# Processing Stages & Metadata Expression Language

CDNI Working Group

IETF 123

7/21/25

# Processing Stages

**What is the draft about?**

- Document specifies a set of objects that can be conditionally applied at various points in a dCDNs processing of requests

**Usage**

- Processing Stages introduces the concept of stages during a CDN's processing pipeline where a given stage represents an opportunity to examine requests and responses and make alterations as needed
- Metadata, such as caching rules, can be applied conditionally based on an HTTP request/response headers. For e.g. dynamically adding or dropping an HTTP header
- Uses Metadata expression language

# Examples of usage

**Positive Cache Control**

- Allows the uCDN to specify both internal and external caching policies for the dCDN

**Negative Cache Control**

- Allows the uCDN to specify caching policies based on error response codes. For e.g., it may be desirable to cache error responses at the dCDN for a short period of time to prevent an overwhelmed origin source service or uCDN from being flooded with requests

**Cache Bypass Control**

- Allows content providers to bypass CDN caching when needed (typically for testing or performance benchmarking purposes)

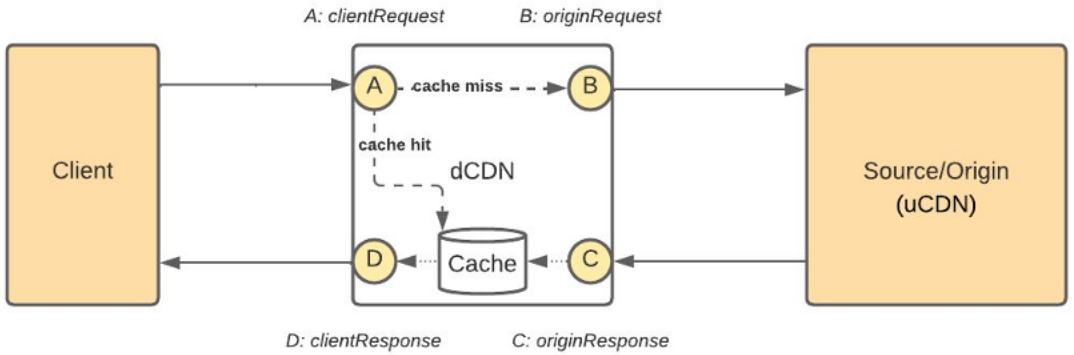
**Stale Content Policy**

- Allows control over how the dCDN should process requests for stale content

**Dynamically Constructed Cache Keys**

- CDN configurations to generate cache keys that are dynamically constructed via lightweight processing of various properties of the Hypertext Transfer Protocol (HTTP) request and/or response *[using Metadata Expression Language]*

# Processing stages metadata context




**clientRequest** - Rules run on the inbound client HTTP request prior to further processing.

**originRequest** - Rules run prior to making an HTTP request to the origin upon a cache miss.

**originResponse** - Rules run after an HTTP response is received from the origin and before being placed in the cache or forwarded to the client.

**clientResponse** - Rules run prior to sending the HTTP response to the client. If the response is from the cache, rules are applied to the response retrieved from the cache prior to sending to the client.

- It is typical in CDN configurations to define matching rules and metadata to be applied at specific stages in the HTTP request processing pipeline
- Append a host header prior to forwarding a request to an origin, or modify the response returned from an origin prior to storing in the cache

# Metadata Expression Language: Background

- Content delivery networks often need to conditionally modify responses before forwarding them to an end-user client. These modifications typically involve adding, deleting, or modifying HTTP response headers or synthesizing responses
- RFC8006 allows metadata to be applied conditionally based on matches to an HTTP request hostname and/or path which in most case might be sufficient
- However, there often is a need for configuration metadata (such as caching rules) to be applied conditionally based on matching values of HTTP request headers

# Metadata expression language

What is the spec for?

- Specifies syntax and provides usage examples for an expression language

Usage

- Metadata Expression Language (MEL) is to be used conditionally and dynamically on HTTP request/response headers

Applicability

- By defining a set of CDNI metadata objects for conditionally applying metadata and modifying responses as defined by the *Processing Stages Metadata Specification* which in turn requires an expression language to enable the specification of match expressions and synthetically generated values

# Metadata expression language: Example

Metadata Expression Language (MEL) provides a syntax with a rich set of variables, operators, and built-in functions to facilitate two key use cases:

- **Match Expressions:** Expressions that evaluate to a Boolean value are used to dynamically determine if metadata should be applied based on evaluation of aspects of an inbound HTTP request (matching to a header value, for example)
- **Value Expressions:** Enabling dynamic construction of a value to be used in scenarios such as constructing a cache key, setting an HTTP response header or status code, rewriting a request Uniform Resource Identifier (URI), or dynamically generating a response body

# Next steps

- Drafts processing stages and metadata expression language kicked off a need for an update to the current charter and based on discussions at IETF 121 and IETF 122, there appears to be an agreed upon text
- Awaiting chairs/AD to seek charter approval
- In the meantime, can we call for an adoption of the drafts so the working group can start to work the document?

# Thank you