and cs-hdr-* naming format

* Mandatory-to-Specify: No

Property: account-id

* Description: An account identifier value that the dCDN must include in log records.

* Type: String

* Mandatory-to-Specify: No

Property: endpoint

* Description: An object with any configuration necessary for the specific logging transport, discriminated on the type property. Objects for each transport type are described in the sections below.

* Type: Object

* Mandatory-to-Specify: Yes

Property: transforms

* Description: Transformations on sets of fields as defined in an MI.LoggingTransform. The sets of record-fields field names MUST NOT overlap for the transport.

* Type: Array of MI.LoggingTransforms objects

* Mandatory-to-Specify: No

6.3.1. MI.LoggingTransportKafka

Property: bootstrap-servers

* Description: An array of the bootstrap connection servers in host:port format.

* Type: Array of strings

* Mandatory-to-Specify: Yes

Property: topic

* Description: The Kafka topic to which log records SHOULD be published.

* Type: String

* Mandatory-to-Specify: Yes

The following is an MI.LoggingMetadata Apache Kafka example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transports": [{
                    "transport-type": "MI.LoggingTransportKafka",
                    "record-type": "opencaching_standard_json_v1",
                    "account-id": "dCDN1",
                    "endpoint": {
                            "bootstrap-servers": ["kafka.ucdn.example.com:9092"],
                            "topic": "dcdn-logs"
                    }
            }]
    }
}
```

Figure 6

6.3.2. MI.LoggingTransportSFTP

Property: host

* Description: The hostname of the SFTP server, which MAY be specified as either hostname or hostname:port.

* Type: String

* Mandatory-to-Specify: Yes

Property: username

* Description: The remote username to use for connecting to the SFTP server.

* Type: String

* Mandatory-to-Specify: Yes

Property: password

* Description: The remote password to use for connecting to the SFTP server when using password-based authentication.

* Type: MI.SecretValue

* Mandatory-to-Specify: No

The following is an MI.LoggingTransportSFTP example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transports": [{
                    "type": "MI.LoggingTransportSFTP",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "filename-template": "uCDN/logs/%Y-%m/%d-access.log.%n",
                            "interval": 86400,
                            "maximum-size": "100M"
                    },
                    "account-id": "dCDN1",
                    "endpoint": {
                            "host": "logs.ucdn.example.com:22",
                            "username": "dcdnlogs"
                    }
            }]
    }
}
```

Figure 7

6.3.3. MI.LoggingTransportS3API

Property: host

* Description: The hostname to connect to the S3-compatible API.

* Type: String

* Mandatory-to-Specify: Yes

Property: access-key-id

* Description: The access key ID to use when connecting to the API.

* Type: String

* Mandatory-to-Specify: No

Property: access-key-secret

* Description: The access key secret to use when connecting to the API.

* Type: MI.SecretValue

* Mandatory-to-Specify: No

Property: bucket-name

* Description: The bucket name to use for pushed log file objects.

* Type: String

* Mandatory-to-Specify: Yes

The following is an MI.LoggingTransportS3API example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transports": [{
                    "type": "MI.LoggingTransportS3API",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "name-template": "%Y-%m-%d-access.log",
                            "interval": 86400
                    },
                    "account-id": "dCDN1",
                    "endpoint": {
                            "host": "s3.amazonaws.com",
                            "access-key-id": "xxxxxxxxx",
                            "access-key-secret": {
                                    "secret-store-id": "store-1",
                                    "secret-store-path": "/logging/dCDN1/dcdnlogs/s3"
                            },
                            "bucket-name": "dcdnlogs"
                    }
            }]
    }
}
```

Figure 8

6.4. MI.LoggingTransforms

Logging fields can contain personal user information that cannot be shared upstream or stored in the operating jurisdictions. These operations allow for obfuscation and anonymization of logging record fields.

An array of MI.LoggingTransforms objects can be enclosed in the MI.LoggingTransport object transform field for uCDN configured obfuscation.

The dCDN specifies privacy obfuscations in the FCI.Logging advertisement applied-transforms field.

Property: record-fields

* Description: The list of privacy-sensitive log record fields for which obfuscation is applied.

* Type: Array of strings

* Mandatory-to-Specify: Yes

Property: transforms

* Description: The transform operations to apply on the record fields, in order.

* Type: Array of MI.LoggingTransform objects

* Mandatory-to-Specify: Yes

6.4.1. MI.LoggingTransform

Property: type

* Description: The type discriminator for the value property. The following values are supported:

  - MI.LoggingTransformReplace

  - MI.LoggingTransformTruncate

  - MI.LoggingTransformUrlRemoveParam

  - MI.LoggingTransformUrlStripParams

  - MI.LoggingTransformMaskIp

  - MI.LoggingTransformEncrypt

  - MI.LoggingTransformHash

  - MI.LoggingTransformEncode

* Type: String

* Mandatory-to-Specify: Yes

Property: value

* Description: The privacy operation configuration value.

* Type: Object of type

* Mandatory-to-Specify: Yes

6.4.2. MI.LoggingTransformReplace

Property: regex

* Description: A POSIX substitution expression.
* Type: String
* Mandatory-to-Specify: Yes

Property: value

* Description: The privacy operation configuration value.
* Type: Object of type
* Mandatory-to-Specify: Yes

6.4.3. MI.LoggingTransformTruncate

This object removes characters from the end of a field value if it exceeds a defined length.

Property: length

* Description: The maximum field length.
* Type: Number
* Mandatory-to-Specify: Yes

6.4.4. MI.LoggingTransformUrlRemoveParam

This object parses record-fields values as URLs, stripping specified query parameters along with their values.

Property: remove-param

* Description: Omits URL query parameters whose name match a regular expression.
* Type: POSIX Regular expression string
* Mandatory-to-Specify: Yes

6.4.5. MI.LoggingTransformUrlStripParams

Property: strip-params

* Description: Strips all query parameters from the URL.

* Type: Boolean

* Mandatory-to-Specify: Yes

The following is an MI.LoggingTransformUrlParam example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transforms": [{
                    "record-fields": [
                            "c-uri",
                            "c-hdr-Referer"
                    ],
                    "operations": [{
                            "type": "MI.LoggingTransformUrlParam",
                            "value": {
                                    "replace-param": "^.?[iI][dD]$|.*_[iI][dD]$",
                                    "strip-params": False
                            }
                    }]
            }]
    }
}
```

Figure 9

6.4.6. MI.LoggingTransformUrlPath

This object parses record-fields values as URLs, and can replace or truncate path elements from the front or end. At least one of replace-path or path-trim MUST be present.

Property: replace-path

* Description: Modify URL path elements with a regular expression.

* Type: String. POSIX

* Mandatory-to-Specify: No, if URL parameter modification is not desired.

Property: path-trim

* Description: Trims elements from URL paths to a specified depth. A positive value will truncate the specified number of path elements from the end. A negative value will truncate from the front.

* Type: Number

* Mandatory-to-Specify: No, if path truncation is not desired.

The following is an MI.LoggingTransformUrlPath example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transforms": [{
                    "record-fields": [
                            "c-uri",
                            "c-hdr-Referer"
                    ],
                    "operations": [{
                            "type": "MI.LoggingTransformUrlPath",
                            "value": {
                                    "path-trim": -1,
                                    "replace-path": "s/sid=[^\/]+/sid-XXXX/g"
                            }
                    }]
            }]
    }
}
```

Figure 10

6.4.7. MI.LoggingTransformMaskIp

This object parses values as IPv4 or IPv6 strings, and outputs an address that strips the indicated number of bits.

Property: mask-lsb-v4

* Description: The number of bits to strip from the end of an IPv4 address.

* Type: Number, maximum 32 for IPv4

* Mandatory-to-Specify: No, if IPv4 masking is not desired

Property: mask-lsb-v6

* Description: The number of bits to strip from the end of an IPv6 address.

* Type: Number, maximum 128 for IPv6

* Mandatory-to-Specify: No, if IPv6 masking is not desired

The following is an MI.LoggingTransformMaskIp example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transforms": [{
                    "record-fields": [
                            "c-ip"
                    ],
                    "transforms": [{
                            "type": "MI.LoggingTransformMaskIp",
                            "value": {
                                    "mask-lsb-v4": 4,
                                    "mask-lsb-v6": 16
                            }
                    }]
            }]
    }
}
```

Figure 11

6.4.8. MI.LoggingTransformHash

This object replaces record-fields values with their Hash-Based Message Authentication Code (HMAC).

Property: function

* Description: The hashing function to use in computing the HMAC of the private field value.

* Type: String. The following values are supported:

  - MD5

  - SHA256

* Mandatory-to-Specify: Yes

Property: key

* Description: The key used in computing the HMAC.

* Type: MI.SecretValue

* Mandatory-to-Specify: No, if no key is used or an empty string, if the key is confidential to the dCDN

The following is an MI.LoggingTransformHash example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transforms": [{
                    "record-fields": [
                            "c-ip"
                    ],
                    "transforms": [{
                            "type": "MI.LoggingTransformHash",
                            "value": {
                                    "function": "MD5",
                                    "key": {
                                            "secret-store-id": "store-1",
                                            "secret-path": "/logging/hmac/c-ip"
                                    }
                            }
                    }]
            }]
    }
}
```

Figure 12

6.4.9. MI.LoggingTransformEncrypt

With this object, fields are encrypted with the selected scheme, and the nonce used is prepended to the outputted ciphertext.

The encryption key is specified as an MI.SecretValue and is transmitted in the transforms field with log records in the metadata header or as a sidecar. When a new key is configured, a new file and sidecar must be created with an updated key reference.

Property: algorithm

* Description: The algorithm to use in encrypting the private field value.

* Type: String. The following values are supported:
  - AEAD-XCHACHA20-POLY1305-IETF
  - AEAD-CHACHA20-POLY1305-IETF
  - AEAD-AES256-GCM
  - RC4
* Mandatory-to-Specify: Yes

Property: key

* Description: A reference to an externally defined encryption key.
* Type: MI.SecretValue
* Mandatory-to-Specify: No

The following is an MI.LoggingTransformEncrypt example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transforms": [{
                    "record-fields": [
                            "c-ip"
                    ],
                    "operations": [{
                            "type": "MI.LoggingTransformEncrypt",
                            "value": {
                                    "algorithm": "AEAD-XCHACHA20-POLY1305-IETF",
                                    "key": {
                                            "secret-store-id": "store-1",
                                            "secret-path": "/logging/encrypt/c-ip"
                                    }
                            }
                    }]
            }]
    }
}
```

Figure 13

6.4.10. MI.LoggingTransformEncode

Property: encoding

* Description: The encoding to use for the obfuscated value.

* Type: String. The following values are supported:

  - BASE64

  - BIN2HEX

* Mandatory-to-Specify: Yes

The following is an MI.LoggingTransformEncode example:

```
{
    "generic-metadata-type": "MI.LoggingMetadata",
    "generic-metadata-value": {
            "transports": [{
                    "type": "MI.LoggingTransportS3API",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "name-template": "%Y-%m-%d-access.log",
                            "interval": 86400
                    },
                    "endpoint": {
                            "host": "s3.amazonaws.com",
                            "bucket-name": "dcdnlogs"
                    },
                    "transforms": [
                            {
                                "generic-metadata-type": "MI.LoggingTransforms",
                                "generic-metadata-value": {
                                        "transforms": [{
                                                "record-fields": [
                                                        "c-ip"
                                                ],
                                                "operations": [{
                                                        "type": "MI.LoggingTra
nsformEncrypt",
                                                        "value": {
                                                                "algorithm": "
RC4",
                                                                "key": {
                                                                        "secre
t-store-id": "store-1",
                                                                        "secre
t-path": "/logging/encrypt/c-ip"
                                                                },
                                                                "id": 622,
```

```
                                                                "wrap-id": tru
e
                                                            }
                                                        },
                                                        {
                                                            "type": "MI.LoggingTra
nsformEncode",
                                                            "value": {
                                                                "encoding": "B
ASE64"
                                                            }
                                                        }
                                                    ]
                                                }]
                                            }
                                        }
                                    ]
                                }]
                            }
                        }
```

Figure 14

6.5. MI.LoggingFilenameTemplate

The MI.LoggingFilenameTemplate type is used to specify the path and filename for files created by batch mode MI.LoggingTransport and FCI.LoggingTransport services.

If the resulting filename is longer than 255 characters, it will be truncated to 255 characters in length, including the filename extension that will be retained. Fields that MAY be truncated are noted in the table below.

It is the responsibility of the dCDN to ensure that file names are unique. It is strongly RECOMMENDED that the %id field is used to ensure the uniqueness requirement is satisfied.

The overall path length is limited to 32 KiB, and each path component is limited to 255 characters. Any element exceeding the length limitation will be truncated to the maximum length.

An instance of MI.LoggingFilenameTemplate is a string with the following supported substitution variables:

Variable	Definition	Length	Values
%cY	File creation timestamp: Year.	4	Example: 2023

%cM	File creation timestamp: Month. Numeric. Length 2. 01-12.	2	01-12
%cD	File creation timestamp: Day. Length 2. 0-31.	2	0-31
%cH	File creation timestamp: Hour. Length 2. 00-23.	2	00-23
%cM	File creation timestamp: Minutes. Length 2. 00-59.	2	00-59
%cS	File creation timestamp: Seconds. 00-59.	2	0-59
%sY	Timestamp of first record: Year. Length 4. Example: 2023.	4	Example: 2023
%sM	Timestamp of first record: Month. Numeric. Length 2. 01-12.	2	01-12
%sD	Timestamp of first record: Day. Length 2. 0-31.	2	00-31
%sH	Timestamp of first record: Hour. Length 2. 00-23.	2	00-23
%sM	Timestamp of first record: Minutes. Length 2. 00-59.	2	00-59
%sS	Timestamp of first record: Seconds. 00-59.	2	00-59
%eY	Timestamp of last record: Year. Length 4. Example: 2023.	4	Example: 2023
%eM	Timestamp of last record: Month. Numeric. Length 2. 01-12.	2	01-12
%eD	Timestamp of last record: Day. Length 2. 0-31.	2	0-31
%eH	Timestamp of last record:	2	00-23

	Hour. Length 2. 00-23.		
%eM	Timestamp of last record: Minutes. Length 2. 00-59.	2	00-59
%eS	Timestamp of last record: Seconds. 00-59.	2	00-59
%source	Logging source identifier.MAY be truncated.	Variable	Alphanumeric:[A-Za-z0-9]+
%account	Account ID as configured via MI.LoggingMetadata. MAY be truncated.	Variable	Alphanumeric: [A-Za-z0-9\-]+
%id	A unique identifier for this file. MAY be truncated.	Variable	Alphanumeric: [A-Za-z0-9\-]+
%ccid	ccid assigned via MI.Grouping configuration. MAY be truncated.	Variable	Alphanumeric: [A-Za-z0-9\-]+

Table 6

6.5.1. MI.LoggingFilenameTemplate Examples

Template	Example Filename
/%cY-%cM-%cD/%cH_%cM_%cS.%source.%id.json	/2023-01-21/11_23_58.region1.fbdc303a.json
/%source/%id.csv	/region1/fbdc303a.csv

Table 7

6.6. MI.LoggingContainerMetadata

This object defines configuration parameters used in file-based logging transports to define how files are created and formatted.

Property: container-type

* Description: The container type as defined in section 4 (Section 3)

* Type: String. The following values are supported:
  - cdni_v1
  - json_v1
  - linefeed_v1
  - carriage_return_v1
  - tar_v1
* Mandatory-to-Specify: Yes

Property: filename-template

* Description: The template to be used for files pushed by a batch mode transport.
* Type: MI.LoggingFilenameTemplate
* Mandatory-to-Specify: No

Property: include-metadata-sidecar

* Description: Used to specify if a metadata sidecar file is generated for each log file. See section 7.1.1 (Section 3.1.1)
* Type: Boolean
* Mandatory-to-Specify: No

Property: interval

* Description: For file-based logging transports, the time period for which a single log file SHOULD cover before being pushed to the endpoint, with the duration in seconds.
* Type: Integer
* Mandatory-to-Specify: No

Property: maximum-size

* Description: For batch-type logging transports, the maximum file or object size before the log file SHOULD be pushed to the endpoint and a new file created for further entries. The value of the format is "<Integer><Specifier>" where Specifier is one of the following:

  - M: megabyte

  - G: gigabyte

* Type: String

* Mandatory-to-Specify: No

Property: archive-metadata

* Description: Used with the tar_v1 container type, it specifies the container metadata for files inside of the archive.

* Type: MI.LoggingContainerMetadata

* Mandatory-to-Specify: No

7. Logging Capabilities

7.1. FCI.LoggingPush

The FCI.Logging object described in [RFC8008] is restricted to announcing available record types and OPTIONAL fields supported by the dCDN from those listed in [RFC7937].

This specification includes more record types that can be used by the uCDN in the log configuration as well as other possible metadata that can be used as container formats and possible pull transport methods.

To permit the dCDN to announce multiple record types, container formats, and transport methods for the same footprint, this capability extends that described in [RFC8008] with a new capability dedicated to the push mechanisms.

The following is an FCI.LoggingPush example:

```
{
    "capability-type": "FCI.LoggingPush",
    "capability-value": {
            "record-types": [
                    "opencaching_minimal_json_v1",
                    "opencaching_extended_whitespace_v1"
            ],
            "allow-include-headers": true,
            "container-formats": ["json_v1", "carriage_return_v1"],
            "applied-transforms": [{
                    "record-fields": [
                            "c-ip"
                    ],
                    "operations": [{
                            "type": "MI.LoggingTransformMaskIp",
                            "value": {
                                    "mask-lsb-v4": 4,
                                    "mask-lsb-v6": 16
                            }
                    }]
            }]
    }
}
```

Figure 15

Property: record-types

* Description: The supported list of CDNI logging record-types.

* Type: Array of strings corresponding to an entry from the "CDNI logging record-types" registry [RFC7937], or this specification

* Mandatory-to-Specify: Yes

Property: container-formats

* Description: The supported list of container formats according to this specification.

* Type: Array of strings corresponding to available container formats

* Mandatory-to-Specify: No. If only streaming transports are used, container formats are not used. If not included, no non-streaming containers would be available.

Property: allow-include-headers

* Description: A flag indicating support for custom headers configured by the uCDN for inclusion in the log records.

* Type: Boolean

* Mandatory-to-Specify: No

Property: applied-transforms

* Description: Transforms applied by the dCDN on all logs to the upstream. The sets of record-fields field names MUST NOT overlap for the transport.

* Type: Array of MI.LoggingTransforms objects

* Mandatory-to-Specify: No

Property: sftp-public-keys

7.2. FCI.LoggingSource

A dCDN can provide information on available logging endpoints to the uCDN that do not require metadata configuration. The FCI.LoggingSource capability object can be announced with the relevant information for the uCDN to get the log from the dCDN.

The dCDN could provide different endpoints, depending on footprint and/or delegated hosts. It could be possible that the sources in FCI.LoggingSource in the dCDN are not available for all delegated hosts or all available footprints, permitting flexibility and control of its own infrastructure (e.g., a dCDN could limit logging endpoints to suitable hosts included in an agreement between a dCDN and uCDN).

The following is an FCI.LoggingSource example that shows use of an S3 bucket for retrieving JSON formatted logs in a JSON container:

```
{
    "capability-type": "FCI.LoggingSource",
    "capability-value": {
            "hosts": [
                    "a.service123.ucdn.example.com",
                    "b.service123.ucdn.example.com"
            ],
            "transports": [{
                    "type": "FCI.LoggingPullTransportS3",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "filename-template": "%Y-%m-%d-access.%i.log",
                            "interval": 86400
                    },
                    "account-id": "dCDN1",
                    "endpoint": {
                            "host": "s3.amazonaws.com",
                            "access-key-id": "xxxxxxxxx",
                            "bucket-name": "dcdnlogs"
                    }
            }]
    }
}
```

Figure 16

Property: hosts

* Description: One or more uCDN hosts for which the logs are available for pull logging transports. A redirecting host SHOULD be a host that was published in an MI.HostMatch object by the uCDN as defined in section 4.1.2 of [RFC8006].

* Type: A list of endpoint objects (see section 4.3.3 of [RFC8006])

* Mandatory-to-Specify:No. If absent or empty, the logging endpoint is available for all hosts of the redirecting uCDN.

Property: transports

* Description: List of available sources for an uCDN to obtain the logs for its delegated services

* Type: Array of FCI.LoggingPullTransport objects

* Mandatory-to-Specify: No

7.3. FCI.LoggingPullTransport

Property: transport-type

* Description: The type discriminator for the endpoint property. The following values are supported:
  - FCI.LoggingPullTransportAtomFeed
  - FCI.LoggingPullTransportSFTP
  - FCI.LoggingPullTransportS3
* Type: String
* Mandatory-to-Specify: Yes

Property: record-type

* Description: The format of the individual log records. This must be a valid value as defined in section 5. (Section 4)
* Type: String
* Mandatory-to-Specify: Yes

Property: container-metadata

* Description: The configuration for the generation of log files when operating in file-based modes.
* Type: MI.LoggingContainerMetadata
* Mandatory-to-Specify:
  - File or object-based transports: Yes
  - Streaming transports: No

Property: transforms

* Description: A list of transformations to be applied to sets of record fields before transport. A record field name SHOULD NOT appear more than once.
* Type: Array of MI.LoggingTransforms objects
* Mandatory-to-Specify: No

Property: endpoint

* Description: An object with any configuration necessary for the specific logging transport, discriminated on the type property. Objects for each transport type are described below.

* Type: Object

* Mandatory-to-Specify: Yes

7.4. FCI.LoggingPullTransportAtomFeed

Property: feed-href

* Description: The URL of the CDNI logging feed in Atom format per [RFC7937].

* Type: String

* Mandatory-to-Specify: Yes

The following is an FCI.LoggingSourceAtomFeed example:

```
{
    "capability-type": "FCI.LoggingSource",
    "capability-value": {
            "hosts": [
                    "a.service123.ucdn.example.com",
                    "b.service123.ucdn.example.com"
            ],
            "transports": [{
                    "transport-type": "FCI.LoggingPullTransportAtomFeed",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "cdni_v1",
                            "filename-template": "%Y-%m-%d-access.%i.log",
                            "interval": 86400
                    },
                    "endpoint": {
                            "feed-href": "https://occ.example.com/logs/index.atom"
                    }
            }]
    }
}
```

Figure 17

7.5. FCI.LoggingPullTransportSFTP

Property: host

* Description: The hostname of the SFTP server, which MUST be specified as either hostname or hostname:port.

* Type: String

* Mandatory-to-Specify: Yes

Property: username

* Description: The remote username to use for connecting to the SFTP server.

* Type: String

* Mandatory-to-Specify: Yes

Property: password

* Description: The remote password to use for connecting to the SFTP server when using password-based authentication. This field is not used when SSH keys are provided by the uCDN via MI.LoggingPullTransportMetadata.

* Type: MI.SecretValue

* Mandatory-to-Specify: No

The following is an FCI.LoggingPullTransportSFTP example:

```
{
    "capability-type": "FCI.LoggingSource",
    "capability-value": {
            "hosts": [
                    "a.service123.ucdn.example.com",
                    "b.service123.ucdn.example.com"
            ],
            "transports": [{
                    "type": "FCI.LoggingPullTransportSFTP",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "filename-template": "%Y-%m-%d-access.%i.log",
                            "interval": 86400
                    },
                    "endpoint": {
                            "host": "logs.dcdn.example.com",
                            "username": "logsucdn"
                    }
            }]
    }

}
```

Figure 18

7.6. FCI.LoggingPullTransportS3API

Property: host

* Description: The hostname to connect to the S3-compatible API.

* Type: String

* Mandatory-to-Specify: Yes

Property: access-key-id

* Description: The access key ID to use when connecting to the API.

* Type: String

* Mandatory-to-Specify: Yes

Property: access-key-secret

* Description: The access key secret to use when connecting to the API.

* Type: MI.SecretValue

* Mandatory-to-Specify: Yes

Property: bucket-name

* Description: The bucket name to use to pull log file objects.

* Type: String

* Mandatory-to-Specify: Yes

The following is an FCI.LoggingPullTransportS3API example:

```
{
    "capability-type": "FCI.LoggingSource",
    "capability-value": {
            "hosts": [
                    "a.service123.ucdn.example.com",
                    "b.service123.ucdn.example.com"
            ],
            "transports": [{
                    "type": "FCI.LoggingPullTransportS3API",
                    "record-type": "opencaching_standard_json_v1",
                    "container-metadata": {
                            "container-type": "json_v1",
                            "name-template": "%Y-%m-%d-access.log",
                            "interval": 86400
                    },
                    "endpoint": {
                            "host": "xxxxxxxxxxxxx",
                            "access-key-id": "xxxxxxxxx",
                            "access-key-secret": {
                                    "secret-store-id": "store-1",
                                    "secret-store-path": "/logging/dCDN1/dcdnlogs/s3"
                            },
                            "bucket-name": "ucdnlogs"
                    }
            }]
    }
}
```

Figure 19

8. Appendix

8.1. Protobuf Record Type Definitions

This subsection provides the protobuf definitions in the .proto file format for the protobuf record types defined in section 9.1.3

```
syntax = "proto3";
message OCLogRequestV1 {
  google.protobuf.Timestamp timestamp = 1;

  uint64 s_time_total_ms = 3;
  string c_groupid = 4;

  Method cs_method = 5;
  float cs_version = 6;
  string cs_uri = 7;
  string cs_uri_transformed = 8;
  uint32 sc_status = 9;
  uint64 sc_total_bytes = 10;
  string ccid = 11;
  string s_sid = 12;
  CacheStatus s_cached = 13;
  uint64 sc_header_size_bytes = 14;
  string c_ip = 15;
  string s_shortname = 16;
  string s_id = 17;
  uint64 s_time_first_ms = 18;
  uint64 s_sdur_ms = 19;
  uint64 s_time_upstream_ms = 20;
  uint64 s_upstream_bytes = 21;
  repeated HttpHeader cs_hdr = 22;
  repeated HttpHeader sc_hdr = 23;
}
```

Figure 20

9. Security Considerations

9.1. Field Privacy

Logging records may contain a user’s personally identifiable information (PII), which may be inappropriate for the dCDN to share, and/or for the uCDN to store. This is particularly important when the dCDN is supporting custom headers. This specification provides MI.LoggingTransforms defined in section 7.4. (Section 6.4) as a mechanism for truncating, replacing, encrypting, and/or hashing record field values.

The uCDN can request transformations be applied to record fields with the transforms property of MI.LoggingTransport object defined in section 7.3. (Section 6.3)

The dCDN advertises transformations applied to record fields with the applied-transforms property of FCI.LoggingPush defined in section 8.1 (Section 7.1)

9.2. Field Encryption

In using the MI.LoggingTransformEncrypt defined in section 7.4.9 (Section 6.4.9), the nonce SHOULD NOT be reused across OCN nodes using a counter permutation. When the uniqueness of nonces cannot be ensured, you can use the XChaCha20 algorithm with a nonce from a random source with sufficient entropy, or you can take the hash of a weaker random source, concatenate it with the message and hash it with a cryptographic hash function that is safe against length-extension attacks, such as BLAKE2 or the HMAC construction.

Rivest Cipher 4 (RC4) is listed to support annotating values originating from a packet cores' header enrichment network function for sharing subscriber or network identifiers with web applications. It SHOULD NOT be used as a security construct for concealing data in logs. An implementation would effectively be a no-op passing the value with bound credentials. It is RECOMMENDED to chain another MI.LoggingTransformEncrypt operation with a modern algorithm for actual confidentiality.

9.3. Field Length

It is important to consider using an MI.LoggingTransformTruncate in implementations that MAY have constraints on individual field record lengths. The dCDN can advertise these field length constraints in the applied-transforms property of FCI.LoggingPush and the transforms property of FCI.LoggingPullTransport*.

It is strongly RECOMMENDED to include an MI.LoggingTransformTruncate when including any cs-hdr-* OR sc-hdr-* fields in the logging format to protect against any denial of service (DoS) risks that arise from inadvertently malformed or actively malicious values in the headers.

9.4. Atom Endpoints

Authentication and authorization mechanisms for the Atom feed are outside the scope of this document. The endpoint should be secured in a manner similar to the OCC interfaces as defined in the Open Caching Architecture Specification [SVTAXXXX].

10. IANA Considerations

11. Acknowledgements

The authors would like to express their gratitude to the members of the Streaming Video Technology Alliance [SVTA] Open Caching Working Group for their guidance / contribution / reviews ...)

Particulary the following people contributed to the content of this draft:

* Glenn Goldstein (Lumen)

* Christoph Neumann (Broadpeak)

* Rajeev RK (PicoNETS)

* Dave Seddon (Siden)

* Alfonso Silniz (Telefnica)

* Matt Stock (Viasat)

12. 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/info/rfc2119>.

[RFC4180] Shafranovich, Y., "Common Format and MIME Type for Comma-Separated Values (CSV) Files", RFC 4180, DOI 10.17487/RFC4180, October 2005, <https://www.rfc-editor.org/info/rfc4180>.

[RFC7937] Le Faucheur, F., Ed., Bertrand, G., Ed., Oprescu, I., Ed., and R. Peterkofsky, "Content Distribution Network Interconnection (CDNI) Logging Interface", RFC 7937, DOI 10.17487/RFC7937, August 2016, <https://www.rfc-editor.org/info/rfc7937>.

[RFC8006] Niven-Jenkins, B., Murray, R., Caulfield, M., and K. Ma, "Content Delivery Network Interconnection (CDNI) Metadata", RFC 8006, DOI 10.17487/RFC8006, December 2016, <https://www.rfc-editor.org/info/rfc8006>.

[RFC8008] Seedorf, J., Peterson, J., Previdi, S., van Brandenburg, R., and K. Ma, "Content Delivery Network Interconnection (CDNI) Request Routing: Footprint and Capabilities Semantics", RFC 8008, DOI 10.17487/RFC8008, December 2016, <https://www.rfc-editor.org/info/rfc8008>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

13. Informative References

[SVTA] SVTA, "Streaming Video Technology Alliance Home Page", <https://www.svta.org>.

Authors' Addresses

Ben Rosenblum
Vecima
United States of America
Email: ben@rosenblum.dev

Omar Ramadan
Blockcast
Email: omar@blockcast.net

Kenton Seward
Lumen
Email: kenton.seward@gmail.com