CDN 2.3

Overview

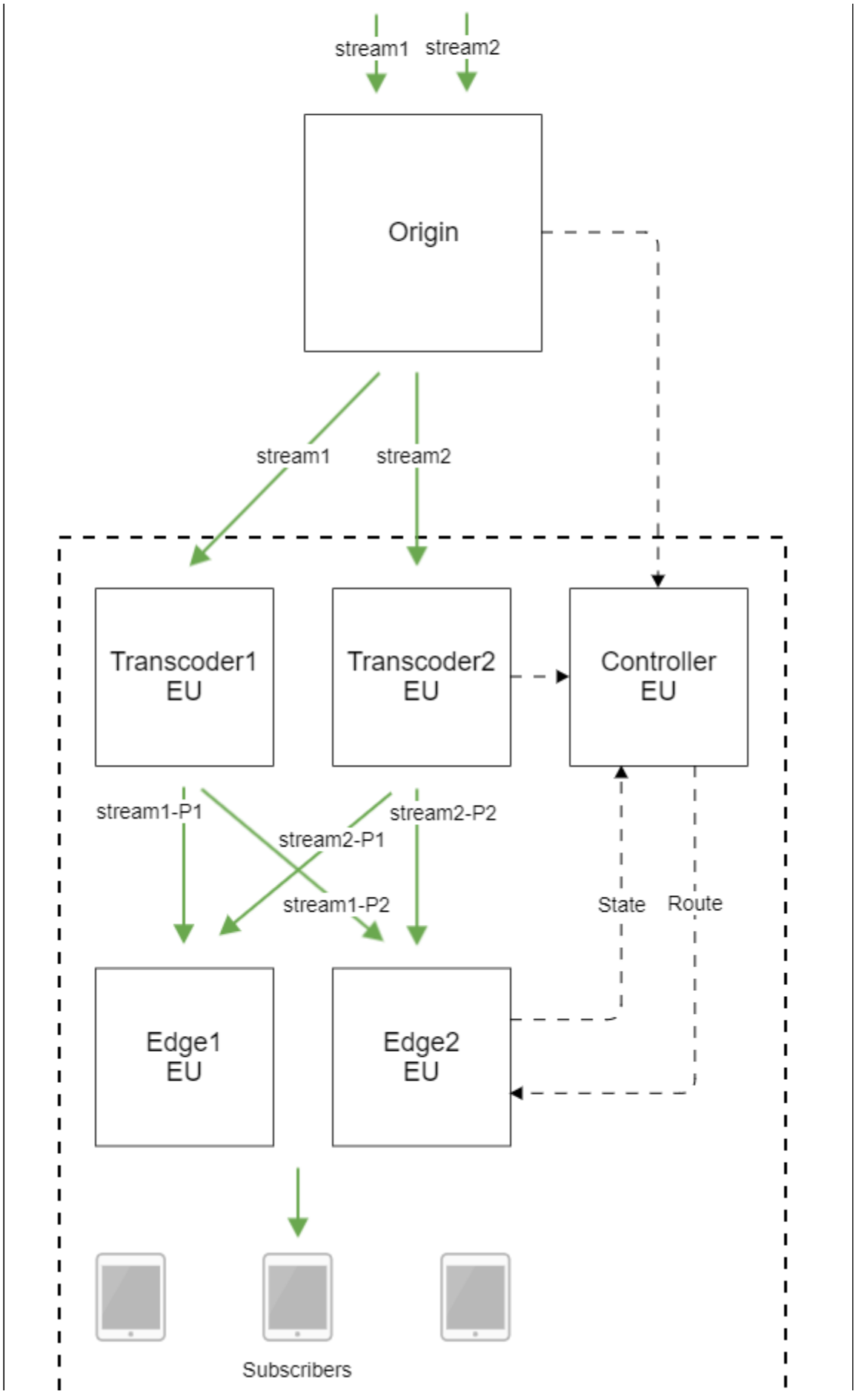
CDN 2.3 implements the following features in addition to CDN 2.1 and 2.2:

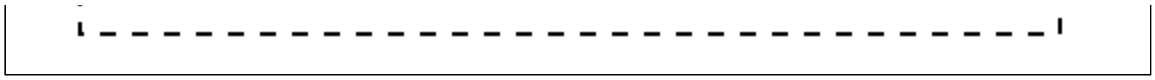
- prevention of redundant transcoding of the same stream on two or more transcoders in CDN
- CDN extended information displaying
- channel quality control between CDN nodes
- node state periodic sending

Prevention of redundant transcoding of the same stream on two or more transcoders in CDN

CDN 23 allows to prevent redundant transcoding of the same stream published to CDN on two or more transcoders in the same group. This can occur in large CDNs with big amount of incoming and outgoing media traffic, when a many subscribers connecting simultaneously to the CDN servers. It creates excessive load on transcoders which may lead to servers out of work.

To solve the problem, in every CDN group the special node with Controller role should be allocated which receives stream information from all th CDN nodes and collects current CDN state. The nodes with Edge role should request Controller node if it is available to choose a route for a stream playback. If the stream is already transcoded on any Transcoder node, Controller node does not allow to build the route for the stream playback trough another Trancsoder. If Controller node is not available, Edge node should choose a route for a stream playback according to CDN 2.1 method.


stream1
stream2
Origin
stream1
stream2
Transcoder1
EU
Transcoder2
EU
Controller
EU
stream1-P1
stream2-P1
stream2-P2
stream1-P2
State
Route
Edge1
EU
Edge2
EU
Subscribers



Controller node should not be used to publish or play any streams.

Configuration

Controller role should be set to node with the following parameter

```
cdn_role=controller
```

Controller request timeout can be defined on Edge node with the following parameter

```
cdn_controller_request_timeout=5000
```

The timeout is set in milliseconds, 5000 ms by default.

Controller response caching duration can be defined on Edge node with the following parameter

```
cdn_controller_response_cache_expire=10000
```

The cache duration is set in milliseconds, 10000 ms by default.

Controller response timeout and caching parameters should be set only on Edge nodes.

Redundant stream transcoding checking with REST API

Redundant stream transcoding can be checked with REST query to Edge server.

REST query should be HTTP/HTTPS POST request as follows:

- HTTP: `http://test.flashphoner.com:8081/rest-api/cdn/show_redundant_transcodings`
- HTTPS: `https://test.flashphoner.com:8444/rest-api/cdn/show_redundant_transcodings`

Where:

- `test.flashphoner.com` - WCS server address
- `8081` - a standard WCS REST / HTTP port
- `8444` - a standard WCS HTTPS port
- `rest-api` - mandatory part of URL
- `/cdn/show_redundant_transcodings` - REST query used

REST queries should be sent to Edge server.

REST queries and response states

REST query	Response body	Response status	Description
`/cdn/show_redundant_transcodings`	`{ "test6":"192.168.1.17,192.168.1.18" }`	200 Redundant transcoding found 404 No redundant transcoding	Check redundant transcoding

Parameters

Description	Example		
Stream name	`test6`	Transcoding nodes list	`192.168.1.17,192.168.1.18`

CDN extended information displaying

Since build 5.2.471 CDN information output was added or extended to statistics page, to CLI and to REST API

Using CLI

WCS CLI command

```
show cdn-nodes
```

displays CDN version supported by node and group which node belongs to

```
Ip            State  Processing state    Role       Version Group
-----------------------------------------------------------------
192.168.0.102 ACTIVE NEW_STREAMS_ALLOWED TRANSCODER 2.3     null
192.168.0.187 ACTIVE NEW_STREAMS_ALLOWED EDGE       2.3     null
```

In this case, CDN version matching result will be displayed for `ACTIVE` node and CDN version explicitly set will be displayed for `PASSIVE` node (see below)

Using REST API

REST API query `/cdn/show_nodes` returns CDN version supported by node

```
[
  {
    "version": "2.3",
    "role": "TRANSCODER",
    "globalState": "ACTIVE",
    "processingState": "NEW_STREAMS_ALLOWED",
    "id": "192.168.0.102"
  },
  {
    "version": "2.3",
    "role": "EDGE",
    "globalState": "ACTIVE",
    "processingState": "NEW_STREAMS_ALLOWED",
    "id": "192.168.0.187"
  }
]
```

In this case, CDN version matching result will be displayed for `ACTIVE` node and CDN version explicitly set will be displayed for `PASSIVE` node (see below)

Channel quality control between CDN nodes

Since build 5.2.483 channel quality control ability in Origin -> Transcoder -> Edge direction is added. To enable this feature, set the outbound bitrate sending interval value in seconds

```
outbound_video_rate_stat_send_interval=1
```

on Origin and Transcoder nodes.

In this case, the outbound and inbound bitrate values on subscriber node side are periodically comparing with publisher side one. The steady divergence of those values means channel bandwidth decrease. The peaks and sudden changes are smoothed by Kalman filter. Channel quality mertic and statistics for the stream can be checked by REST API query.

Checking channel quality metric and statistics using REST API

REST query should be HTTP/HTTPS POST request as follows:

- HTTP: `http://test.flashphoner.com:8081/rest-api/cdn/stats/print`
- HTTPS: `https://test.flashphoner.com:8444/rest-api/cdn/stats/print`

Where:

- `test.flashphoner.com` - WCS server address
- `8081` - REST / HTTP port

- `8444` - HTTPS port
- `rest-api` - mandatory URL part
- `/cdn/stats/print` - REST method used

REST queries should be sent to node from which statistics should be get. For example, on Edge node statistics can be checked for the stream received from Origin or Transcoder, and on Transcoder node for the stream received from Origin

REST queries and response states

REST query	Request body	Response bos y	Response stat us	Description

`/cdn/stats/pr int`	<pre>{
	
	"name":" test",
	
	"host":" 192.168. 0.111",
	
	"format" :"json" }</pre>
		<pre>{
		
		"192.168 .0.111": [
		
		{
		
		"fps": 30,
		
		"inbound Bitrate" : 1658292,
		
		"nack": 0,
		
		"name": "test",
		
		"outboun dBitrate ": 1692676,
		
		"quality ": "PERFECT "
		
		}
		]
		}</pre>
			200 OK 404 St ream not foun d	Get the strea m statisctics

Parameters

Parameter	Description	Example
name	Stream name	`test`
host	Node IP address from wh ich stream is receiving	`192.168.0.111`
format	Response format	`json`
inboundBitrate	Inbound (received) bitrate	`1658292`
outboundBitrate	Outbound (sent) bitrate	`1692676`
nack	NACK count	`0`
fps	FPS	`30`
quality	Channel quality metric	`PERFECT`

The query without stream name returns all the sreams statistics received from the CDN node defined. The qury without node returns all the streams statistics received from other CDN nodes if the stream name is not set

Response formats

By default, statistics is returned in JSON format:

```
POST /rest-api/cdn/stats/print HTTP/1.1
Content-Length: 19
Content-Type: application/json
Host: test.flashphoner.com:8081
{
    "name": "test"
}

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Length: 125
Content-Type: application/json
{
    "192.168.0.111": [
        {
            "fps": 31,
            "inboundBitrate": 1052794,
            "nack": 81,
            "name": "test",
            "outboundBitrate": 1048518,
            "quality": "PERFECT"
        }
    ]
}
```

Prometeus format is also supported:

```
POST /rest-api/cdn/stats/print HTTP/1.1
Content-Length: 43
Content-Type: application/json
Host: test.flashphoner.com:8081
{
    "format": "prometheus",
    "name": "test"
}

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Length: 397
Content-Type: plain/text

cdn_connection_quality{param="inboundBitrate",node="192.168.0.111",name="test"}
 1249049
cdn_connection_quality{param="outboundBitrate",node="192.168.0.111",name="test"}
 1240717
cdn_connection_quality{param="quality",node="192.168.0.111",name="test"} 3
cdn_connection_quality{param="fps",node="192.168.0.111",name="test"} 30
cdn_connection_quality{param="nack",node="192.168.0.111",name="test"} 81
```

Channel quality metric is shown as follows

JSON	Prometheus	Description
UNKNOWN	0	Quality is unknown, media packets are not received
BAD	1	Quality is bad
GOOD	2	Quality is good
PERFECT	3	Quality is perfect

Node state periodic sending

To reduce impact of channels quality between CDN nodes to CDN current state collection on each node, there was a couple of parameters added to control the period of CDN node state components sending to another nodes

Parameter	Default value	Description
`cdn_nodes_acl_refresh_interval`	60000	Interval to send ACL keys list, ms
`cdn_nodes_group_refresh_interval`	60000	Interval to send node groups, ms

Parameter	Default value	Description
`cdn_nodes_role_refresh_interval`	60000	Interval to send node role, ms
`cdn_nodes_route_refresh_interval`	60000	Interval to send node routes, ms
`cdn_nodes_state_refresh_interval`	60000	Interval to send node state, ms
`cdn_nodes_version_refresh_interval`	90000	Interval to send CDN version supported, ms

Backward compatibility with CDN 2.2, 2.1, 2.0

Controller node in previous CDN version

For previous CDN versions node, Controller node is Origin.

CDN 2.1 nodes will choose a route to stream playback without Controller node even if this node is in the group.

CDN versions matching

Since build 5.2.471, CDN versions matching works as follows: when CDN signaling connection is established, nodes send version supported to each other. If versions are differ, node with later version will fall it to earlier one. For example if Edge 2.3 is connecting to CDN 2.2, it will send CDN 2.2 compatible messages only to another nodes. At all the CDN point this Edge server will look like CDN 2.2 supporting node.

Every node periodically sends its version to another nodes to escape versions mismatch after losing the connection. The version refresh period is set in milliseconds with the following parameter

```
cdn_nodes_version_refresh_interval=90000
```

By default, version refresh period is 90 seconds.

`PASSIVE` node version assign

When node goes to `PASSIVE` state after losing the connection or by setting with `/cdn/enforce_state` REST query, the message exchange with this node will stop. Another nodes will suppose `PASSIVE` node to support CDN version set by the following parameter

```
cdn_force_version=2.0
```

By default, CDN 2.0 is assignade to `PASSIVE` node.

For example, if one Edge in CDN 2.3 loses the connection with another servers, CDN nodes will suppose this Edge to be CDN 2.0 until it restores CDN connection and passes CDN versions matching.

Known limits

1. To choose a stream playback route using Controller node, Edge node should be in the same group, and this group must be set in CDN node settings:

   ```
   cdn_groups=g1
   ```

2. Controller node choose all the routes through Transcoder nodes in the same group only.
3. Controller node without any group set cannot be used to choose a route.