Open Caching CDNI extensions proposals

Sanjay Mishra, Ori Finkelman

IETF-100, Singapore

November 2017

Open Caching and CDNI

- Open Caching is a specific case of CDNI where a commercial CDN is the uCDN and ISP caching layer serves as the dCDN
  - Open Caching is built on CDNI interfaces
  - Some functionalities and interfaces required for Open Caching does not currently exist in CDNI
- We propose to
  - Add Open Caching as a use case of CDNI
  - Add missing interfaces as an extensions to CDNI RFCs
  - Specify missing functionalities as new drafts under CDNI

REQUEST ROUTING

Request routing

- Open Caching uses iterative Request Redirect as described in RFC7336
  - Asynchronous Footprint advertisement
  - Asynchronous Metadata advertisement
- Request router address per dCDN footprint
- uCDN fallback address

dCDN request router address advertisement

```
"capabilities": [
 {
  "capability-type":  "FCI.RequestRouterAddress",
  "capability-value": {
     "address": <endpoint object>
  },
  "footprints": [
     <Footprint objects>
  ]
 }
]
```

Request router address per dCDN footprint

- **Use cases**
  - Different RRs for different footprints
  - Scaling by adding more RRs in new locations
- **Proposal**
  - Use FCI for RR address advertisement
  - Add new capability object FCI.RequestRouterAddress

uCDN fallback address

```
"generic-metadata-type": "MI.FallbackAddress",
  "generic-metadata-value":
   {
    "endpoints": [
     "fallback-a.service123.ucdn.example",
     "fallback-b.service123.ucdn.example"
     ]
   }
```

- **Use cases**
  - No cache available (e.g. network failure)
  - Cache cannot properly handle a request
- **Proposal**
  - uCDN to have a fallback address. Requests arriving to that address will not redirected to dCDN
  - Use Metadata to advertise uCDN fallback address
  - Add a new generic metadata type: MI.FallbackAddress

CONTENT MANAGEMENT

Content management

- Open Caching Content Management is a set of content operations that CDN can instruct the ISP to execute
  - Purge
  - Revalidate / Invalidate
  - Pre-position
- The content management interface is built on CDNI Control Interface / Triggers RFC8007

Content matching by regexp

- **Use case**
  - Purging specific content within a specific directory path. In some cases wildcard MAY be constraining or overreaching and may expose the assets to purge further than desired.
- **Proposal**
  - Add ***content.regexs*** to trigger specification

Content matching by playlist

- **Use case**
  - Pre-position requires a full list of objects. Playlist / manifest file is a natural interface for video CDN to pass a list of content objects
- **Proposal**
  - Add ***playlist.urls*** to trigger specification
  - Add FCI object to advertise which types and versions of manifests (HLS m3u8, MSS, DASH MPD) are supported

Geo limits

```
"locations": [
      {
       "action": "allow" / "deny",
       "footprints": [
        {
         "footprint-type": "countrycode",
         "footprint-value": ["us"]
        }
       ]
      }
    ],
```

- A trigger operation may apply for a specific geo, or should be excluded from a specific geo.
  The limit here is on cache location rather than client location
- **Use case**
  - In some cases, due to regulatory or royalties reasons, certain content cannot ever knowingly touch servers in a specific country – including caches. Therefore, these geos would need to be excluded from a pre-positioning operation
- **Proposal**
  - Add a GEO locations as an option in the trigger specification, for example:
  - Discussion point: should we use MI.LoactionAcl which was designed for client footprint rather than caches' ?

```
"time-windows": [
      {
       "time-type": "local" / "UTC",
       "start": "<seconds since UNIX epoch>",
       "end": "<seconds since UNIX epoch>"

      }
     ],
```

Scheduled triggers

- A uCDN may wish to perform an operation on the dCDN with a defined **local time** schedule.
- **Use case**
  - A content provider wishes to pre-populate a new episode at off-peak time (reduced costs) so that it would be ready on caches (for example home caches) at prime time when the episode is released for viewing. This requires an interface to direct the dCDN when to pre-position the content; the time frame is local time per area as the off-peak time is also localized.
- **Proposal**
  - Add a time-windows object as an option in the trigger specification
  - Discussion point: should we use MI.TimeWindowAcl which was designed for access restriction rather than operation execution window ?

Trigger Extensibility

```
"generic-trigger-spec-type": <type-name>,
"generic-trigger-spec-value":
    {
      <properties of this object>
    }
```

- Enabling extensions for the trigger spec
- **Use case**
  - Invalidation under some scope, for example content that was acquired in the past two hours
  - Cache layer pre-position, for example pre-position only at home caches
- **Proposal**
  - Add trigger extensibility mechanism using generic objects
- Note, a generic object can support the geo limit and scheduled trigger

Capabilities

- The capabilities added to the triggers interface are not mandatory to support and are, therefore, best negotiated via the FCI
- **Use case**
  - Advertise which content operations are supported
  - Advertise which content matching types are supported (HLS / DASH / SS)
  - Advertise which trigger spec objects are supported
- **Proposal**
  - Define generic objects mechanism, similar to metadata generic objects. FCI can declare which generic objects are supported.

AUTHENTICATION & ACCESS CONTROL

Request Authentication and Access Control at the dCDN

- Different client access control and authentication methods are used between CPs and CDNs
- A dCDN cannot implement all methods any uCDN supports
- Some methods are under NDA
- Sharing of symmetric keys used by CP / uCDN with the dCDN is a security challenge, expanding existing trust boundaries

Use cases

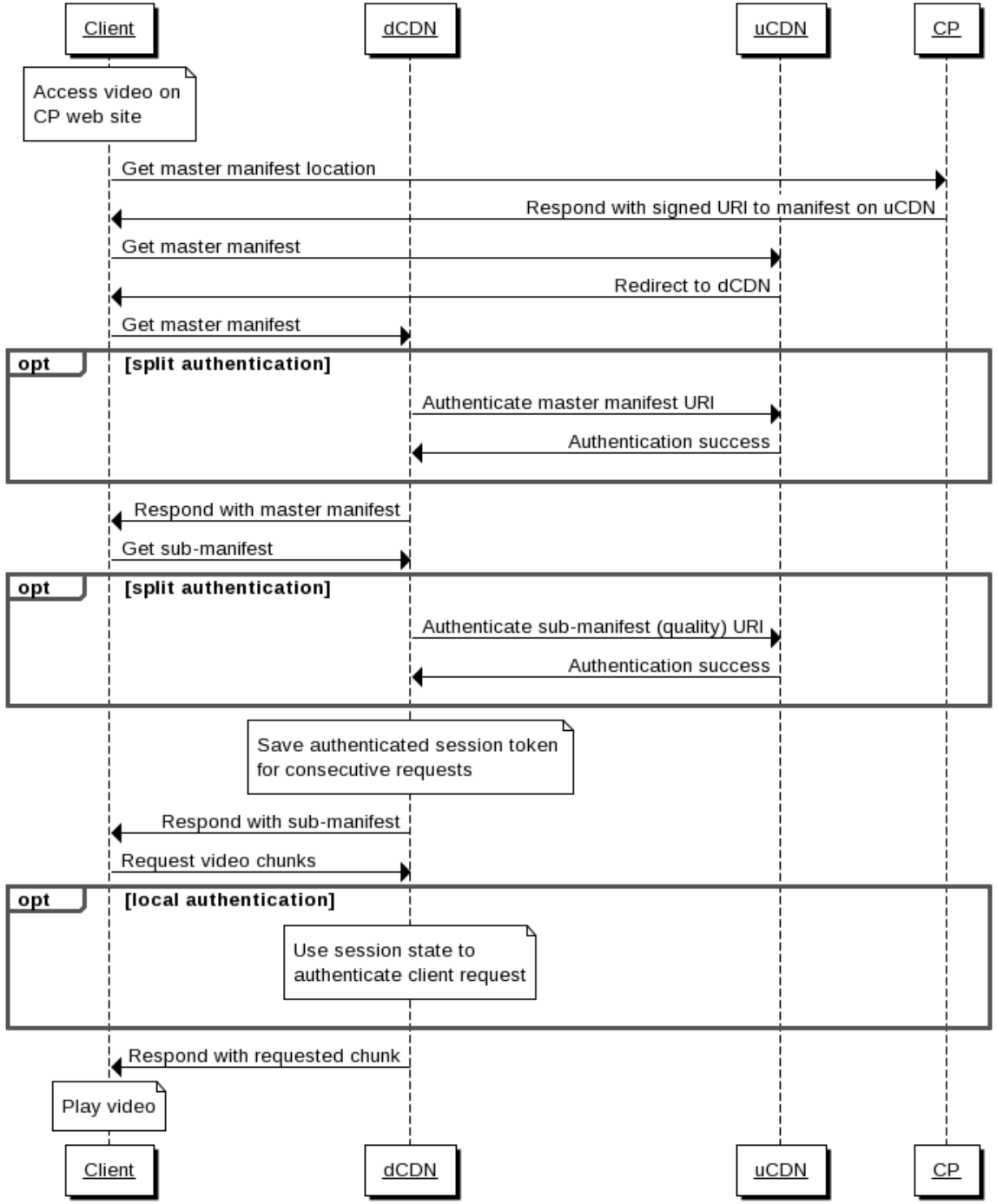
- URI signing
- Token based authentication
- CORS delegation (see COR section)

Relayed token authentication

- Designed for long ABR sessions or long progressive download transactions
- Minimum exposure of dCDN to authentication or access control algorithm (only expose the token fields)
- Zero exposure of the dCDN to CP symmetric keys
- Leave the verification logic in the uCDN, enable the dCDN to verify consecutive requests using verification state

Relayed token authentication

- Video sessions are long and chunked into small requests
- dCDN cache relays the authentication verification to the uCDN by sending a HEAD request for a new session
- dCDN cache caches the session state for a defined time and uses it for subsequent requests of the same session
- URI signing sequence example



Proposal

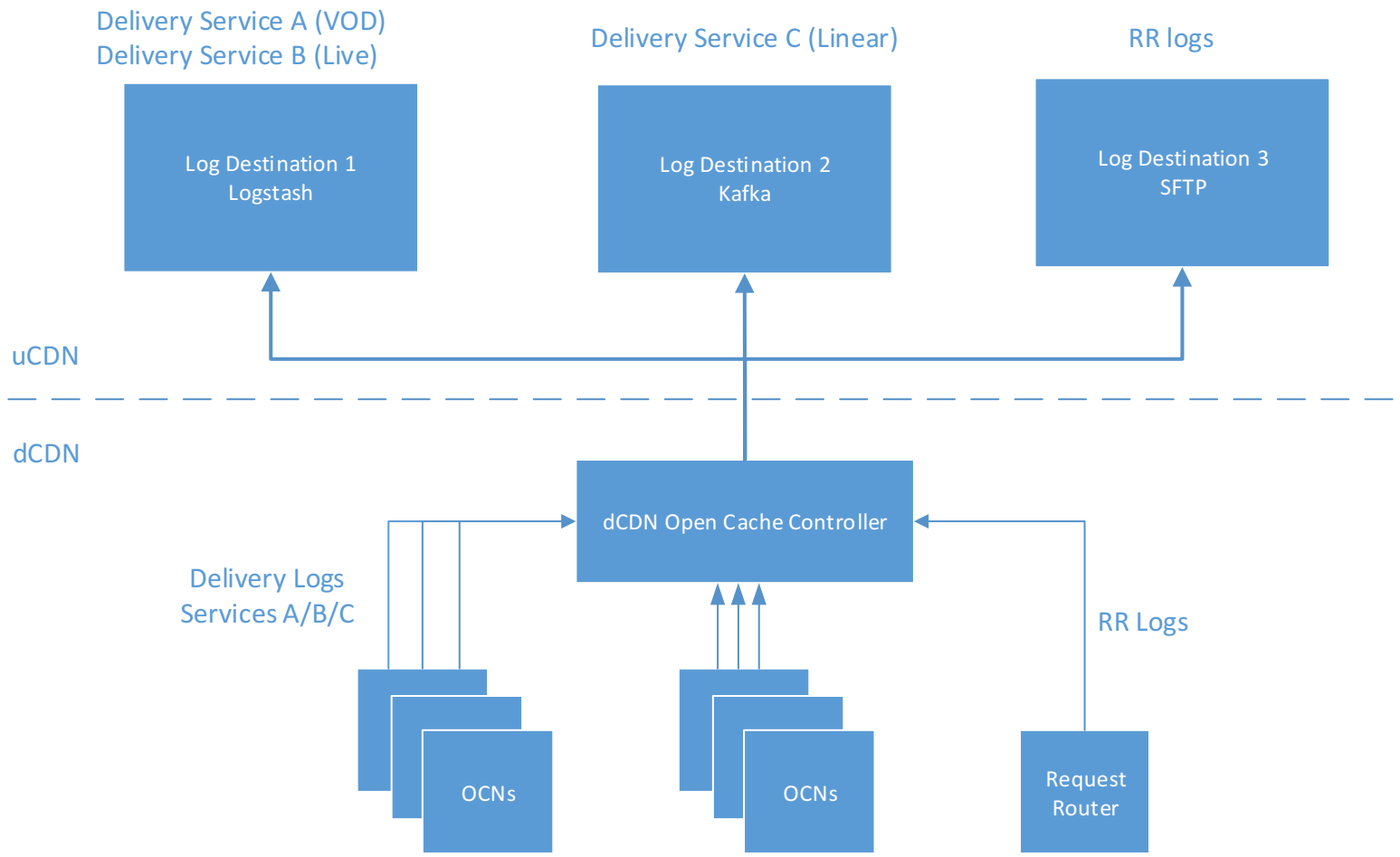
- We propose to publish internet draft specifying the Relayed Token Authentication mechanism
- Add new auth-type to MI.DeliveryAuthorization metadata
  - auth-type: MI.RelayTokenAuth
    - **Title cache key directives** – parts of the URL to be used for the title cache key
    - **Token fields** – query parameter used as authentication token
    - **Session keys** – fields from query params and headers used as part of the session key
    - **Client IP** – required client IP authentication
    - **TTL** – TTL field in query param or headers
- FCI - Advertise support for MI.RelayTokenAuth delivery authorization type as a generic metadata object

LOGGING

Log format and Log Integration

- Open Caching Logging Requirements document specifies a sub-set of KPIs based on Squid Log format
- Open Caching logging uses **Squid log format** as it is the common format used by commercial CDNs
- Open Caching Logging Integration specification defines the process of provisioning log transfers
  - dCDN advertises supported Log Integration Types via the footprint and Capabilities interface
  - uCDN provisions Log Integration services via the Logging Configuration Interface (specified in Logging Int. specification)
  - dCDN configures Log producing entities
  - Log records are transferred from dCDN entities to the uCDN destination(s)

Log integration system



Proposal

- Define an FCI.Logging capability
  - Transport types
  - Record types
  - Log file format types
  - Hashed fields
  - Optional fields
- Add MI.Logging generic metadata object to setup required log for specific service
  - Log destination type and address
  - Record structure and optional fields
  - Log file format
- Add Squid log (and perhaps other format) as alternative complementary formats
- Discussion: possibly publish an internet draft to describe the log integration provisioning process

CORS DELEGATION

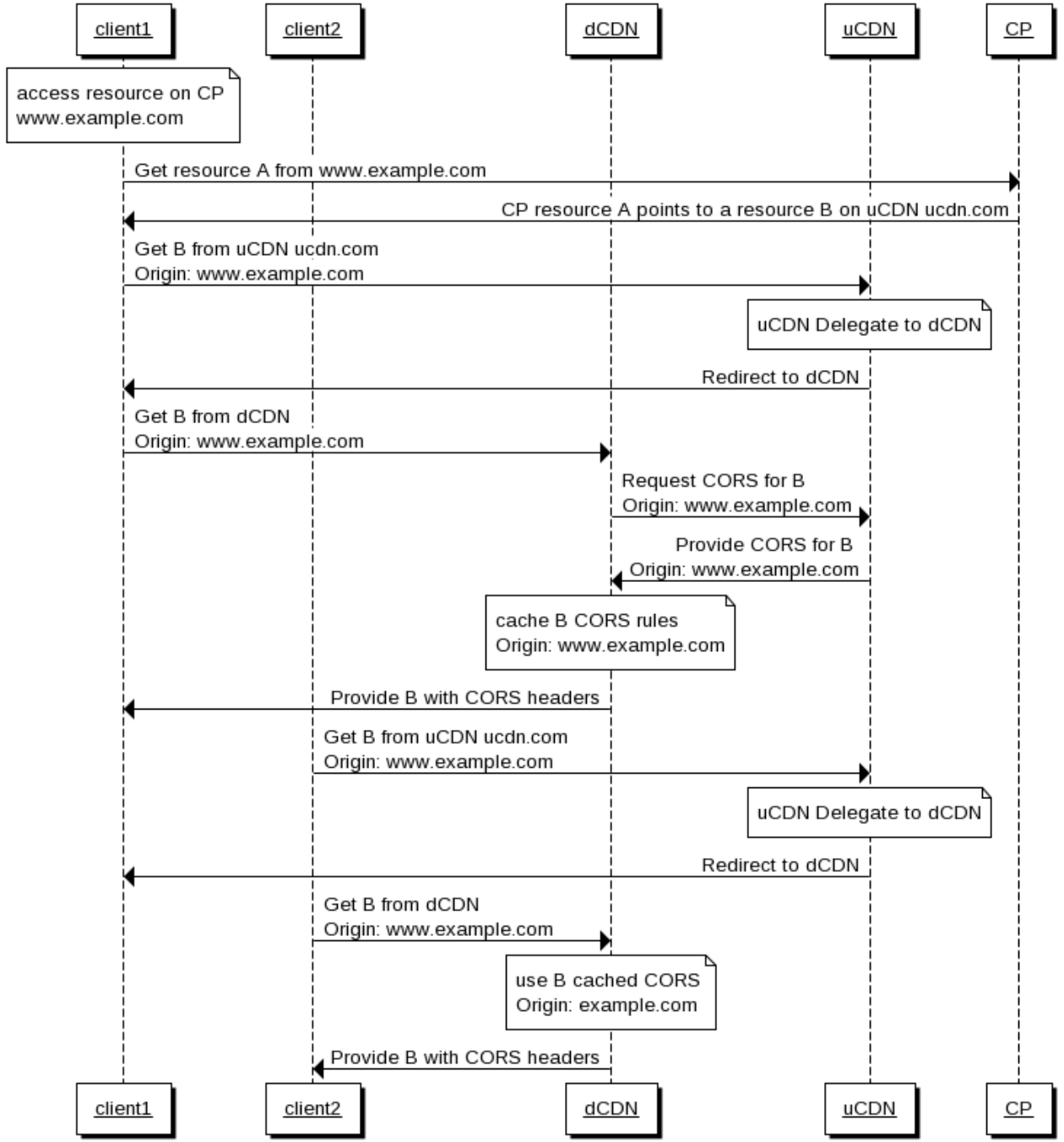
CORS delegation

- The dCDN is required to comply with the same CORS server behavior of the uCDN
- If on the uCDN domain A resource is accessible for resources of domain B but not for resources of domain C, the same logic must be performed by the dCDN
- Use case
  - At the uCDN Origin:www.video.example.com is allowed while other domains should be rejected.
  - For Origin: www.video.example.com the dCDN should reply with Access-Control-Allow-Origin: www.video.example.com
  - For Origin: www.video.other.com the dCDN should reject by omitting the ACAO header

CORS delegation alternatives

- **Echo** – respond to client with CORS headers allowing the content, regardless of origin
  - Client can access the content
  - **CORS policy may be breached**
- **Delegation by metadata** – pass the required information for CORS response via metadata interface
- **Delegation by caching** – cache uCDN response and use them to respond properly to CORS requests
- **Delegation by session access control (split authentication)** – this is only needed if the CORS responses are dynamic, per session

Caching CORS



Proposal

- We propose to publish internet draft specifying the CORS delegation mechanism
- Add MI.CORS metadata object for delegating CORS domains
- FCI - Advertise support CORS delegation methods and for the metadata MI.CORS

```
{
    "generic-metadata-type": "MI.CORS",
    "generic-metadata-value": {
        "methods": [
            {
                "method": "GET",
                "allowed-origins": [http://foo.com , http://bar.com],
                "allowed-credentials": true,
                "expose-headers": true
            },
            {
                "method": "HEAD",
                "allowed-origins": [http://foo.com , http://bar.com],
                "allowed-credentials": true,
                "expose-headers": true
            },
            {
                "method": "OPTIONS",
                "allowed-origins": [http://foo.com , http://bar.com],
                "allowed-headers": ["X-<HEADER-A>", "X-<HEADER-B>"],
                " allowed-methods": ["GET", "HEAD", "OPTIONS"],
                "max-age": 86400
            }
        ]
    }
}
```

Discussion points

- Content operations Geo limit
  - should we use MI.LoactionAcl which was designed for client footprint rather than caches' ?
- Scheduled operations
  - should we use MI.TimeWindowAcl which was designed for access restriction rather than operation execution window ?
- Log integration
  - Publish an internet draft to describe the log integration provisioning process

- Q & A
- Next steps ?
- Thank You !

The Streaming Video Alliance

- The Streaming Video Alliance (SVA) is an industry consortium focusing on video streaming technologies
- The Open Caching Working Group (OCWG) objectives
  - To identify the critical components of a non-proprietary caching system
  - To establish basic architectural guidelines for implementation of an open caching system
- OCWG mission
  - Create the functional requirements and specifications for delegation of video traffic from commercial CDNs to ISPs caching layer
- OCWG ongoing work
  - Published requirements and specification documents
  - Ongoing trials and PoCs of the Open Caching architecture