Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: August 26, 2013

G. Bertrand, Ed.
I. Oprescu, Ed.
E. Stephan
France Telecom - Orange
R. Peterkofsky
Skytide, Inc.
F. Le Faucheur, Ed.
Cisco Systems
P. Grochocki
Orange Polska

February 22, 2013

CDNI Logging Interface
draft-ietf-cdni-logging-01

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 actual protocol for CDNI logging information exchange covering the information elements as well as the transport of those elements.

Status of this Memo

Copyright Notice

Table of Contents

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 actual protocol for CDNI logging information exchange covering the information elements as well as the transport of those elements.

The reader should be familiar with the work of the CDNI WG:

o CDNI problem statement [RFC6707] and framework [I-D.ietf-cdni-framework] identify a Logging interface,

o Section 7 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 information structure and Transport (Section 4),

o The CDNI Logging Fields (Section 5),

o The CDNI Logging Records (Section 6),

o The CDNI Logging File format (Section 7),

o The CDNI Logging File Transport Protocol (Section 8),

In the Appendices, the document provides:

o A list of identified requirements (Appendix B.1), which should be considered for inclusion in [I-D.ietf-cdni-requirements],

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.

Separator Character: a specific character used to enable the parsing of Logging Records. This character separates the Logging Fields that compose a Logging Record.

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.

End-User experience management: study of Logging data using statistical analysis to discover, understand, and predict user behavior patterns.

Class-of-requests: A Class-of-requests identifies a set of content Requests, related to a specific CSP, received from clients in a given footprint and sharing common properties. These properties include:

- o Any header, URL parameter, query parameter of an HTTP (or RTMP) content request

- o Any header, or sub-domain of the FQDN of a DNS lookup request

Examples:

- o Class-of-Requests = all the requests that include the HTTP header "User-Agent: Mozilla/5.0" related to CSP "http://*.cdn.example.com" from AS3215

- o Class-of-Requests = all the DNS requests from anywhere and related to CSP "cdn*.example.com"

Delivery Service: A Delivery Service is defined by a set of Class-of-Requests and a list of parameters that apply to all these Class-of-Requests (logging format, delivery quality/capabilities requirements...)

Service Agreement: A service agreement is defined by a uCDN identifier, a dCDN identifier, a set of Delivery Services and a list of parameters that apply to the Service Agreement.

Once a Service Agreement is agreed between the administrative entities managing the CDNs to be interconnected, the upstream CDN and the downstream CDN of the CDNI interconnection must be configured according to this agreed Service Agreement. For instance, a given uCDN (uCDN1) may request a given dCDN (dCDN1) to configure one Delivery Service for handling requests for HTTP Adaptive streaming videos delegated by uCDN1 and related to a specific CSP (CSP1) and another one for handling requests for static pictures delegated by uCDN1 and related to CSP1. These Delivery services would belong to the Service Agreement between uCDN1 and dCDN1 for CSP1. In this simple example, uCDN1 may request dCDN1 to include Delivery Service information in its CDNI Logging, to help uCDN1 to provide relevant reports to CSP1.

1.2. Abbreviations

- API: Application Programming Interface
- CCID: Content Collection Identifier
- CDN: Content Delivery Network
- CDNP: Content Delivery Network Provider
- CoDR: Content Delivery Record
- CSP: Content Service Provider
- DASH: Dynamic Adaptive Streaming over HTTP
- dCDN: downstream CDN
- FTP: File Transfer Protocol
- HAS: HTTP Adaptive Streaming
- KPI: Key Performance Indicator
- PVR: Personal Video Recorder
- SID: Session Identifier
- SFTP: SSH File Transfer Protocol
- SNMP: Simple Network Management Protocol
- uCDN: upstream CDN

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:

- 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 ore the CDNI Metadata interfaces appear as candidate interfaces on which to potentially build such a customisation mechanism. 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).

- 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.

- 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.

- 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.

- 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.

- 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 form 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.

```
                   +-----+
                   | CSP |
                   +-----+
                      ^ Reporting and monitoring data
                      * Billing
                   ,--*--.
       Logging  ,-'       `-.
       Data  =>(     uCDN    )<=   Logging
          //   `-.       _,-'   \\  Data
          ||        `-'-'-'      ||
       ,-----.                 ,-----.
    ,-'       `-.           ,-'       `-.
   (   dCDN-1    )         (   dCDN-2    )<==  Logging
    `-.       ,-'          `-.      _,-'    \\ Data
      `--'--'                  `--'-'        ||
                                          ,-----.
                                        ,'       `-.
                                       (  dCDN-3    )
                                        `.       ,-'
                                          `--'--'

   ===> 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.

```
             Log Consuming    Log Consuming
                 App              App
                 /\              /\
                 |               |
           Rectification--------
           /\
           |
           Filtering
            /\
            |
        Collection                        uCDN
        /\       /\
        |        |
        |     Generation
        |
   CDNI Logging ---------------------------------------------
   exchange
        /\         Log Consuming    Log Consuming
        |                 App              App
        |                  /\              /\
        |                  |               |
   Rectification     Rectification---------
           /\       /\
           |        |
           Filtering
            /\
            |
         Collection                         dCDN
         /\       /\
         |        |
   Generation    Generation
```

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-chunck delivery may include some information (a Content Collection IDentifier and a Session IDentifier as discussed in Section 5) 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 beyound which the transmission is automatically triggered (and thus allow for an asynchrounous 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 enables the uCDN to check the total amount of traffic delivered by every dCDN and for every Delivery Service, 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)

- Maximum and mean number of simultaneous sessions established by End Users in a given region, for a given Delivery Service, and during a given period of time (e.g., hour/day/week/month)

- Volume of traffic delivered for sessions established by End Users in a given region, for a given Delivery Service, and during a given period of time (e.g., hour/day/week/month)

- Maximum, mean, and minimum delivery throughput for sessions established by End Users in a given region, for a given Delivery Service, and during a given period of time (e.g., hour/day/week/month)

- 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)

- Top 10 of the most popularly requested content (during a given day/week/month),

- 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 APIs. 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 Transport Requirements

3.1. 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.

3.2. Reliability

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

3.3. 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.

3.4. 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 chunck 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.

3.5. 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 tody in order to minimize systematic translation at CDN/CDNI boundary.

3.6. 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.

4. CDNI Logging Information Structure and Transport

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 non-real-time transport of CDNI Logging information over the CDNI interface. For such non-real-time transport, this documents defines a third level of structure, the CDNI Logging File, that is a collection of CDNI Logging Records. This structure is described in Figure 3. This document then specifies how to transport such CDNI Logging Files across interconnected CDNs. We observe that this approach can be tuned in a real deployment to achieve near-real time exchange of CDNI Logging information, e.g., by increasing the frequency of logging file creation and distribution throughout the Logging chain, but it is not expected that this approach can support real time transport (e.g., sub-second) of CDNI logging information.

```
   +------------------------------------------------------+
   |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

It is expected that future version of this document will also specify real time transport of CDNI Logging information over the CDNI interface. We note that this might involve direct transport of CDNI Logging Records without prior grouping into a file structure to avoid the latency associated with creating and transporting such a file structure throughout the logging chain.

The semantics and encoding of the CDNI Logging fields are specified in Section 5. The semantics and encoding of CDNI Records are specified in Section 6. The CDNI Logging File format is specified in Section 7. The protocol for transport of CDNI Logging File is specified in Section 8.

5. CDNI Logging Fields

Existing CDNs Logging functions collect and consolidate logs performed by their Surrogates. Surrogates usually store the logs using a format derived from Web servers' and caching proxies' log standards such as W3C, NCSA [ELF] [CLF], or Squid format [squid]. In practice, these formats are adapted to cope with CDN specifics. Appendix A presents examples of commonly used log formats.

5.1. Semantics of CDNI Logging Fields

This section specifies the semantics of the CDNI Logging Fields. The specific subset of CDNI Logging fields that can be found in each type of Logging Record is specified in Section 6.

The semantics of the CDNI Logging Fields are specified in Table 1.

Name	Description
Start-time	A start date and time associated with a logged event; for instance, the time at which a Surrogate received a content delivery request or the time at which an origin server received a content acquisition request.
End-time	An end date and time associated with a logged event. For instance, the time at which a Surrogate completed the handling of a content delivery request (e.g., end of delivery or error).
Duration	The duration of an operation in milliseconds. For instance, this field could be used to provide the time it took the Surrogate to send the requested file to the End-User or the time it took the Surrogate to acquire the file on a cache-miss event. In the case where Start-time, End-time, and Duration appear in a Logging Record, the Duration is to be interpreted as a total activity time related to the logged operation.
Client-IP	The IP address of the User Agent that issued the logged request or of a proxy, for instance "203.0.113.1".
Client-port	The source port of the logged request (e.g., 9542)
Destination-IP	The IP address of the host that received the logged request (e.g., 192.0.2.2).
Destination-hostname	The hostname of the host that received the logged request (e.g., Surrogate1.cdna.com).
Destination-port	The destination port of the logged request (e.g., 80).
Operation	The kind of operation that is logged; for instance Delivery or Purging.
URI_full	The full requested URL (e.g., "http://node1.peer-a.op-b.net/cdn.csp.com/movies/potter.avi?param=11&user=toto"). When HTTP request redirection is used, this URI includes the Surrogate FQDN. If the association of requests t oSurrogates is confidential, the dCDN can present only URI_part to uCDN.

URI_part	The requested URL path (e.g., /cdn.csp.com/movies/potter.avi?param=11&user=toto if the full request URL was "http://node1.peer-a.op-b.net/cdn.csp.com/movies/potter.avi?param=11&user=toto"). The URI without host-name typically includes the "CDN domain" (ex.cdn.csp.com) - cf. [I-D.ietf-cdni-framework]: it enables the identification of the CSP service agreed between the CSP and the CDNP operating the uCDN.
Protocol	The protocol and protocol version of the message that triggered the Logging entry (e.g., HTTP/1.1).
Request-method	The protocol method of the request message that triggered the Logging entry.
Status	The protocol status of the reply message related to the Logging entry
Bytes-Sent	The number of bytes at application-layer protocol-level (e.g., HTTP) of the reply message related to the Logging entry. It includes the size of the response headers.
Headers-Sent	The number of bytes corresponding to response headers at application-layer protocol-level (e.g., HTTP) of the reply message related to the Logging entry.
Bytes-received	The number of bytes (headers + body) of the message that triggered the Logging entry.
Referrer	The value of the Referrer header in an HTTP request.
User-Agent	The value of the User Agent header in an HTTP request.
Cookie	The value of the Cookie header in an HTTP request.
Byte-Range	[Ed. note: to be defined]
Cache-control	The value of the cache-control header in an HTTP answer. This header is particularly important for content acquisition logs.
Record-digest	A digest of the Logging Record; it enables detecting corrupted Logging Records.
CCID	A Content Collection IDentifier (CCID) eases the correlation of several Logging Records related to a Content Collection (e.g., a movie split in chunks).
SID	A Session Identifier (SID) eases the correlation (and aggregation) of several Logging Records related to a session. The SID is especially relevant for summarizing HAS Logging information [I-D.brandenburg-cdni-has].

uCDN-ID	An element authenticating the operator of the uCDN as the authority having delegated the request to the dCDN.
Delivering-CDN-ID	An identifier (e.g., an aggregation of an IP address and a FQDN) of the Delivering CDN. The Delivering-CDN-ID might be considered as confidential by the dCDN. In such case, the dCDN could either not provide this field to the uCDN or overwrite the Delivering-CDN-ID with its on identifier.
Cache-bytes	The number of body bytes served from caches. This quantity permits the computation of the byte hit ratio.
Action	The Action describes how a given request was treated locally: through which transport protocol, with or without content revalidation, with a cache hit or cache miss, with fresh or stale content, and (if relevant) with which error. Example with Squid format [squid]: "TCP_REFRESH_FAIL_HIT" means that an expired copy of an object requested through TCP was in the cache. Squid attempted to make an If-Modified-Since request, but it failed. The old (stale) object was delivered to the client.
MIME-Type	The MIME-Type of the requested content
dCDN identifier	An element authenticating the operator of the dCDN as the authority requesting the content to the uCDN
Caching_date	Date at which the delivered content was stored in cache
Validity_headers	A copy of all headers related to content validity: Pragma or Cache-Control (no-cache), ETag, Vary, last-modified...
Lookup_duration	Duration of the DNS resolution for resolving the FQDN of (uCDN's or CSP's) origin server.
Delay_to_first_bit	Duration of the operations from the sending of the content acquisition request to the reception of the first bit of the requested content.
Delay_to_last_bit	Duration of the operations from the sending of the content acquisition request to the reception of the last bit of the requested content.

Table 1: Semantics of CDNI Logging Fields

NB: we define three fields related to the timing of logged operations: Start-time, End-time, and Duration. Start-time is typically useful for human readers (e.g., while debugging), however,

some servers log the operation's End-time which corresponds to the time of log record generation. In absence of Logging summarization, only two of these three fields are required to obtain relevant timing information on the operation. However, when some kind of Logging aggregation/summarization is used, it can be advantageous to keep the three fields: for instance, in the case of HAS, keeping the three fields permits computing an average delivery bitrate from a single Logging Record aggregating information on the delivery of multiple consecutive video chunks.

Multiple header fields, in addition to the ones explicitly listed in the table could be reproduced in the Logging records.

Note that uCDN may want to filter Logging data by user (and not by IP address) to provide more relevant information to the CSP. In such case, a user may be identified as a combination of several pieces of information such as the client IP and User Agent or through the SID.

The URI_full provides information on the Surrogate that provided the content. This information can be relevant, for instance, for the Inter-Affiliates use case described in [RFC6770]. However, in some cases it may be considered as confidential and the dCDN may provide URI_part instead.

Other information that could be logged include operations that refer to the general state of the request, before it gets processed locally. Such information is related to the authorization of the requests, URL rewriting rules enforced, the X-FORWARDED-FOR non standard HTTP header...

[Editor's Note: CDNI Logging information may be used for debugging. Therefore, various CDN operations might be logged, depending on the agreement between the dCDN and the uCDN, such as operations related to Request Routing and Metadata. These may call for a few additional Fields to be defined].

5.2. Syntax of CDNI Logging Fields

This section is intended to contain the specification for the syntax and encoding of the CDNI Logging fields. For now, Table 2 illustrates the definition of some information elements. It provides examples using Apache log format strings [apache] when they exist.

[Ed. note: specify for all Logging Fields the type (e.g., varchar, int, float, ...) and the maximum size (e.g., varchar(200))]

Name	String	Example
Time	%t	[10/Oct/2000:13:55:36-0700]
Duration	%D	-
Client-IP	%a	203.0.113.45
Operation	-	-
URI_full	%U	-
Protocol	%H	HTTP/1.0
Request method	%m	GET
Status	%>s	200
Bytes Sent	%O	2326
Bytes received	%I	432
Header	\"%{Referrer}i\" \"%{User-agent}i\"	"http://www.example.com/start.html ""Mozilla/4.08 [en] (Win98; I ;Nav)"

Table 2: Examples using Apache format

6. CDNI Logging Records

[Ed. note: we need to specify the encoding of the file, the separation character, etc...]

This section defines the events for which a CDNI Logging record can be exchanged over the CDNI Logging interafce and for each type of Logging Record indicates the allowed set of CDNI Information Elements.

We classify the logged events depending on the CDN operation to which they relate: Content Delivery, Content Acquisition, Content Invalidation/Purging, etc.

6.1. Content Delivery

The content delivery event triggering the generation of a Logging Record include:

o Reception by a dCDN Surrogate of a content request

The Logging Record for Content Delivery contains the following set of

CDNI Logging Elements:

Name	Mandatory/Optional
Start-time	Mandatory
Duration	Mandatory
Client-IP	Mandatory
Client-port	Optional
Destination-IP	Mandatory if Destination-Hostname is absent
Destination-Hostname	Mandatory if Destination-IP is absent
Destination-port	Optional
Operation	Optional
URI_full	Mandatory if URI_part is absent
URI_part	Mandatory if URI_full is absent
Protocol	Mandatory if protocol is different to HTTP/1.1
Request-method	Mandatory
Status	Mandatory
Bytes-Sent	Mandatory
Headers-Sent	Optional
Bytes-received	Optional
Referrer	Optional
User-Agent	Optional
Cookie	Optional
Byte-Range	?
Cache-control	Optional
Record-digest	?
CCID	Optional. Only applicable to HTTP Adaptive Streaming delivery.
SID	Optional. Only applicable to HTTP Adaptive Streaming delivery.
Cache-bytes	Optional
Action	Mandatory (in particulat re cache Hit/Miss)
MIME-Type	Mandatory

Table 3: CDNI Logging Fields in Delivery Logging Record

In Table 3, "Mandatory" means that this field MUST be included in each Delivery Record and "Optional" means that it can be included based on the agreement between the dCDN and the uCDN as established via mechanism outside the scope of this document (e.g., by human agreement).

6.2. Content Invalidation and Purging

Given that the Purge interface is expected to contain a mechanism to report on completion of the Invalidation/purge request, there is no need to specify separate Log Records for these events.

6.3. Request Routing

[Editor's Note: Is there a requirement for the dCDN to provide logs for request routing events?]

6.4. Logging Extensibility

Future usages might introduce the need for additional Logging fields. In addition, some use-cases such as an Inter-Affiliate Interconnection [RFC6770], might take advantage of extended Logging exchanges. Therefore, it is important to permit CDNs to use additional Logging fields besides the standard ones, if they want. For instance, an "Account-name" identifying the contract enforced by the dCDN for a given request could be provided in extended fields.

The required Logging Records may depend on the considered services. For instance, static file delivery (e.g., pictures) typically does not include any delivery restrictions. By contrast, video delivery typically implies strong content delivery restrictions, as explained in [RFC6770], and Logging could include information about the enforcement of these restrictions. Therefore, to ease the support of varied services as well as of future services, the Logging interface should support optional Logging Records.

7. CDNI Logging File Format

Interconnected CDNs may support various Logging formats. However, they must support at least the default Logging File format described here.

7.1. Logging Files

[Ed. Note: How many files (one per type of Delivery Service (e.g., HTTP, WMP) and per type of Event (e.g., Errors, Delivery, Acquisition,...?)and what would be inside... These aspects needs to be detailed...]

7.2. File Format

The Logging file format should be independent from the selected transport protocol, to guarantee a flexible choice of transport

protocols. [Ed. note: for the real time Logging exchanges, this might be hard]

All Logging Records in a Logging File must share the same format (same set of Logging Fields, in the same order, with the same semantics, separated by the same Separator Character), to ease the parsing of the Logging data by the CDN that receives the Logging File. The CDN that provides the Logging data is responsible for guaranteeing the consistency of the Logging records' formats, typically via its log filtering and aggregation processes (see Section 2.2.3).

7.2.1. Headers

Logging files must include a header with the information described in Figure 4.

Field	Description	Examples
Format	Identification of CDNI Log format.	standard_cdni_errors_http_v1
Fields	A description of the record format (list of fields).	
Log-ID	Identifier for the CDNI Log file (facilitates detection of duplicate Logs and tracking in case of aggregation).	abcdef1234
Log-Timestamp	Time, in milliseconds, the CDNI Log was generated.	[20/Feb/2012:00:29.510+0200]
Log-Origin	Identifier of the authority (e.g., dCDN or uCDN) providing the Log- -ging	cdn1.cdni.example.com

Figure 4: Logging Headers

All time-related Logging Fields and data in the Logging File headers/footers must provide a time zone and be at least at millisecond (ms) accuracy. The accuracy must be consistent to permit the computation of KPIs involving operations realized on several CDNs.

[Ed. note: would it make sense to add a kind of "example Logging Record" in the Logging file and associated semantic (e.g., in a structure data format) ?]

7.2.2. Body (Logging Records) Format

[Ed. note: the W3C extended log format is a good base candidate to look at.]

Since records for real time information and non-real time information could use different formats, we do not yet solve the problem of real time logging exchanges in this version.

7.2.3. Footer Format

Logging files must include a footer with the information described in Figure 5.

Field	Description	Examples
Log Digest	Digest of the complete Log (facilitates detection of Log corruption)	

Figure 5: Logging footers

This digest field permits the detection of corrupted Logging files. 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. Additional mechanisms to avoid corrupted Logging files are expected to be provided by the Logging transport protocol, cf. Section 8.

8. CDNI Logging File Transport Protocol

As presented in [RFC6707], several protocols already exist that could potentially be used to exchange CDNI Logging between interconnected CDNs.

The offline exchange of non real-time Logging could rely on several protocols. In particular, the dCDN could publish the Logging on a server where the uCDN would retrieve them using a secure protocol.

For managed file transfer, the recommended protocol is SSH File Transfer Protocol (SFTP) [I-D.ietf-secsh-filexfer]. SFTP is widely deployed and it guarantees the respect of the criteria expressed by the CDNI Logging Transport Requirements: timeliness, reliability, security and scalability.

[Ed note: include options for lossless compression]

9. Open Issues

The main remaining tasks on this ID are the following:

- o Finalise the list of CDNI Logging Fields
- o Finalise the encoding of CDNI Logging Fields, Records and File.
- o Identify what can be done (if anything) to maximise reuse of Logging Fields and Logging Records encoding for future support of real-time CDNI Logging exchange

[Ed. Note: The format for Time is still to be agreed on. RFC 5322 (Section 3.3) format could be used or ISO 8601 formatted date and time in UTC (same format as proposed in [draft-caulfield-cdni-metadata-core-00]). Also see RFC5424 Section 6.2.3.]

[Ed. note: (comment from Kevin) how are errors handled ? If the client gets handed a bunch of 403s and 404s, but still gets the content eventually, without triggering an event, are those still logged? For Bytes-Sent, if there were aborted requests, do those get counted as well? Not all client behavior can be correlated with the simplified log]

10. IANA Considerations

TBD

11. Security Considerations

11.1. 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 End-Users privacy protection concerns.

11.2. Non Repudiation

Logging provides the raw material for charging. It permits the dCDN to bill the uCDN for the content deliveries that the dCDN makes on behalf of the uCDN. It also permits the uCDN to bill the CSP for the content Delivery Service. Therefore, non-repudiation of Logging data is essential.

12. Acknowledgments

The authors would like to thank Sebastien Cubaud, Anne Marrec, Yannick Le Louedec, and Christian Jacquenet for detailed feedback on early versions of this document and for their input on existing Log formats.

The authors would like also to thank 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.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, March 2009.

13.2. Informative References

[CLF] A. Luotonen, "The Common Log-file Format, W3C (work in progress)", 1995, <http://www.w3.org/pub/WWW/Daemon/User/Config/Logging.html>.

[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-04 (work in progress), January 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-04 (work in progress), December 2012.

[I-D.ietf-secsh-filexfer]
Galbraith, J. and O. Saarenmaa, "SSH File Transfer Protocol", draft-ietf-secsh-filexfer-13 (work in progress), July 2006.

[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.

[apache] "Apache 2.2 log files documentation", Feb. 2012, <http://httpd.apache.org/docs/current/logs.html>.

[squid] "Squid Log-Format documentation", Feb. 2012, <http://wiki.squid-cache.org/SquidFaq/SquidLogs>.

Appendix A. Examples Log Format

This section provides example of log formats implemented in existing CDNs, web servers, and caching proxies.

Web servers (e.g., Apache) maintain at least one log file for logging accesses to content (the Access Log). They can typically be configured to log errors in a separate log file (the Error Log). The log formats can be specified in the server's configuration files. However, webmasters often use standard log formats to ease the log processing with available log analysis tools.

A.1. W3C Common Log File (CLF) Format

The Common Log File (CLF) format defined by the World Wide Web Consortium (W3C) working group is compatible with many log analysis tools and is supported by the main web servers (e.g., Apache) Access Logs.

According to [CLF], the common log-file format is as follows: remotehost rfc931 authuser [date] "request" status bytes.

Example (from [apache]): 127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326

The fields are defined as follows [CLF]:

Element	Definition
remotehost	Remote hostname (or IP number if DNS hostname is not available, or if DNSLookup is Off.
rfc931	The remote logname of the user.
authuser	The username that the user employed to authenticate himself.
[date]	Date and time of the request.
"request"	An exact copy of the request line that came from the client.
status	The status code of the HTTP reply returned to the client.
bytes	The content-length of the document transferred.

Table 4: Information elements in CLF format

A.2. W3C Extended Log File (ELF) Format

The Extended Log File (ELF) format defined by W3C extends the CLF with new fields. This format is supported by Microsoft IIS 4.0 and 5.0.

The supported fields are listed below [ELF].

Element	Definition
date	Date at which transaction completed
time	Time at which transaction completed
time-taken	Time taken for transaction to complete in seconds
bytes	bytes transferred
cached	Records whether a cache hit occurred
ip	IP address and port
dns	DNS name
status	Status code
comment	Comment returned with status code
method	Method
uri	URI
uri-stem	Stem portion alone of URI (omitting query)
uri-query	Query portion alone of URI

Table 5: Information elements in ELF format

Some fields start with a prefix (e.g., "c-", "s-"), which explains which host (client/server/proxy) the field refers to.

o Prefix Description

o c- Client

o s- Server

o r- Remote

o cs- Client to Server.

o sc- Server to Client.

o sr- Server to Remote Server (used by proxies)

o rs- Remote Server to Server (used by proxies)

Example: date time s-ip cs-method cs-uri-stem cs-uri-query s-port cs-username c-ip cs(User-Agent) sc-status sc-substatus sc-win32-status time-taken

```
2011-11-23 15:22:01 x.x.x.x GET /file 80 y.y.y.y Mozilla/
5.0+(Windows;+U;+Windows+NT+6.1;+en-US;+rv:1.9.1.6)+Gecko/
20091201+Firefox/3.5.6+GTB6 200 0 0 2137
```

A.3. National Center for Supercomputing Applications (NCSA) Common Log Format

This format for Access Logs offers the following fields:

o host rfc931 date:time "request" statuscode bytes

o x.x.x.x userfoo [10/Jan/2010:21:15:05 +0500] "GET /index.html HTTP/1.0" 200 1043

A.4. NCSA Combined Log Format

The NCSA Combined log format is an extension of the NCSA Common log format with three (optional) additional fields: the referral field, the user_agent field, and the cookie field.

o host rfc931 username date:time request statuscode bytes referrer user_agent cookie

o Example: x.x.x.x - userfoo [21/Jan/2012:12:13:56 +0500] "GET /index.html HTTP/1.0" 200 1043 "http://www.example.com/" "Mozilla/4.05 [en] (WinNT; I)" "USERID=CustomerA;IMPID=01234"

A.5. NCSA Separate Log Format

The NCSA Separate log format refers to a log format in which the information gathered is separated into three separate files. This way, every entry in the Access Log (in the NCSA Common log format) is complemented with an entry in a Referral log and another one in an Agent log. These three records can be correlated easily thanks to the date:time value. The format of the Referral log is as follows:

o date:time referrer

o Example: [21/Jan/2012:12:13:56 +0500] "http://www.example.com/index.html"

The format of the Agent log is as follows:

o date:time agent

o [21/Jan/2012:12:13:56 +0500] "Microsoft Internet Explorer - 5.0"

A.6. Squid 2.0 Native Log Format for Access Logs

Squid [squid] is a popular piece of open-source software for transforming a Linux host into a caching proxy. Variations of Squid log format are supported by some CDNs.

Squid common access log format is as follow: time elapsed remotehost code/status bytes method URL rfc931 peerstatus/peerhost type.

Squid also supports a more detailed native access log format: Timestamp Elapsed Client Action/Code Size Method URI Ident Hierarchy/ From Content

According to Squid 2.0 documentation [squid], these fields are defined as follows:

Element	Definition
time	Unix timestamp as UTC seconds with a millisecond resolution.
duration	The elapsed time in milliseconds the transaction busied the cache.
client address	The client IP address.
bytes	The size is the amount of data delivered to the client, including headers.
request method	The request method to obtain an object.
URL	The requested URL.
rfc931	may contain the ident lookups for the requesting client (turned off by default)
hierarchy code	The hierarchy information provides information on how the request was handled (forwarding it to another cache, or requesting the content to the Origin Server).
type	The content type of the object as seen in the HTTP reply header.

Table 6: Information elements in Squid format

Squid also uses a "store log", which covers the objects currently kept on disk or removed ones, for debugging purposes typically.

Appendix B. Requirements

B.1. Additional Requirements

Section 7 of [I-D.ietf-cdni-requirements], already specifies a set of requirements for Logging (LOG-1 to LOG-16). Some security requirements also affect Logging (e.g., SEC-4).

This section is a placeholder for requirements identified in the work on logging, before they are proposed to the requirements draft authors.

Logging data is sensitive as it provides the raw material for producing bills etc. Therefore, the protocol delivering the Logging data must be reliable to avoid information loss. In addition, the protocol must scale to support the transport of large amounts of Logging data.

CDNs need to trust Logging information, thus, they want to know:

- o who issued the Logging (authentication), and

- o if the Logging has been modified by a third party (integrity).

Logging also contains confidential data, and therefore, it should be protected from eavesdropping.

All these needs translate into security requirements on both the Logging data format and on the Logging protocol.

Finally, this protocol must comply with the requirements identified in [I-D.ietf-cdni-requirements].

[Ed. note: cf. requirements draft: "SEC-4 [MED] The CDNI solution should be able to ensure that the Downstream CDN cannot spoof a transaction log attempting to appear as if it corresponds to a request redirected by a given Upstream CDN when that request has not been redirected by this Upstream CDN. This ensures non-repudiation by the Upstream CDN of transaction logs generated by the Downstream CDN for deliveries performed by the Downstream CDN on behalf of the Upstream CDN."]

B.2. Compliancy with Requirements draft

This section checks that all the identified requirements in the Requirements draft are fulfilled by this document.

[Ed. node: to be written later]

Appendix C. 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.

C.1. Syslog

[Ed. node: to be written later]

C.2. XMPP

[Ed. node: to be written later]

C.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

Stephan Emile
France Telecom - Orange
2 avenue Pierre Marzin
Lannion F-22307
France

Email: emile.stephan@orange.com

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

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

Francois Le Faucheur (editor)
Cisco Systems
Greenside, 400 Avenue de Roumanille
Sophia Antipolis 06410
FR

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

Pawel Grochocki
Orange Polska
ul. Obrzezna 7
Warsaw 02-691
Poland

Email: pawel.grochocki@orange.com