

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: December 02, 2013

G. Bertrand, Ed.
I. Oprescu, Ed.
France Telecom - Orange
F. Le Faucheur, Ed.
Cisco Systems
R. Peterkofsky
Skytide, Inc.
May 31, 2013

CDNI Logging Interface
draft-ietf-cdni-logging-03

Abstract

This memo specifies the Logging interface between a downstream CDN (dCDN) and an upstream CDN (uCDN) that are interconnected as per the CDN Interconnection (CDNI) framework. First, it describes a reference model for CDNI logging. Then, it specifies the CDNI Logging File format and the actual protocol for exchange of CDNI Logging Files.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 02, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Introduction	3
1.1.	Terminology	4
2.	CDNI Logging Reference Model	5
2.1.	CDNI Logging interactions	5
2.2.	Overall Logging Chain	8
2.2.1.	Logging Generation and During-Generation Aggregation	9
2.2.2.	Logging Collection	10
2.2.3.	Logging Filtering	10
2.2.4.	Logging Rectification and Post-Generation Aggregation	11
2.2.5.	Log-Consuming Applications	12
2.2.5.1.	Maintenance/Debugging	12
2.2.5.2.	Accounting	12
2.2.5.3.	Analytics and Reporting	13
2.2.5.4.	Security	13
2.2.5.5.	Legal Logging Duties	13
2.2.5.6.	Notions common to multiple Log Consuming Applications	13
3.	CDNI Logging File Format	15
3.1.	CDNI Logging File Directives	16
3.2.	Logging Records	20
3.2.1.	HTTP Request Logging Record	21
3.2.2.	CDNI Logging File Example	27
3.3.	Fields and Directives Formats	28
4.	CDNI Logging File Exchange Protocol	28
4.1.	CDNI Logging Feed	29
4.2.	CDNI Logging File Pull	29
5.	Open Issues	30
6.	IANA Considerations	31

7.	Security Considerations	31
7.1.	Authentication, Confidentiality, Integrity Protection . .	31
7.2.	Non Repudiation	32
7.3.	Privacy	32
8.	Acknowledgments	33
9.	References	33
9.1.	Normative References	33
9.2.	Informative References	34
Appendix A.	Requirements	34
A.1.	Compliance with cdni-requirements	34
A.2.	Additional Requirements	35
A.2.1.	Timeliness	35
A.2.2.	Reliability	35
A.2.3.	Security	35
A.2.4.	Scalability	35
A.2.5.	Consistency between CDNI Logging and CDN Logging . .	36
A.2.6.	Dispatching/Filtering	36
Appendix B.	Analysis of candidate protocols for Logging	
	Transport	36
B.1.	Syslog	36
B.2.	XMPP	36
B.3.	SNMP	37
	Authors' Addresses	37

[1.](#) Introduction

This memo specifies the Logging interface between a downstream CDN (dCDN) and an upstream CDN (uCDN). First, it describes a reference model for CDNI logging. Then, it specifies the CDNI Logging File format and the actual protocol for exchange of CDNI Logging Files.

The reader should be familiar with the following documents:

- o CDNI problem statement [[RFC6707](#)] and framework [[I-D.ietf-cdni-framework](#)] identify a Logging interface,
- o Section 8 of [[I-D.ietf-cdni-requirements](#)] specifies a set of requirements for Logging,
- o [[RFC6770](#)] outlines real world use-cases for interconnecting CDNs. These use cases require the exchange of Logging information between the dCDN and the uCDN.

As stated in [[RFC6707](#)], "the CDNI Logging interface enables details of logs or events to be exchanged between interconnected CDNs".

The present document describes:

- o The CDNI Logging reference model ([Section 2](#)),
- o The CDNI Logging File format ([Section 3](#)),
- o The CDNI Logging File Exchange protocol ([Section 4](#)).

1.1. Terminology

In this document, the first letter of each CDNI-specific term is capitalized. We adopt the terminology described in [[RFC6707](#)] and [[I-D.ietf-cdni-framework](#)], and extend it with the additional terms defined below.

For clarity, we use the word "Log" only for referring to internal CDN logs and we use the word "Logging" for any inter-CDN information exchange and processing operations related to CDNI Logging interface. Log and Logging formats may be different.

CDN Logging information: logging information generated and collected within a CDN

CDNI Logging information: logging information exchanged across CDNs using the CDNI Logging Interface

Logging information: logging information generated and collected within a CDN or obtained from another CDN using the CDNI Logging Interface

CDNI Logging Field: an atomic element of information that can be included in a CDNI Logging Record. The time an event/task started, the IP address of an End user to whom content was delivered, and the URI of the content delivered are examples of CDNI Logging Fields.

CDNI Logging Record: an information record providing information about a specific event. This comprises a collection of CDNI Logging Fields.

CDNI Logging File: a file containing CDNI Logging Records, as well as additional information facilitating the processing of the CDNI Logging Records.

CDN Reporting: the process of providing the relevant information that will be used to create a formatted content delivery report provided to the CSP in deferred time. Such information typically includes aggregated data that can cover a large period of time (e.g., from hours to several months). Uses of Reporting include the collection of charging data related to CDN services and the computation of Key Performance Indicators (KPIs).

CDN Monitoring: the process of providing content delivery information in real-time. Monitoring typically includes data in real time to provide visibility of the deliveries in progress, for service operation purposes. It presents a view of the global health of the services as well as information on usage and performance, for network services supervision and operation management. In particular, monitoring data can be used to generate alarms.

2. CDNI Logging Reference Model

2.1. CDNI Logging interactions

The CDNI logging reference model between a given uCDN and a given dCDN involves the following interactions:

- o customization by the uCDN of the CDNI logging information to be provided by the dCDN to the uCDN (e.g. control of which logging fields are to be communicated to the uCDN for a given task performed by the dCDN, control of which types of events are to be logged). The dCDN takes into account this CDNI logging customization information to determine what logging information to provide to the uCDN, but it may, or may not, take into account this CDNI logging customization information to influence what CDN logging information is to be generated and collected within the dCDN (e.g. even if the uCDN requests a restricted subset of the logging information, the dCDN may elect to generate a broader set of logging information). The mechanism to support the customisation by the uCDN of CDNI Logging information is outside the scope of this document and left for further study. We note that the CDNI Control interface or the CDNI Metadata interface appear as candidate interfaces on which to potentially build such a customisation mechanism in the future. Before such a mechanism is available, the uCDN and dCDN are expected to agree off-line on what CDNI logging information is to be provide by dCDN to UCDN and rely on management plane actions to configure the CDNI Logging functions to generate (respectively, expect) in dCDN (respectively, in uCDN).
- o generation and collection by the dCDN of logging information related to the completion of any task performed by the dCDN on behalf of the uCDN (e.g., delivery of the content to an end user) or related to events happening in the dCDN that are relevant to the uCDN (e.g., failures or unavailability in dCDN). This takes place within the dCDN and does not directly involve CDNI interfaces.
- o communication by the dCDN to the uCDN of the logging information collected by the dCDN relevant to the uCDN. This is supported by

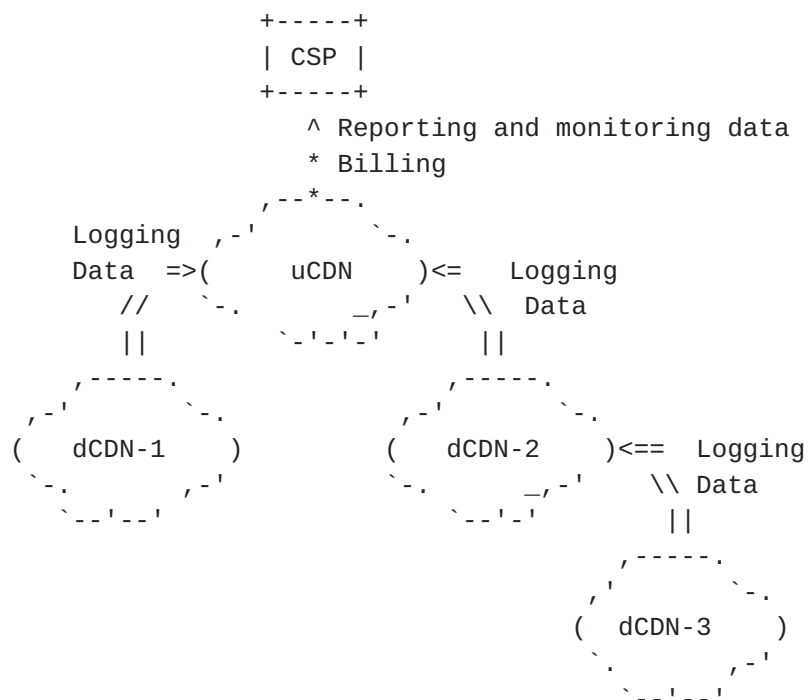
the CDNI Logging interface and in the scope of the present document. For example, the uCDN may use this logging information to charge the CSP, to perform analytics and monitoring for operational reasons, to provide analytics and monitoring views on its content delivery to the CSP or to perform trouble-shooting.

- o customization by the dCDN of the logging to be performed by the uCDN on behalf of the dCDN. The mechanism to support the customisation by the dCDN of CDNI Logging information is outside the scope of this document and left for further study.
- o generation and collection by the uCDN of logging information related to the completion of any task performed by the uCDN on behalf of the dCDN (e.g., serving of content by uCDN to dCDN for acquisition purposes by dCDN) or related to events happening in the uCDN that are relevant to the dCDN. This takes place within the uCDN and does not directly involve CDNI interfaces.
- o communication by the uCDN to the dCDN of the logging information collected by the uCDN relevant to the dCDN. For example, the dCDN might potentially benefit from this information for security auditing or content acquisition troubleshooting. This is outside the scope of this document and left for further study.

Figure 1 provides an example of CDNI Logging interactions (focusing only on the interactions that are in the scope of this document) in a particular scenario where 4 CDNs are involved in the delivery of content from a given CSP: the uCDN has a CDNI interconnection with dCDN-1 and dCDN-2. In turn, dCDN2 has a CDNI interconnection with dCDN3. In this example, uCDN, dCDN-1, dCDN-2 and dCDN-3 all participate in the delivery of content for the CSP. In this example, the CDNI Logging interface enables the uCDN to obtain logging information from all the dCDNs involved in the delivery. In the example, uCDN uses the Logging data:

- o to analyze the performance of the delivery operated by the dCDNs and to adjust its operations (e.g., request routing) as appropriate,
- o to provide reporting (non real-time) and monitoring (real-time) information to CSP.

For instance, uCDN merges Logging data, extracts relevant KPIs, and presents a formatted report to the CSP, in addition to a bill for the content delivered by uCDN itself or by its dCDNs on his behalf. uCDN may also provide Logging data as raw log files to the CSP, so that the CSP can use its own logging analysis tools.



==> CDNI Logging Interface

***> outside the scope of CDNI

Figure 1: Interactions in CDNI Logging Reference Model

A dCDN (e.g., dCDN-2) integrates the relevant logging information obtained from its dCDNs (e.g., dCDN-3) in the logging information that it provides to the uCDN, so that the uCDN ultimately obtains all logging information relevant to a CSP for which it acts as the authoritative CDN.

Note that the format of Logging information that a CDN provides over the CDNI interface might be different from the one that the CDN uses internally. In this case, the CDN needs to reformat the Logging information before it provides this information to the other CDN over the CDNI Logging interface. Similarly, a CDN might reformat the Logging data that it receives over the CDNI Logging interface before injecting it into its log-consuming applications or before providing some of this logging information to the CSP. Such reformatting operations introduce latency in the logging distribution chain and introduce a processing burden. Therefore, there are benefits in specifying CDNI Logging format that are suitable for use inside CDNs and also are close to the CDN Log formats commonly used in CDNs today.

2.2. Overall Logging Chain

This section discusses the overall logging chain within and across CDNs to clarify how CDN Logging information is expected to fit in this overall chain. Figure 2 illustrates the overall logging chain within the dCDN, across CDNs using the CDNI Logging interface and within the uCDN. Note that the logging chain illustrated in the Figure is obviously only indicative and varies depending on the specific environments. For example, there may be more or less instantiations of each entity (i.e., there may be 4 Log consuming applications in a given CDN). As another example, there may be one instance of Rectification process per Log Consuming Application instead of a shared one.

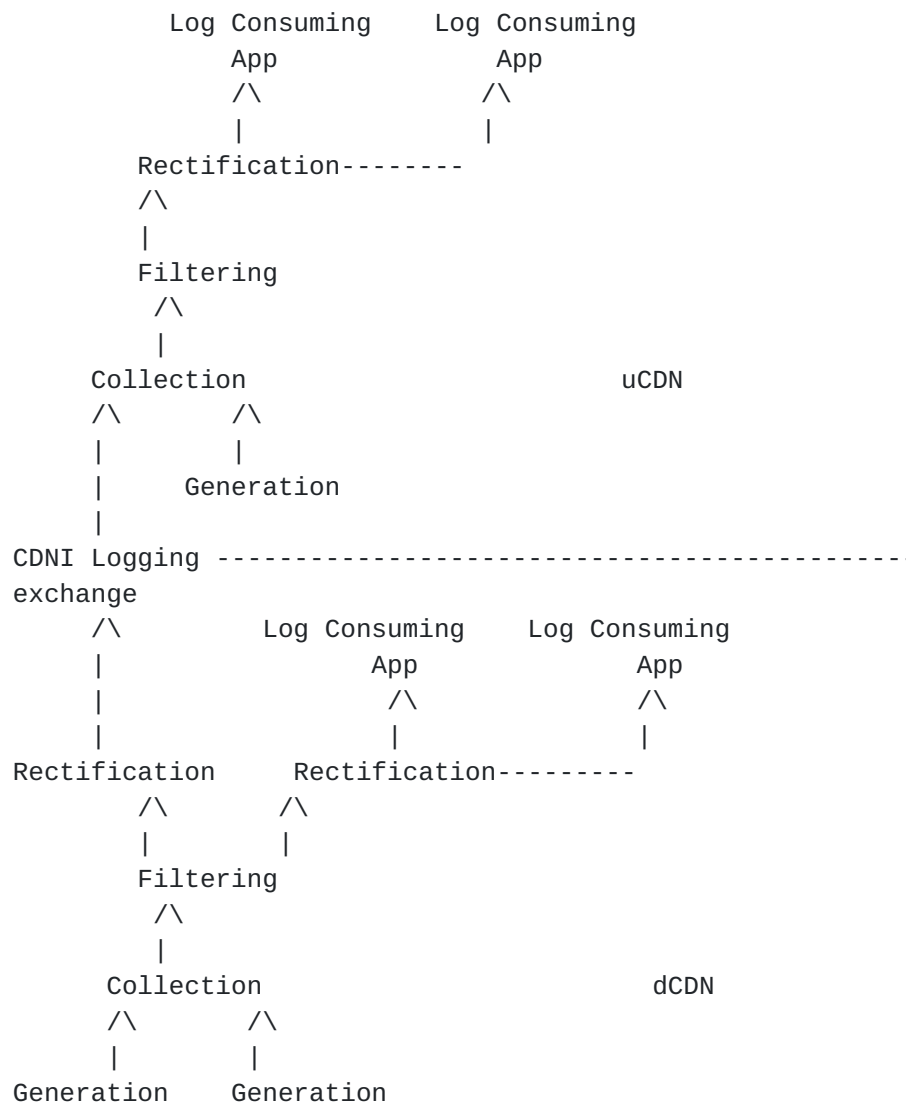


Figure 2: CDNI Logging in the overall Logging Chain

The following subsections describe each of the processes potentially involved in the logging chain of Figure 2.

2.2.1. Logging Generation and During-Generation Aggregation

CDNs typically generate logging information for all significant task completions, events, and failures. Logs are typically generated by many devices in the CDN including the surrogates, the request routing system, and the control system.

The amount of Logging information generated can be huge. Therefore, during contract negotiations, interconnected CDNs often agree on a

Logging retention duration, and optionally, on a maximum size of the Logging data that the dCDN must keep. If this size is exceeded, the dCDN must alert the uCDN but may not keep more Logs for the considered time period. In addition, CDNs may aggregate logs and transmit only summaries for some categories of operations instead of the full Logging data. Note that such aggregation leads to an information loss, which may be problematic for some usages of Logging (e.g., debugging).

[I-D.brandenburg-cdni-has] discusses logging for HTTP Adaptive Streaming (HAS). In accordance with the recommendations articulated there, it is expected that a surrogate will generate separate logging information for delivery of each chunk of HAS content. This ensures that separate logging information can then be provided to interconnected CDNs over the CDNI Logging interface. Still in line with the recommendations of [I-D.brandenburg-cdni-has], the logging information for per-chunk delivery may include some information (a Content Collection IDentifier and a Session IDentifier) intended to facilitate subsequent post-generation aggregation of per-chunk logs into per-session logs. Note that a CDN may also elect to generate aggregate per-session logs when performing HAS delivery, but this needs to be in addition to, and not instead of, the per-chunk delivery logs. We note that this may be revisited in future versions of this document.

Note that in the case of non real-time logging, the trigger of the transmission or generation of the logging file appears to be a synchronous process from a protocol standpoint. The implementation algorithm can choose to enforce a maximum size for the logging file beyond which the transmission is automatically triggered (and thus allow for an asynchronous transmission process).

2.2.2. Logging Collection

This is the process that continuously collects logs generated by the log-generating entities within a CDN.

In a CDNI environment, in addition to collecting logging information from log-generating entities within the local CDN, the Collection process also collects logging information provided by another CDN, or other CDNs, through the CDNI Logging interface. This is illustrated in Figure 2 where we see that the Collection process of the uCDN collects logging information from log-generating entities within the uCDN as well as logging information coming through CDNI Logging exchange with the dCDN through the CDNI Logging interface.

2.2.3. Logging Filtering

A CDN may require to only present different subset of the whole logging information collected to various log-consuming applications. This is achieved by the Filtering process.

In particular, the Filtering process can also filter the right subset of information that needs to be provided to a given interconnected CDN. For example, the filtering process in the dCDN can be used to ensure that only the logging information related to tasks performed on behalf of a given uCDN are made available to that uCDN (thereby filtering all the logging information related to deliveries by the dCDN of content for its own CSPs). Similarly, the Filtering process may filter or partially mask some fields, for example, to protect End Users' privacy when communicating CDNI Logging information to another CDN. Filtering of logging information prior to communication of this information to other CDNs via the CDNI Logging interface requires that the downstream CDN can recognize the set of log records that relate to each interconnected CDN.

The CDN will also filter some internal scope information such as information related to its internal alarms (security, failures, load, etc).

In some use cases described in [\[RFC6770\]](#), the interconnected CDNs do not want to disclose details on their internal topology. The filtering process can then also filter confidential data on the dCDNs' topology (number of servers, location, etc.). In particular, information about the requests served by every Surrogate may be confidential. Therefore, the Logging information must be protected so that data such as Surrogates' hostnames is not disclosed to the uCDN. In the "Inter-Affiliates Interconnection" use case, this information may be disclosed to the uCDN because both the dCDN and the uCDN are operated by entities of the same group.

2.2.4. Logging Rectification and Post-Generation Aggregation

If Logging is generated periodically, it is important that the sessions that start in one Logging period and end in another are correctly reported. If they are reported in the starting period, then the Logging of this period will be available only after the end of the session, which delays the Logging generation.

A Logging rectification/update mechanism could be useful to reach a good trade-off between the Logging generation delay and the Logging accuracy. Depending on the selected Logging protocol(s), such mechanism may be invaluable for real time Logging, which must be provided rapidly and cannot wait for the end of operations in progress.

In the presence of HAS, some log-consuming applications can benefit from aggregate per-session logs. For example, for analytics, per-session logs allow display of session-related trends which are much more meaningful for some types of analysis than chunk-related trends. In the case where the log-generating entities have generated during-generation aggregate logs, those can be used by the applications. In the case where aggregate logs have not been generated, the Rectification process can be extended with a Post-Generation Aggregation process that generates per-session logs from the per-chunk logs, possibly leveraging the information included in the per-chunk logs for that purpose (Content Collection IDentifier and a Session IDentifier). However, in accordance with [\[I-D.brandenburg-cdni-has\]](#), this document does not define exchange of such aggregate logs on the CDNI Logging interface. We note that this may be revisited in future versions of this document.

[2.2.5. Log-Consuming Applications](#)

[2.2.5.1. Maintenance/Debugging](#)

Logging is useful to permit the detection (and limit the risk) of content delivery failures. In particular, Logging facilitates the resolution of configuration issues.

To detect faults, Logging must enable the reporting of any CDN operation success and failure, such as request redirection, content acquisition, etc. The uCDN can summarize such information into KPIs. For instance, Logging format should allow the computation of the number of times during a given epoch that content delivery related to a specific service succeeds/fails.

Logging enables the CDN providers to identify and troubleshoot performance degradations. In particular, Logging enables the communication of traffic data (e.g., the amount of traffic that has been forwarded by a dCDN on behalf of an uCDN over a given period of time), which is particularly useful for CDN and network planning operations.

[2.2.5.2. Accounting](#)

Logging is essential for accounting, to permit inter-CDN billing and CSP billing by uCDNs. For instance, Logging information provided by dCDNs enables the uCDN to compute the total amount of traffic delivered by every dCDN for a particular Content Provider, as well as, the associated bandwidth usage (e.g., peak, 95th percentile), and the maximum number of simultaneous sessions over a given period of time.

2.2.5.3. Analytics and Reporting

The goal of analytics is to gather any relevant information to track audience, analyze user behavior, and monitor the performance and quality of content delivery. For instance, Logging enables the 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 any 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.

2.2.5.4. Security

The goal of security is to prevent and monitor unauthorized access, misuse, modification, and denial of access of a service. A set of information is logged 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.

2.2.5.5. Legal Logging Duties

Depending on the country considered, the CDNs may have to retain specific Logging information during a legal retention period, to comply with judicial requisitions.

2.2.5.6. Notions common to multiple Log Consuming Applications

2.2.5.6.1. Logging Information Views

Within a given log-consuming application, different views may be provided to different users depending on privacy, business, and scalability constraints.

For example, an analytics tool run by the uCDN can provide one view to an uCDN operator that exploits all the logging information available to the uCDN, while the tool may provide a different view to each CSP exploiting only the logging information related to the content of the given CSP.

As another example, maintenance and debugging tools may provide different views to different CDN operators, based on their operational role.

2.2.5.6.2. Key Performance Indicators (KPIs)

This section presents, for explanatory purposes, a non-exhaustive list of Key Performance Indicators (KPIs) that can be extracted/produced from logs.

Multiple log-consuming applications, such as analytics, monitoring, and maintenance applications, often compute and track such KPIs.

In a CDNI environment, depending on the situation, these KPIs may be computed by the uCDN or by the dCDN. But it is usually the uCDN that computes KPIs, because uCDN and dCDN may have different definitions of the KPIs and the computation of some KPIs requires a vision of all the deliveries performed by the uCDN and all its dCDNs.

Here is a list of important examples of KPIs:

- o Number of delivery requests received from End-Users in a given region for each piece of content, during a given period of time (e.g., hour/day/week/month)
- o Percentage of delivery successes/failures among the aforementioned requests
- o Number of failures listed by failure type (e.g., HTTP error code) for requests received from End Users in a given region and for each piece of content, during a given period of time (e.g., hour/day/week/month)
- o Number and cause of premature delivery termination for End Users in a given region and for each piece of content, during a given period of time (e.g., hour/day/week/month)
- o Maximum and mean number of simultaneous sessions established by End Users in a given region, for a given Content Provider, and during a given period of time (e.g., hour/day/week/month)
- o Volume of traffic delivered for sessions established by End Users in a given region, for a given Content Provider, and during a given period of time (e.g., hour/day/week/month)
- o Maximum, mean, and minimum delivery throughput for sessions established by End Users in a given region, for a given Content Provider, and during a given period of time (e.g., hour/day/week/month)

- o Cache-hit and byte-hit ratios for requests received from End Users in a given region for each piece of content, during a given period of time (e.g., hour/day/week/month)
- o Top 10 of the most popularly requested content (during a given day /week/month),
- o Terminal type (mobile, PC, STB, if this information can be acquired from the browser type header, for example).

Additional KPIs can be computed from other sources of information than the Logging, for instance, data collected by a content portal or by specific client-side application programming interfaces. Such KPIs are out of scope for the present memo.

The KPIs used depend strongly on the considered log-consuming application -- the CDN operator may be interested in different metrics than the CSP is. In particular, CDN operators are often interested in delivery and acquisition performance KPIs, information related to Surrogates' performance, caching information to evaluate the cache-hit ratio, information about the delivered file size to compute the volume of content delivered during peak hour, etc.

Some of the KPIs, for instance those providing an instantaneous vision of the active sessions for a given CSP's content, are useful essentially if they are provided in real-time. By contrast, some other KPIs, such as the one averaged on a long period of time, can be provided in non-real time.

3. CDNI Logging File Format

As defined in [Section 1.1](#) a CDNI logging field is as an atomic logging information element and a CDNI Logging Record is a collection of CDNI Logging Fields containing all logging information corresponding to a single logging event. This document defines a third level of structure, the CDNI Logging File, that is a collection of CDNI Logging Records. This structure is illustrated in Figure 3. The CDNI Logging File structure and encoding is specified in the present section.

```

+-----+
|CDNI Logging File|
|               |
| +-----+ |
| |CDNI Logging Record| | | | | | | |
| | +-----+ +-----+ +-----+ | |
| | |CDNI Logging | |CDNI Logging | |CDNI Logging | | |
| | | Field      | | Field      | | Field      | | |

```



```

| | +-----+ +-----+ +-----+ | |
| +-----+ |
| |
| +-----+ |
| |CDNI Logging Record | | | | | | | |
| | +-----+ +-----+ +-----+ | |
| | |CDNI Logging | |CDNI Logging | |CDNI Logging | | |
| | | Field | | Field | | Field | | |
| | +-----+ +-----+ +-----+ | |
| +-----+ |
| |
| +-----+ |
| |CDNI Logging Record | | | | | | | |
| | +-----+ +-----+ +-----+ | |
| | |CDNI Logging | |CDNI Logging | |CDNI Logging | | |
| | | Field | | Field | | Field | | |
| | +-----+ +-----+ +-----+ | |
| +-----+ |
+-----+

```

Figure 3: Structure of Logging Files

The CDNI Logging File format is inspired from the W3C Extended Log File Format [[ELF](#)]. However, it is fully specified by the present document. Where the present document differs from the W3C Extended Log File Format, an implementation of CDNI Logging MUST comply with the present document.

A CDNI Logging File MUST contain a sequence of lines containing US-ASCII characters [[CHAR_SET](#)] terminated by either the sequence LF or CRLF. A CDNI Logging implementation consuming CDNI Logging Files MUST accept lines terminated by either LF or CRLF.

Each line of a CDNI Logging File MUST contain either a directive or a CDNI Logging Record.

Directives record information about the CDNI Logging process itself. Lines containing directives MUST begin with the "#" character. Directives are specified in [Section 3.1](#).

Logging Records provide actual details of the logged event. Logging Records are specified in [Section 3.2](#).

[3.1](#). CDNI Logging File Directives

An implementation of the CDNI Logging interface MUST support the following directives (formats specified in the form <...> are specified in [Section 3.3](#)):

o Version:

- * format: <digit>.<digit>
- * semantic: indicates the version of the CDNI Logging File format. The value MUST be "1.0" for the version specified in the present document.
- * occurrence: there MUST be one and only one instance of this directive. It MUST be the first line of the CDNI Logging file.

o UUID:

- * format: <string>
- * semantic: this is Universally Unique Identifier for the CDNI Logging File as specified in [\[RFC4122\]](#).
- * occurrence: there MUST be one and only one instance of this directive.

o Claimed-Origin:

- * format: <host>
- * semantic: this contains the claimed identification of the entity transmitting the CDNI Logging File (e.g. the host in a dCDN supporting the CDNI Logging interface) or the entity responsible for transmitting the CDNI Logging File (e.g. the dCDN).
- * occurrence: there MUST be zero or one instance of this directive. This directive MAY be included by the dCDN. It MUST NOT be included or modified by the uCDN.

o Verified-Origin:

- * format: <host>

- * semantic: this contains the identification, as established by the entity receiving the CDNI Logging file, of the entity transmitting the CDNI Logging File (e.g. the host in a dCDN supporting the CDNI Logging interface) or the entity responsible for transmitting the CDNI Logging File (e.g. the dCDN).
 - * occurrence: there MUST be zero or one instance of this directive. This directive MAY be added by the uCDN. It MUST NOT be included by the dCDN. The mechanisms used by the uCDN to establish and validate the entity responsible for the CDNI Logging File is outside the scope of the present document. We observe that, in particular, this may be achieved through authentication mechanisms that are part of the CDNI Logging File pull mechanism ([Section 4.2](#)).
- o Record-Type:
- * format: <string>
 - * semantic: indicates the type of the CDNI Logging Records that follow this directive, until another Record-Type directive (or the end of the CDNI Logging File). "cdni_http_request_v1" MUST be indicated in the Record-Type directive for CDNI Logging records corresponding to HTTP request (e.g. a HTTP delivery request) as specified in [Section 3.2.1](#).
 - * occurrence: there MUST be at least one instance of this directive. The first instance of this directive MUST precede a Fields directive and precede any CDNI Logging Record.
- o Fields:
- * format: <field-name>[<field-name>], where the allowed list of <field-name> are specified for each Record-Type in [Section 3.2](#).
 - * semantic: this lists the names of all the fields for which a value is to appear in the CDNI Logging Records that are after this directive. The names of the fields, as well as their possible occurrences, are specified for each type of CDNI Logging Records in [Section 3.2](#). The field names listed in this directive MUST be separated by the "horizontal tabulation (TAB)" character.
 - * occurrence: there MUST be at least one instance of this directive per Record-Type directive. The first instance of this directive for a given Record-Type MUST precede any CDNI Logging Record for this Record-Type.

- o Integrity-Hash:

- * format: <string>
- * semantic: This directive permits the detection of a corrupted CDNI Logging File. This can be useful, for instance, if a problem occurs on the filesystem of the dCDN Logging system and leads to a truncation of a logging file. The Integrity-Hash value is computed, and included in this directive by the entity that transmits the CDNI Logging File. It is computed by applying the MD5 ([\[RFC1321\]](#)) cryptographic hash function on the CDNI Logging File, including all the directives and logging records, up to the Integrity-Hash directive itself, excluding the Integrity-Hash directive itself and, when present, also excluding the Non-Repudiation-Hash directive. The Integrity-Hash value is represented as a US-ASCII encoded hexadecimal number, 32 digits long (representing a 128 bit hash value). The entity receiving the CDNI Logging File also computes in a similar way the MD5 hash on the received CDNI Logging File and compares this hash to the value of the Integrity-Hash directive. If the two values are equal, then the received CDNI Logging File MUST be considered non-corrupted. If the two values are different, the received CDNI Logging File MUST be considered corrupted. The behavior of the entity that received a corrupted CDNI Logging File is outside the scope of this specification; we note that the entity MAY attempt to pull again the same CDNI Logging file from the transmitting entity. If the entity receiving the CDNI Logging File adds a Verified-Origin directive, it MUST recompute and update the Integrity-Hash directive so it also protects the added Verified-Origin directive.
- * occurrence: there MUST be zero or one instance of this directive. There SHOULD be one instance of this directive. One situation where that directive could be omitted is where integrity protection is already provided via another mechanism (for example if an integrity hash is associated to the CDNI Logging file out of band through the CDNI Logging Logging Feed [Section 4.1](#) leveraging ATOM extensions such as those proposed in [\[I-D.snell-atompub-link-extensions\]](#). When present, this field MUST be the last line of the CDNI Logging File when the Non-Repudiation-Hash is absent, and MUST be the one before last line of the CDNI Logging File when the Non-Repudiation-Hash is present.

- o Non-Repudiation-Signature:

- * format: <string>

- * semantic: This field contains a signature that supports the non-repudiation of the CDNI Logging File by the entity that transmitted the CDNI Logging File. [Editor's Note: Detailed description To Be Added. David Mandelberg has the lead for drafting text. The text needs to indicate that the Claimed-Origin directive MUST be covered and the Verified-Origin directive MUST NOT be covered by the signature. We may want to refer to [RFC4949](#) for terminology around Non-Redudiation]
- * occurrence: there MAY be one and only one instance of this directive. When present, this directive MUST be the last line of the CDNI Logging File.

3.2. Logging Records

A CDNI Logging Record consists of a sequence of CDNI Logging Fields relating to that single CDNI Logging Record.

CDNI Logging Fields MUST be separated by the "horizontal tabulation (TAB)" character.

Some CDNI Logging field names use a prefix scheme similar to the one used in W3C Extended Log File Format [[ELF](#)] to facilitate readability. The semantics of the prefix in the present document is:

- o c: refers to the User Agent that issues the request (corresponds to the "client" of W3C Extended Log Format)
- o d: refers to the dCDN (relative to a given CDN acting as a uCDN)
- o s: refers to the dCDN Surrogate that serves the request (corresponds to the "server" of W3C Extended Log Format)
- o u: refers to the uCDN (relative to a given CDN acting as a dCDN)
- o cs: refers to communication from the dCDN Surrogate towards the User-Agent
- o sc: refers to communication from the User-Agent towards the dCDN Surrogate

An implementation of the CDNI Logging interface as per the present specification MUST support the CDNI HTTP Delivery Records as specified in [Section 3.2.1](#). [Editor's Note: other types of delivery records will be listed here if we specify other types for this version eg Request Routing].

The formats listed in this section in the form <...> are specified in [Section 3.3](#)).

3.2.1. HTTP Request Logging Record

The HTTP Request Logging Record contains the following CDNI Logging Fields, listed by their field name:

- o date:
 - * format: <date>
 - * semantic: the date at which the processing of request completed on the Surrogate.
 - * occurrence: there MUST be one and only one instance of this field.
- o time:
 - * format: <time>
 - * semantic: the time at which the processing of request completed on the Surrogate.
 - * occurrence: there MUST be one and only one instance of this field.
- o time-taken:
 - * format: <fixed>
 - * semantic: duration, in seconds, between the start of the processing of the request and the completion of the delivery by the Surrogate.
 - * occurrence: there MUST be one and only one instance of this field.
- o c-ip:
 - * format: <address>
 - * semantic: the source IPv4 or IPv6 address (i.e. the "client" address) in the request received by the Surrogate.
 - * occurrence: there MUST be one and only one instance of this field.

- o c-ip-anonimizing:
 - * format: <integer>
 - * semantic: the number of bits of the address in the c-ip field that are zeroed-out in order to anonymize the logging record. The mechanism by which the two ends of the CDNI Logging interface agree on whether anonymization is to be supported and the number of bits that need to be zeroed-out for this purpose are outside the scope of the present document.
 - * occurrence: there MUST be zero or one instance of this field.
- o c-port:
 - * format: <integer>
 - * semantic: the source TCP port (i.e. the "client" port) in the request received by the Surrogate.
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-ip:
 - * format: <address>
 - * semantic: the IPv4 or IPv6 address of the Surrogate that served the request (i.e. the "server" address).
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-hostname:
 - * format: <host>
 - * semantic: the hostname of the Surrogate that served the request (i.e. the "server" hostname).
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-port:
 - * format: <integer>

- * semantic: the destination TCP port (i.e. the "server" port) in the request received by the Surrogate.
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o cs-method:
- * format: <string>
 - * semantic: this is the HTTP method of the HTTP request received by the Surrogate.
 - * occurrence: There MUST be one and only one instance of this field.
- o cs-uri:
- * format: <uri>
 - * semantic: this is the complete http_URL (as specified in [\[RFC2616\]](#)) of the request received by the Surrogate.
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o u-uri:
- * format: <uri>
 - * semantic: this is a complete http_URL (as specified in [\[RFC2616\]](#)) derived from the complete URI of the request received by the Surrogate but transformed by the entity generating or transmitting the CDNI Logging Record, in a way that is agreed upon between the two ends of the CDNI Logging interface, so the transformed URI is meaningful to the uCDN. For example, the two ends of the CDNI Logging interface could agree that the u-uri is constructed from the cs-uri by removing the part of the hostname that exposes which individual Surrogate actually performed the delivery. The details of modification performed to generate the u-uri, as well as the mechanism to agree on these modifications between the two sides of the CDNI Logging interface are outside the scope of the present document.
 - * occurrence: there MUST be one and only one instance of this field.

- o protocol:
 - * format: <string>
 - * semantic: this is value of the HTTP-Version field as specified in [\[RFC2616\]](#) of the Request-Line of the request received by the Surrogate (e.g. "HTTP/1.1").
 - * occurrence: there MUST be one and only one instance of this field.
- o sc-status:
 - * format: <digit><digit><digit>
 - * semantic: this is the HTTP Status-Code in the HTTP response from the Surrogate.
 - * occurrence: There MUST be one and only one instance of this field.
- o sc-total-bytes:
 - * format: <integer>
 - * semantic: this is the total number of bytes of the HTTP response sent by the Surrogate in response to the request. This includes the bytes of the Status-Line (including HTTP headers) and of the message-body.
 - * occurrence: There MUST be one and only one instance of this field.
- o sc-entity-bytes:
 - * format: <integer>
 - * semantic: this is the number of bytes of the message-body in the HTTP response sent by the Surrogate in response to the request. This does not include the bytes of the Status-Line (and therefore does not include the bytes of the HTTP headers).
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o cs(<HTTP-header>):
 - * format: <string>

- * semantic: the value of the HTTP header identified in the field name as it appears in the request processed by the Surrogate.
- * occurrence: there MUST be zero, one or any number of instance of this field.
- o sc(<HTTP-header>):
 - * format: <string>
 - * semantic: the value of the HTTP header identified in the field name as it appears in the response issued by the Surrogate to serve the request.
 - * occurrence: there MUST be zero, one or any number of instance of this field.
- o s-ccid:
 - * format: [Editor's Note: to be based on cdni-metadata or relevant companion I-D]
 - * semantic: this contains the value of the Content Collection Identifier specified in [[I-D.ietf-cdni-metadata](#)] and associated to the content served by the Surrogate through the CDNI Metadata interface.
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-sid:
 - * format: [Editor's Note: add reference to the I-D defining the format of Session ID>?]
 - * semantic: this contains the value of the Session Identifier specified in ??? and associated to the served request by the Surrogate.
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-cached: [Editor's Note: W3C uses "cached" . is "s-cached" better?]
 - * format: <string>

- * semantic: this characterises whether the Surrogate could serve the request using content already stored on its local cache. The allowed values are "0" (for miss) and "1" for hit). "1" MUST be used when the Surrogate could serve the request using exclusively content already stored on its local cache. "0" MUST be used otherwise (including cases where the Surrogate served the request using some, but not all, content already stored on its local cache). Note that a "0" only means a cache miss in the Surrogate and does not provide any information on whether the content was already stored, or not, in another device of the dCDN i.e. whether this was a "dCDN hit" or "dCDN miss".
 - * occurrence: there MUST be zero or exactly one instance of this field.
- o s-uri-signing:
- * format: <string>
 - * semantic: this characterises the uri signing validation performed by the Surrogate on the request. The allowed values are:
 - *
 - + "0" : no uri signature validation performed
 - + "1" : uri signature validation performed and validated
 - + "2" : uri signature validation performed and rejected
 - * occurrence: there MUST be zero or exactly one instance of this field.

The "Fields" directive corresponding to a HTTP Request Logging Record MUST list all the fields whose occurrence is specified above as "There MUST be one and only one instance of this field". These fields MUST be present in every HTTP Request Logging Record.

The "Fields" directive corresponding to a HTTP Request Logging Record MAY list all the fields whose occurrence is specified above as "there MUST be zero or exactly one instance of this field" or "there MUST be zero, one or any number of instance of this field". The set of such fields actually listed in the "Fields" directive is selected by the implementation generating the CDNI Logging File based on agreements between the interconnected CDNs established through mechanisms outside the scope of this specification (e.g. contractual

agreements) . When such a field is not listed in the "Fields" directive, it MUST NOT be included in the Logging Record. When such a field is listed in the "Fields" directive, it MUST be included in the Logging Record; in that case, if the value for the field is not available, this MUST be conveyed via a dash character ("-").

The fields listed in the "Fields" directive can be listed in the order in which they are listed in [Section 3.2.1](#) or in any other order.

[Editor's Note: discuss private fields]

3.2.2. CDNI Logging File Example

```
#Version:1.0<CRLF>

#UUID: urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6???<CRLF>

#Claimed-Origin: cdni-logging-entity.dcdn.example.com<CRLF>

#Record-Type: cdni_http_request_v1<CRLF>

#Fields:<TAB>date<TAB>time<TAB>time-taken<TAB>c-ip<TAB>cs-
method<TAB>u-uri<TAB>protocol<TAB>sc-status<TAB>sc-total-bytes<TAB>cs-
(User-Agent)<TAB>cs(Referer)<TAB>s-cached<CRLF>

2013-05-17<TAB>00:38:06.825<TAB>88.958<TAB>10.5.7.1<TAB>GET<TAB>http
://cdni-ucdn.dcdn.example.com/video/movie100.mp4<TAB>HTTP/
1.1<TAB>200<TAB>672989<TAB>Mozilla/5.0 (Windows; U; Windows NT 6.0;
en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.127
Safari /533.4<TAB>host1.example.com <TAB>1<CRLF>

2013-05-17<TAB>00:39:09.145<TAB>169.790<TAB>10.5.10.5<TAB>GET<TAB>htt
p://cdni-ucdn.dcdn.example.com/video/movie118.mp4<TAB>HTTP/
1.1<TAB>200<TAB>1579920<TAB>Mozilla/5.0 (Windows; U; Windows NT 6.0;
en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.127
Safari /533.4<TAB>host1.example.com<TAB>1<CRLF>

2013-05-17<TAB>00:42:53.437<TAB>2.879<TAB>10.5.10.5<TAB>GET<TAB>http
://cdni-ucdn.dcdn.example.com/video/picture11.mp4<TAB>HTTP/
1.0<TAB>200<TAB>17724<TAB>Mozilla/5.0 (Windows; U; Windows NT 6.0;
en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.127
Safari /533.4<TAB>host5.example.com<TAB>0<CRLF>

#Integrity-Hash: 9e107d9d372bb6826bd81d3542a419d6 [Editor's Note:
include the correct MD5-hash value for the actual example]<CRLF>
```


3.3. Fields and Directives Formats

[Editor's Note: still needs work to minimise the number of types defined across this section and specific types defined inside the field definitions themselves]

- o <digit> = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
- o <integer> = 1*<digit>
- o <address> = <integer> ["." *<integer>] [":" <integer>]
- o <host> = as specified in [[RFC3986](#)].
- o <date> = 4<digit> "-" 2<digit> "-" 2<digit>
 - * Dates are recorded in the format YYYY-MM-DD where YYYY, MM and DD stand for the numeric year, month and day respectively. All dates are specified in Universal Time Coordinated (UTC).
- o <time> = 2<digit> ":" 2<digit> ":" 2<digit> ["." *<digit>]
 - * Times are recorded in the form HH:MM:SS or HH:MM:SS.S where HH is the hour in 24 hour format, MM is minutes and SS is seconds. All times are specified in Universal Time Coordinated (UTC).
- o <uri> = <string> containing a http_URL as specified in [[RFC2616](#)].
- o <fixed> = Fixed Format Float = 1*<digit> ["." *<digit>]
- o <HTTP-header> = <string> containing a HTTP header field name (e.g. "User-Agent", "Referer") as specified in [[RFC2616](#)].

4. CDNI Logging File Exchange Protocol

This document specifies a protocol for the exchange of CDNI Logging Files as specified in [Section 3](#).

This protocol comprises:

- o a CDNI Logging feed, allowing the dCDN to notify the uCDN about the CDNI Logging files that can be retrieved by that uCDN from the dCDN, as well as all the information necessary for retrieving each of these CDNI Logging File. The CDNI Logging feed is specified in [Section 4.1](#).

- o a CDNI Logging File pull mechanism, allowing the uCDN to obtain from the dCDN a given CDNI Logging File at the uCDN convenience. The CDNI Logging File pull mechanisms is specified in [Section 4.2](#).

An implementation of the CDNI Logging interface as per the present document generating CDNI Logging file (i.e. on the dCDN side) MUST support the server side of the CDNI Logging feed and the server side of the CDNI Logging pull mechanism.

An implementation of the CDNI Logging interface as per the present document consuming CDNI Logging file (i.e. on the uCDN side) MUST support the client side of the CDNI Logging feed and the client side of the CDNI Logging pull mechanism.

[Editor's note: verify that the client side and server side are well defined in the respective sections]

We note that implementations of the CDNI Logging interface MAY also support other mechanisms to exchange CDNI Logging Files, for example in view of exchanging logging information with minimum time-lag (e.g. sub-minute or sub-second) between when the event occurred in the dCDN and when the corresponding Logging Record is made available to the uCDN (e.g. for log-consuming applications requiring extremely fresh logging information such as near-real-time content delivery monitoring). Such mechanism might be defined in future version of the present document.

[4.1.](#) CDNI Logging Feed

[Editor's Note: text to be added. Feed is based on ATOM and contains a UUID + URI for each CDNI Logging File in "window" - if appropriate the text should refer to the side generating the CDNI Logging Feed "as server-side", and the side consuming the Feed as the client-side].

[4.2.](#) CDNI Logging File Pull

A client-side implementation of the CDNI Logging interface MAY pull at its convenience any CDNI Logging File that is advertised by the server-side in the CDNI Logging Feed. To do so, the client-side:

- o MUST use HTTP v1.1
- o SHOULD use TLS (i.e. use what is loosely referred to as "HTTPS")
- o MUST use the URI associated to the CDNI Logging File in the CDNI Logging Feed

- o SHOULD indicate the compression schemes it supports

Note that a client-side implementation of the CDNI Logging interface MAY pull a CDNI Logging File that it has already pulled, as long as the file is still advertised by the server-side in the CDNI Logging Feed.

The server-side implementation MUST respond to any valid pull request by a client-side implementation for a CDNI Logging File advertised by the server-side in the CDNI Logging Feed. The server-side implementation:

- o MUST handle the client-side request as per HTTP v1.1
- o MUST include the CDNI Logging File identified by the request URI inside the body of the HTTP response
- o MUST support the gzip and deflate compression schemes
- o MAY support other compression schemes
- o when the client-side request indicates client-supported compression schemes, SHOULD use a compression scheme that it supports and is supported by the client-side

[Editor's Note: discuss Non-Repudiation : it is a nice to have and how it could be supported, via a different digest than the one for integrity]

5. Open Issues

- o What separator should be used between a directive name and a directive value e.g. "#Version:1.0" or "#Version: 1.0" or "#Version:<TAB>1.0"?
- o Compression: <Ben>When we say the server MUST support gzip & deflate we probably need to think through whether we mean content-encoding, transfer-encoding or both. The semantics get a little confusing so we probably just need to think them through to ensure we allow a server to store compressed logs as transmit them compressed.
- o Handling of Event logs and notifications: There are two aspects of that question:
 - * non-real-time exchange of event logs from dCDN to uCDN for audit purposes. This could be added to current spec presumably in the form of additional Record-Types and without requiring a

significant change to the current CDNI Logging file exchange approach. It is proposed that this be handled as a [MED] requirement. e.g. try first specify what events and what information needs to be exchanged; and depending on progress, decide to include in initial logging spec or not

- * real-time exchange of event notification from dCDN to uCDN for immediate operational action (eg on notification by dCDN that dCDN request routing is down, uCDN stops redirecting to this dCDN). This would presumably require definition/extension of another CDNI interface or significant change/extension to the current CDNI logging spec. It is proposed that this be kept out of the scope of the current cdni-logging spec .

Another question is what set of events should be logged/notified. The first type of events relates to "service-level" events i.e. high level events that affect the service that the dCDN is providing to the uCDN (e.g. dCDN request routing is down, dCDN is overloaded). There is general agreement that it is desirable to be able to log/notify such service-level events. The second type of events is "atomic-level" events i.e. low level events that may be useful to identify or track a component issue or a delivery issue. logging/notifying about such events may be useful in some situations (eg uCDN and dCDN have a particular relationship allowing them to share detailed operational information) and may not be useful in some situations (because the dCDN does not want to expose details of its CDN operation). Ideal approach is to define both types of events and have the first type as MUST and the second type as MAY. Fall back approach would be to only define the first type initially.

- o Add precise definition of what must be supported by transmitting implementation and what must be implemented by receiving application (regardless of what may actually be used in a given deployment). For example, it may be reasonable to mandate that a receiving implementation be able to receive all the directives specified in the doc and all fields.

6. IANA Considerations

TBD

7. Security Considerations

7.1. Authentication, Confidentiality, Integrity Protection

The use of TLS for transport of the CDNI Logging feed mechanism ([Section 4.1](#)) and CDNI Logging File pull mechanism ([Section 4.2](#)) allows:

- o the dCDN and uCDN to authenticate each other (to ensure they are transmitting/receiving CDNI Logging File from an authenticated CDN)
- o the CDNI Logging information to be transmitted with confidentiality
- o the integrity of the CDNI Logging information to be protected during the exchange.

The Integrity-Hash directive inside the CDNI Logging File provides additional integrity protection, this time targeting potential corruption of the CDNI logging information during the CDNI Logging File generation. This mechanism does not allow restoration of the corrupted CDNI Logging information, but it allows detection of such corruption and therefore triggering of appropriate correcting actions (e.g. discard of corrupted information, attempt to re-obtain the CDNI Logging information).

[7.2.](#) Non Repudiation

The Non-Repudiation-Signature directive in the CDNI Logging File allows support of non-repudiation of the CDNI Logging File by the dCDN. The optional Non-Repudiation-Hash can be used on the CDNI Logging interface where needed.

[7.3.](#) Privacy

CDNs have the opportunity to collect detailed information about the downloads performed by End-Users. The provision of this information to another CDN introduces potential End-Users privacy protection concerns. We observe that when CDNI interconnection is realised as per [[I-D.ietf-cdni-framework](#)], the uCDN handles the initial End-User requests (before it is redirected to the dCDN) so, regardless of which information is, or is not, communicated to the uCDN through the CDNI Logging interface, the uCDN has visibility on significant information such as the IP address of the End-User request and the URL of the request. Nonetheless, if the dCDN and uCDN agree that anonymization is required to avoid making some detailed information available to the uCDN (such as how much bytes of the content has been watched by an enduser and/or at what time) or is required to meet some legal obligations, then the uCDN and dCDN can agree to exchange anonymized End-User IP addresses in CDNI Logging files and the c-ip-anonymization field can be used to convey the number of bits that

have been anonymized so that the meaningful information can still be easily extracted from the anonymized addresses (e.g. for geolocation aware analytics).

8. Acknowledgments

This document borrows from the W3C Extended Log Format [[ELF](#)].

The authors would like to thank Sebastien Cubaud, Pawel Grochocki, Christian Jacquenet, Yannick Le Louedec, Anne Marrec and Emile Stephan for their contributions on early versions of this document.

The authors would like also to thank Rob Murray, Fabio Costa, Sara Oueslati, Yvan Massot, Renaud Edel, and Joel Favier for their input and comments.

Finally, they thank the contributors of the EU FP7 OCEAN project for valuable inputs.

9. References

9.1. Normative References

- [I-D.ietf-cdni-metadata]
Niven-Jenkins, B., Murray, R., Watson, G., Caulfield, M., Leung, K., and K. Ma, "CDN Interconnect Metadata", [draft-ietf-cdni-metadata-01](#) (work in progress), February 2013.
- [I-D.snell-atompub-link-extensions]
Snell, J., "Atom Link Extensions", [draft-snell-atompub-link-extensions-09](#) (work in progress), June 2012.
- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", [RFC 1321](#), April 1992.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), January 2005.

- [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", [RFC 4122](#), July 2005.
- [RFC5424] Gerhards, R., "The Syslog Protocol", [RFC 5424](#), March 2009.

9.2. Informative References

- [CHAR_SET]
 , "IANA Character Sets registry", , <<http://www.iana.org/assignments/character-sets/character-sets.xml>>.
- [ELF] Phillip M. Hallam-Baker, and Brian Behlendorf, "Extended Log File Format, W3C (work in progress), WD-logfile-960323", , <<http://www.w3.org/TR/WD-logfile.html>>.
- [I-D.brandenburg-cdni-has]
 Brandenburg, R., Deventer, O., Faucheur, F., and K. Leung, "Models for adaptive-streaming-aware CDN Interconnection", [draft-brandenburg-cdni-has-05](#) (work in progress), April 2013.
- [I-D.ietf-cdni-framework]
 Peterson, L. and B. Davie, "Framework for CDN Interconnection", [draft-ietf-cdni-framework-03](#) (work in progress), February 2013.
- [I-D.ietf-cdni-requirements]
 Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", [draft-ietf-cdni-requirements-06](#) (work in progress), April 2013.
- [RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", [RFC 6707](#), September 2012.
- [RFC6770] Bertrand, G., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", [RFC 6770](#), November 2012.

Appendix A. Requirements

A.1. Compliance with cdni-requirements

This section checks that all the identified requirements in the Section 7 of [[I-D.ietf-cdni-requirements](#)] are fulfilled by this document.

[Editor's node: to be written later]

A.2. Additional Requirements

This section identifies additional requirements that must also be met.

[Editor's node: How do we incorporate this info into the I-D: in appendix? in main body? does it remain after publication or is temporary?]

A.2.1. Timeliness

Some applications consuming CDNI Logging information, such as accounting or trend analytics, only require logging information to be available with a timeliness of the order of a day or the hour. This document focuses on addressing this requirement.

Some applications consuming CDNI Logging information, such as real-time analytics, require logging information to be available in real-time (i.e. of the order of a second after the corresponding event). This document leaves this requirement out of scope.

A.2.2. Reliability

CDNI logging information must be transmitted reliably. The transport protocol should contain an anti-replay mechanism.

A.2.3. Security

CDNI logging information exchange must allow authentication, integrity protection, and confidentiality protection. Also, a non-repudiation mechanism is mandatory, the transport protocol should support it.

A.2.4. Scalability

CDNI logging information exchange must support large scale information exchange, particularly so in the presence of HTTP Adaptive Streaming.

For example, if we consider a client pulling HTTP Progressive Download content with an average duration of 10 minutes, this represents 1/600 CDNI delivery Logging Records per second. If we assume the dCDN is simultaneously serving 100,000 such clients on behalf of the uCDN, the dCDN will be generating 167 Logging Records per second to be communicated to the uCDN over the CDNI Logging interface. Or equivalently, if we assume an average delivery rate of 2Mb/s, the dCDN generates 0.83 CDNI Logging Records per second for every Gb/s of streaming on behalf of the uCDN.

For example, if we consider a client pulling HAS content and receiving a video chunk every 2 seconds, a separate audio chunk every 2 seconds and a refreshed manifest every 10 seconds, this represents 1.1 delivery Logging Record per second. If we assume the dCDN is simultaneously serving 100,000 such clients on behalf of the uCDN, the dCDN will be generating 110,000 Logging Records per second to be communicated to the uCDN over the CDNI Logging interface. Or equivalently, if we assume an average delivery rate of 2Mb/s, the dCDN generates 550 CDNI Logging Records per second for every Gb/s of streaming on behalf of the uCDN.

A.2.5. Consistency between CDNI Logging and CDN Logging

There are benefits in using a CDNI logging format as close as possible to intra-CDN logging format commonly used in CDNs today in order to minimize systematic translation at CDN/CDNI boundary.

A.2.6. Dispatching/Filtering

When a CDN is acting as a dCDN for multiple uCDNs, the dCDN needs to dispatch each CDNI Logging Record to the uCDN that redirected the corresponding request. The CDNI Logging format need to allow, and possibly facilitate, such a dispatching.

Appendix B. Analysis of candidate protocols for Logging Transport

This section will be expanded later with an analysis of alternative candidate protocols for transport of CDNI Logging in non-real-time as well as real-time.

B.1. Syslog

[Ed. node: to be written later]

B.2. XMPP

[Ed. node: to be written later]

B.3. SNMP

Authors' Addresses

Gilles Bertrand (editor)
France Telecom - Orange
38-40 rue du General Leclerc
Issy les Moulineaux 92130
FR

Phone: +33 1 45 29 89 46
Email: gilles.bertrand@orange.com

Iuniana Oprescu (editor)
France Telecom - Orange
38-40 rue du General Leclerc
Issy les Moulineaux 92130
FR

Phone: +33 6 89 06 92 72
Email: iuniana.oprescu@orange.com

Francois Le Faucheur (editor)
Cisco Systems
E.Space Park - Batiment D
6254 Allee des Ormes - BP 1200
Mougins cedex 06254
FR

Phone: +33 4 97 23 26 19
Email: flefauch@cisco.com

Roy Peterkofsky
Skytide, Inc.
One Kaiser Plaza, Suite 785
Oakland CA 94612
USA

Phone: +01 510 250 4284
Email: roy@skytide.com

