В браузере по HLS

Описание

HTTP Live Streaming (HLS) — это технология воспроизведения потокового видео по протоколу HTTP, разработанная Apple. HLS видеопоток кодируется в H.264 и AAC и проигрывается на любом совместимом устройстве, браузере или плеере.

Web Call Server конвертирует в HLS видео, полученное из других поддерживаемых источников трансляции, таких как веб-камеры и профессиональные устройства видеозахвата, SIP-звонки и т.д..

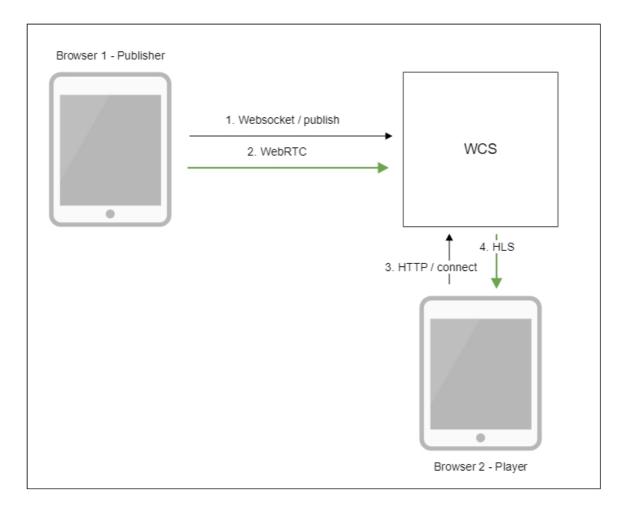
Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari	Edge
Windows	V	V	×	V
Mac OS	V	V	V	V
Android	V	V	×	V
iOS	V	V	V	×

Поддерживаемые кодеки

Видео: H.264Аудио: AAC

Схема работы



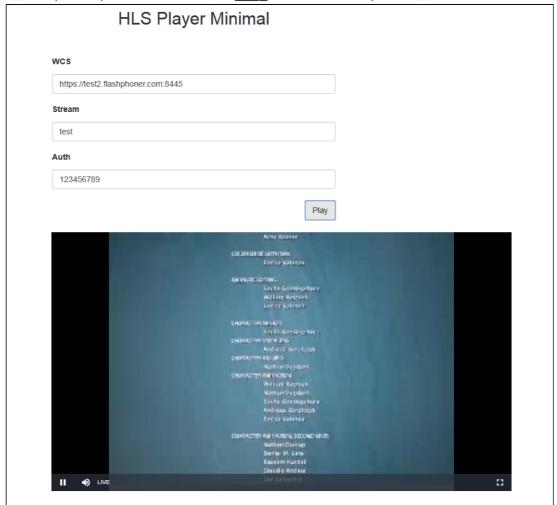
- 1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду publishStream.
- 2. Браузер захватывает микрофон и камеру и отправляет WebRTC поток на сервер.
- 3. Второй браузер устанавливает соединение по НТТР.
- 4. Второй браузер получает HLS поток и воспроизводит этот поток на странице.

Краткое руководство по тестированию

- 1. Для теста используем:
 - WCS сервер
 - веб-приложение Two Way Streaming для публикации потока
 - веб-приложение HLS Player Minimal для воспроизведения потока
- 2. Откройте веб-приложение Two Way Streaming. Нажмите Connect. затем Publish. Скопируйте идентификатор потока:



3. Откройте веб-приложение HLS Player Minimal. Укажите в поле Stream идентификатор потока и нажмите Play. Начнется воспроизведение потока:

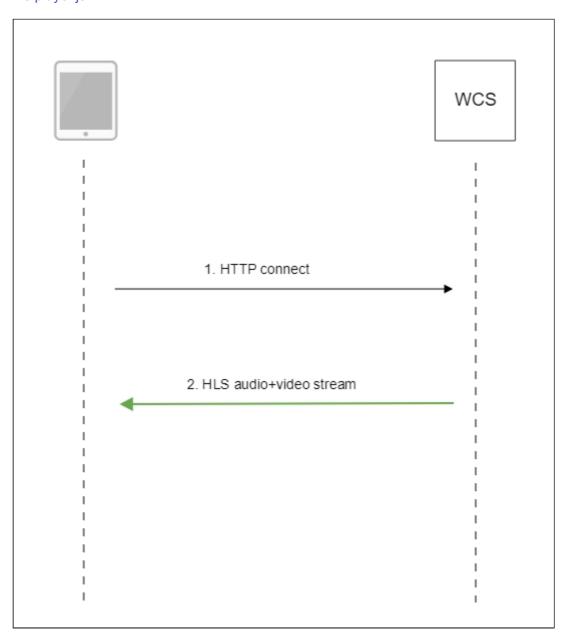


Последовательность выполнения операций

Ниже описана последовательность вызовов при использовании примера HLS Player Minimal для воспроизведения потока по HLS

hls-player.html

hls-player.js



1. Обращение к серверу и воспроизведение code

var player = videojs('remoteVideo');

Hастройка HLS URL

code

```
player.src({
    src: $("#urlServer").val() + "/" + streamName + "/" + streamName +
    ".m3u8",
    type: "application/vnd.apple.mpegurl"
});
```

Запуск воспроизведения

code

```
player.play();
```

2. Получение HLS-потока от сервера

Типы потоков, воспроизводимых по HLS

По HLS может быть воспроизведен любой поток, опубликованный на WCS под заданным именем https://wcs:8445/streamName/streamName.m3u8

Имя может быть задано при публикации из браузера или RTMP кодировщика, либо захвате RTSP, RTMP или VOD потока

Начиная со сборки 5.2.771, можно указать URI RTSP

https://wcs:8445/rtsp%3A%2F%2Frtspserver%2Flive.sdp/rtsp%3A%2F%2Frtspserver%2Fli

RTMP потока

https://wcs:8445/rtmp%3A%2F%2Frtmpserver%3A1935%2Flive%2Fstream/rtmp%3A%2F%2Frtm

или файла для VOD live трансляции

https://wcs:8445/vod-live%3A%2F%2Ffile.mp4/vod-live%3A%2F%2Ffile.mp4.m3u8

В этом случае поток будет захвачен из указанного источника, и после публикации на сервере начнется его проигрывание по HLS. Обратите внимание, что URI должен быть закодирован, все символы, кроме алфавитно-цифровых, должны быть экранированы.

В сборке 5.2.1679 добавлена возможность играть поток по URI источника, как HLS ABR

http://wcs:8082/vod-live%3A%2F%2Ffile.mp4-HLS-ABR-STREAM/vod-live%3A%2F%2Ffile.mp4-HLS-ABR-STREAM.m3u8

При обращении к Edge серверу в CDN, если поток с указанным именем или URI опубликован на Origin сервере, по HLS начнет проигрываться поток из CDN. Если такого потока в CDN нет, Edge попытается захватить поток по указанному URI локально.

Автоматическая нарезка HLS сегментов для опубликованного потока

При необходимости, любой из потоков, опубликованных на сервере по WebRTC, RTMP, MPEG-TS, или захваченный из RTSP или RTMP источника по REST API, может автоматически нарезаться на HLS сегменты. Эта возможность включается настройкой

hls_auto_start=true

В сборке 5.2.1895 добавлена возможность автоматической нарезки HLS ABR, при условии, что используется HLS ABR на одном узле. Эта возможность включается настройкой

hls_abr_auto_start=true

Аутентификация воспроизведения HLS с помощью REST hook

При необходимости, может быть настроена аутентификация клиентов для воспроизведения потока по HLS. В файле flashphoner.properties должна быть установлена настройка

hls_auth_enabled=true

При обращении к потоку на клиенте в HLS URL необходимо добавить параметр с указанием токена, полученного, например, от бэкенд-сервера. Наименование параметра задается настройкой

client_acl_property_name=aclAuth

В этом случае, обращение к потоку должно быть сформировано следующим образом:

```
var src = $("#urlServer").val() + "/" + streamName + "/" + streamName +
".m3u8";
var token = $("#token").val();
if (token.length > 0) {
   src += "?aclAuth=" +token;
}
```

На бэкенд-сервере в приложении defaultApp должен быть реализован REST hook /playHLS. WCS сервер отправляет на бэкенд запрос, содержащий полученный от клиента токен

```
URL:http://localhost:8081/apps/EchoApp/playHLS

OBJECT:
{
    "nodeId" : "NTk1tLorQ001lGbPJuFexrKceubGCR0k@192.168.1.5",
    "appKey" : "defaultApp",
    "sessionId" : "/192.168.1.100:59473/192.168.1.5:8445",
    "mediaSessionId" : "60709c5b-6950-40c3-8a3d-37ea0827ae32-
727473703a2f2f73747238312e63726561636173742e636f6d2f6772616e646c696c6c6574762f6c
HLS",
    "name" : "test",
    "mediaProvider" : "HLS",
    "custom" : {
        "aclAuth" : "12345789"
    }
}
```

Бэкенд сервер должен вернуть 200 ок, если токен клиента проходит проверку, и 403 Forbidden, если не проходит. В свою очередь, клиент получает либо HLS поток, либо 401 Unauthorized.

Настройка

```
hls_auth_token_cache=10
```

задает время кэширования токена в секундах (по умолчанию 10 секунд). До тех пор, пока токен находится в кэше, т.е. либо есть подписчик потока с таким токеном, либо не истекло указанное время, запросы /playHLS с этим токеном не отправляются на бэкенд. Если настройка кэширования установлена в 0

```
hls_auth_token_cache=0
```

запросы /playHLS отправляются на бэкенд при каждом HTTP GET запросе от клиента.

Эти настройки могут быть изменены без перезапуска сервера. При этом настройка hls_auth_enabled влияет на существующих подписчиков, а настройка <a href="https://historycommons.org/line-nable-na

Использование собственного приложения на бэкенде для аутентификации

В сборке 5.2.1008 добавлена возможность указать ключ приложения для аутентификации в HLS URL, например

https://wcs:8445/streamName/streamName.m3u8? appKey=customAppKey&aclAuth=1254789

В этом случае запрос /playHLS будет отправлен в указанное приложение(customAppKey в примере выше)

Предотвращение несанкционированного доступа к сегментам HLS

Для снижения нагрузки на сервер, проверка токена, а также доступности потока в CDN производится при запросе плейлиста. Для защиты отдельных сегментов в сборке 5.2.436 добавлена настройка

hls_segment_name_suffix_randomizer_enabled=true

В этом случае к имени файла сегмента добавляется случайным образом сгенерированный суффикс, например

test16d2da4658f4374953a120f3c95bc715ea.ts

Таким образом, исключается перебор сегментов на стороне клиента.



Attention

Суффикс не добавляется к сегментам прелоадера.

Добавление HTTP-заголовков для управления кроссдоменным воспроизведением HLS

По умолчанию, в ответ 200 ок на запрос HTTP GET добавляются следующие заголовки:

```
Transmission Control Protocol, Src Port: 8082, Dst Port: 57994, Seq: 1, Ack: 421, Len: 506

■ Hypertext Transfer Protocol

    HTTP/1.1 200 OK\r\n

    Connection: keep-alive\r\n
    Content-Type: application/x-mpegURL\r\n
  ▷ Content-Length: 308\r\n
    Access-Control-Allow-Origin: *\r\n
    Access-Control-Allow-Methods: GET\r\n
    Access-Control-Max-Age: 3000\r\n
     \r\n
     [HTTP response 1/29]
     [Time since request: 0.000904000 seconds]
    [Request in frame: 24]
    [Next request in frame: 879]
     [Next response in frame: 881]
    File Data: 308 bytes
```

```
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET
Access-Control-Max-Age: 3000
```

При необходимости, если, например, воспроизводимый контент и страница HLS плеера находятся в разных доменах, можно добавить собственные заголовки при помощи следующей настройки в файле flashphoner.properties:

```
hls_access_control_headers=Access-Control-Allow-Origin: *;Access-Control-Allow-Methods: GET, HEAD;Access-Control-Max-Age: 3000;Access-Control-Expose-Headers: Accept-Ranges, Content-Range, Content-Encoding, Content-Length
```

В этом случае в ответ 200 ок будут добавлены заголовки, перечисленные в настройке:

```
▶ Transmission Control Protocol, Src Port: 8082, Dst Port: 58646, Seq: 1, Ack: 554, Len: 476

■ Hypertext Transfer Protocol

▶ HTTP/1.1 200 OK\r\n

Connection: keep-alive\r\n

Content-Type: application/x-mpegURL\r\n

▶ Content-Length: 177\r\n

Access-Control-Expose-Headers: Accept-Ranges, Content-Range, Content-Encoding, Content-Length\r\n

Access-Control-Allow-Origin: *\r\n

Access-Control-Allow-Methods: GET, HEAD\r\n

Access-Control-Max-Age: 3000\r\n

\r\n

[HTTP response 1/9]

[Time since request: 3.796755000 seconds]
```

Поддержка маски в АСАО заголовке

В некоторых случаях, например, при использовании балансировщика нагрузки AWS LB, в ACAO заголовке, передаваемом в ответ на запрос GET, необходимо указать источник запроса с точностью до порта, например

Access-Control-Allow-Origin: https://test.flashphoner.com:8444

Однако, при конфигурировании сервера, этот адрес не всегда известен. В связи с этим в сборке 5.2.755 была добавлена настройка, которая включает поддержку маски при указании ACAO заголовка в настройках сервера

```
hls_acao_header_domain_mask=true
```

По умолчанию, данная возможность включена. При этом, если указать в настройке символ ★

```
hls_access_control_headers=Access-Control-Allow-Origin: *
```

сервер вернет в ответе на запрос GET ACAO заголовок с указанием источника запроса

```
Access-Control-Allow-Origin: https://lb.yourdomain.com:8444
```

При необходимости, данное поведение можно отключить

```
hls_acao_header_domain_mask=false
```

Использование nginx в качестве обратного прокси для воспроизведения по HLS

В некоторых случаях для воспроизведения потока с сервера по HLS может быть использован веб-сервер nginx в качестве обратного прокси. Как правило, это может потребоваться для обхода ограничений на кросс-доменные запросы к различным портам, если добавление HTTP-заголовков не помогает.

Например, если браузер требует, чтобы страница HLS-плеера и HLS-поток находились в одном домене your.domain и были доступны по одному и тому же порту 443 (HTTPS), nginx должен быть настроен следующим образом:

```
# Перенаправляем HTTP-запросы с 80 порта на 443
server {
   listen 80;
   server_name docs.flashphoner.com;
   return 301 https://$server_name$request_uri;
}

# Сервер обслуживает HTTPS порт 443
server {
   listen 443 ssl;
   ssl_certificate /etc/letsencrypt/live/your.domain/fullchain.pem;
   ssl_certificate_key /etc/letsencrypt/live/your.domain/privkey.pem;
   server_name your.domain;
   server_tokens off;
   client_max_body_size 500m;
   proxy_read_timeout 10m;
```

```
root
              /usr/share/nginx/html;
 location / {
 error_page 404 /404.html;
     location = /40x.html {
 error_page 500 502 503 504 /50x.html;
     location = /50x.html {
 # Примеры веб-приложений будут доступны напрямую по адресу
https://your.domain/client2
 location /client2/ {
     alias /usr/local/FlashphonerWebCallServer/client2/;
 # Плейлисты и сегменты проксируются в наш домен на 443 порт, например
https://your.domain/test.m3u8
  location \sim* ^.+.(m3u8|ts)$ {
     proxy_pass https://localhost:8445;
     proxy_http_version 1.1;
     proxy_set_header Host $server_name:$server_port;
     proxy_set_header X-Forwarded-Host $http_host;
     proxy_set_header X-Forwarded-Proto $scheme;
     proxy_set_header X-Forwarded-For $remote_addr;
     proxy_set_header Upgrade $http_upgrade;
```

Может оказаться полезным кэширование HLS-потока. В этом случае nginx должен быть настроен следующим образом:

1. В секции http файла настроек /etc/nginx.conf указываются параметры кэша

```
proxy_cache_path /var/cache/nginx/proxy levels=1:2
keys_zone=proxy_cache:1024m max_size=2048m inactive=10d;
proxy_cache_min_uses 1;
proxy_ignore_headers X-Accel-Expires;
proxy_ignore_headers Expires;
proxy_ignore_headers Cache-Control;
```

2. В секции server файла настроек сайта настраивается кэширование HLSсегментов, при этом плейлисты не должны кэшироваться

```
location ~* ^.+.(ts)$ {
   proxy_pass https://localhost:8445;
   proxy_http_version 1.1;
```

Передача собственных параметров при запросе плейлиста

Если на обратном прокси сервере настроена авторизация зрителей, необходимо передавать собственные параметры при запросе плейлиста:

```
https://wcs:8445/test/test.m3u8?key1=value1&key2=value2
```

Передача собственных параметров поддерживается, начиная со сборки 5.2.1959. При этом переданные параметры будут указаны в плейлистах для каждого сегмента, это дает возможность корректно провести авторизацию на прокси сервере.

В плейлистах, кроме собственных параметров, будет также параметр sessionId, например

```
av_test_o/av_test_o.m3u8?sessionId=11&key1=value1&key2=value2
```

This is the internal WCS parameter to identify the subscriber for statistical purposes.

Отображение статических HTML страниц на порту HLS

Еще один способ обхода ограничений на кросс-доменные запросы в браузере - отображение статического контента, например, страницы плеера, на том же порту, который отдает HLS контент. Чтобы включить данную возможность, необходимо указать следующую настройку в файле flashphoner.properties

hls_static_enabled=true

Страница плеера должна располагаться в каталоге, определяемом настройкой

hls_static_dir=client2/examples/demo/streaming/hls_static

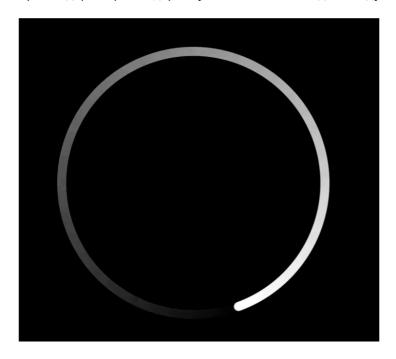
В данном случае (по умолчанию), путь к файлам страницы указан относительно каталога установки WCS. Может быть указан и полный путь, например

hls_static_dir=/var/www/html/hls_static

Если отображение статического контента включено, при обращении к WCS серверу по адресу https://host:8445/hls-player.html браузер отобразит страницу HLS плеера. Если данная возможность отключена, при обращении по такому адресу сервер вернет ошибку 404 Not found.

Preloader для воспроизведения потока по HLS

При подключении первого HLS-подписчика к потоку, в особенности к потоку из CDN, необходимо определенное время, чтобы началась нарезка потока на HLS-сегменты, и был сформирован плейлист. В результате, браузер Safari на устройствах iOS может не подключиться к потоку по HLS с первой попытки. Чтобы подключение всегда проходило успешно, в сборке 5.2.371 добавлено воспроизведение прелоадера. Прелоадер по умолчанию выглядит следующим образом



В сборке 5.2.408 прелоадеры разделены по соотношениям сторон картинки потока: 16:9, 4:3, 2:1

каталог /usr/local/FlashphonerWebCallserver/hls/.preloader при запуске сервера

```
tree /usr/local/FlashphonerWebCallServer/hls/.preloader
/usr/local/FlashphonerWebCallServer/hls/.preloader
  - 16x9
     — index0.ts
     — index10.ts
      index11.ts
      - index12.ts
      - index13.ts
      - index14.ts
      - index15.ts
      - index16.ts
     — index17.ts
      - index18.ts
      index19.ts
      - index1.ts
      - index2.ts
      - index3.ts
      - index4.ts
      — index5.ts
      _ index6.ts
      - index7.ts
      - index8.ts
    └─ index9.ts
   2x1
      - index0.ts
      - index10.ts
     — index11.ts
      index12.ts
      - index13.ts
      index14.ts
      - index15.ts
      - index16.ts
     - index17.ts
      — index18.ts
      index19.ts
      - index1.ts
      - index2.ts
      - index3.ts
       - index4.ts
       index5.ts
      index6.ts
      — index7.ts
      - index8.ts
    index9.ts
   4x3
     — index0.ts
      - index10.ts
     — index11.ts
      index12.ts
      - index13.ts
      - index14.ts
      - index15.ts
      - index16.ts
       - index17.ts
```

```
— index18.ts
— index19.ts
— index1.ts
— index2.ts
— index3.ts
— index4.ts
— index5.ts
— index6.ts
— index7.ts
— index8.ts
```

Минимальная длительность одного сегмента прелоадера по умолчанию составляет 2 секунды, и может быть задана в миллисекундах при помощи настройки

```
hls_preloader_time_min=2000
```

Отключение прелоадера

При необходимости, прелоадер может быть отключен, эта возможность доступна, начиная со сборки 5.2.396. Для отключения HLS прелоадера используется параметр

```
hls_preloader_enabled=false
```

Настройка собственного прелоадера

Чтобы заменить прелоадер по умолчанию на собственный, необходимо сделать следующее:

- 1. Выбрать видеоклип (например, логотип) в трех соотношениях сторон: 16:9, 4:3, 2:1
- 2. С помощью ffmpeg закодировать видео в H264, добавить к видеоклипу аудиодорожку, задать периодичность ключевых кадров и убрать В-фреймы

```
ffmpeg -i clip16x9.mp4 -f lavfi -i
anullsrc=channel_layout=mono:sample_rate=44100 -c:v h264 -g 30 -bf 0 -
shortest 16x9/preloader16x9.mp4
ffmpeg -i clip4x3.mp4 -f lavfi -i
anullsrc=channel_layout=mono:sample_rate=44100 -c:v h264 -g 30 -bf 0 -
shortest 4x3/preloader4x3.mp4
ffmpeg -i clip2x1.mp4 -f lavfi -i
anullsrc=channel_layout=mono:sample_rate=44100 -c:v h264 -g 30 -bf 0 -
shortest 2x1/preloader2x1.mp4
```

- 3. Загрузить и установить инструменты для подготовки HLS-сегментов с сайта Apple
- 4. Подготовить HLS сегменты прелоадера, указав длительность сегмента, например, 2 секунды

```
cd 16x9
mediafilesegmenter -t 2 -B index -start-segments-with-iframe
preloader16x9.mp4
tar -cvzf preloader.tar.gz index*.ts
```

Этот шаг необходимо повторить для всех соотношений сторон.

5. На сервере создать каталог для прелоадера

```
mkdir /opt/custom_preloader
mkdir /opt/custom_preloader/16x9
mkdir /opt/custom_preloader/4x3
mkdir /opt/custom_preloader/2x1
```

6. Распаковать прелоадер из архива, подготовленного на шаге 4

```
cd /opt/custom_preloader/16x9
tar -xvzf ~/preloader16x9.tar.gz
```

Этот шаг также необходимо повторить для всех соотношений сторон

7. Указать в настройках сервера расположение прелоадера и длительность одного сегмента

```
hls_preloader_time_min=2000
hls_preloader_dir=/opt/custom_preloader
```

Управление HLS подписками при помощи REST API

REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде:

- HTTP: http://test.flashphoner.com:8081/rest-api/hls/startup
- HTTPS: https://test.flashphoner.com:8444/rest-api/hls/startup

Здесь:

- test.flashphoner.com адрес WCS-сервера
- 8081 стандартный REST / HTTP порт WCS-сервера
- 8444 стандартный HTTPS порт
- rest-api обязательная часть URL
- /hls/startup используемый REST-метод

REST-методы и статусы ответа

/hls/startup

Запустить HLS нарезку указанного потока

Request example

```
POST /rest-api/hls/startup HTTP/1.1
Host: centos3.flashphoner.com:8081
Content-Length: 16
Content-Type: application/json
{
    "name":"test"
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

Return codes

Code	Reason
200	OK
404	Not found
500	Internal error

/hls/find_all

Найти все потоки, для которых есть HLS нарезки

Request example

```
POST /rest-api/hls/find_all HTTP/1.1
Host: test.flashphoner.com:8081
Connection: keep-alive
{
    "offset":0,
    "size":10
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
```

```
Content-Type: application/json
       "id": "test",
        "streamName": "test",
        "status": "ACTIVE",
        "waitingSize": 0,
        "profiles": [
            "a_test",
            "v_test"
        "subscribers": 1,
        "playlist": "#EXTM3U\n#EXT-X-VERSION:9\n#EXT-X-INDEPENDENT-
SEGMENTS\n#EXT-X-MEDIA:TYPE=AUDIO,URI=\"a_test/a_test.m3u8\",GROUP-
ID=\"audio\", NAME=\"none\", DEFAULT=YES, AUTOSELECT=YES, CHANNELS=\"2\"\n#EXT-X-
INF:BANDWIDTH=2180097, CODECS=\"avc1.640028, mp4a.40.2\", RESOLUTION=1280x720, FRAME
RATE=29.0, AUDIO=\"audio\"\nv_test/v_test.m3u8\n",
        "createdDate": 1697691514126,
        "logs": []
```

Return codes

Code	Reason
200	OK
404	Not found

/hls/terminate

Завершить или перезапустить HLS нарезку указанного потока

Request example

```
POST /rest-api/hls/terminate HTTP/1.1
Host: centos3.flashphoner.com:8081
Content-Length: 16
Content-Type: application/json

{
    "name":"test"
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

Return codes

Code	Reason
200	OK
404	Not found

/hls/profiles

Получить статистику профиля нарезки HLS

Request example

```
POST /rest-api/hls/profiles HTTP/1.1
Host: test.flashphoner.com:8081
Connection: keep-alive
{
    "hlsId":"test",
    "profileName":"v_test"
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
   "name": "v_test",
        "appKey": "defaultApp",
        "sessionId": "test-HLS",
        "mediaSessionId": "94bc92bc-959b-4533-a0e6-7de3d8c89141-test-HLS",
        "name": "test",
        "published": false,
        "hasVideo": false,
        "hasAudio": true,
        "status": "PLAYING",
        "sdp": "v=0\r\no=- 1988962254 1988962254 IN IP4 0.0.0.0\r\nc=IN IP4
0.0.0.0\r\nt=0 0\r\na=sdplang:en\r\nm=video 0 RTP/AVP 112\r\na=rtpmap:112
H264/90000\r\na=fmtp:112 packetization-mode=1; profile-level-
id=42001f\r\na=recvonly\r\n",
        "videoCodec": "H264",
        "record": false,
        "width": 1280,
        "height": 720,
        "minBitrate": 0,
        "maxBitrate": 0,
        "parentMediaSessionId": "8df817dc-c331-4fb5-949d-03e7764bab11",
```

```
"history": false,
    "gop": 0,
"fps": 0,
    "codecImpl": "",
    "transport": "UDP",
    "cvoExtension": true,
    "mediaType": "play",
        "muted": false
    "videoState": {
        "muted": false
    "mediaProvider": "HLS"
"keyFrameReceived": true,
    "type": "video",
    "width": 1280,
    "height": 720,
    "bitrate": 2129,
"codec": "",
    "quality": 0,
   "audioGroupId": "audio"
"metrics": {
    "minFPS": 29.962547,
    "avgFPS": 30.000261,
    "maxFPS": 30.04292,
    "countGaps": 0,
    "resolutionChanges": 0,
    "queueSize": 11,
    "startPts": 560866,
    "currentPts": 561133
```

Return codes

Code	Reason
200	OK
400	Bad request
404	Not found

/hls/subscribers

Получить список подписчиков на HLS нарезку

Request example

```
POST /rest-api/hls/subscribers HTTP/1.1
Host: test.flashphoner.com:8081
Connection: keep-alive
{
    "hlsId":"test"
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
        "id": "192.168.0.83-55832-Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.0.0 Safari/537.36",
        "ip": "192.168.0.83",
        "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/118.0.0.0 Safari/537.36",
        "metrics": {
            "profileTime": {
                "v_test": 598216
            "requestsNumber": 6537,
            "requestsStatuses": {
                "200 OK": 6536
            "profileSwitches": 1,
            "maxResponseTime": 13,
            "minResponseTime": 0,
            "avgResponseTime": 0.4173168119932691
```

Return codes

Code	Reason
200	OK
400	Bad request
404	Not found

/hls/connections

Получить список HLS клиентов, подключенных к серверу

Request example

```
POST /rest-api/hls/connections HTTP/1.1
Host: test.flashphoner.com:8081
Content-Type: application/json
{
    "offset":0,
    "size":10
}
```

Response example

Return codes

Code	Reason
200	OK
400	Bad request
404	Not found

/hls/enableRecording

Включить запись сегментов HLS-потока на диск

Request example

```
POST /rest-api/hls/enableRecording HTTP/1.1
Host: test.flashphoner.com:8081
Content-Type: application/json
{
```

```
"ids": [
    "test",
    "test-HLS-ABR-STREAM"
]
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

Return codes

Code	Reason
200	OK
400	Bad request
404	Not found
409	Conflict
500	Internal server error

/hls/disableRecording

Остановить запись сегментов HLS-потока на диск

Request example

```
POST /rest-api/hls/disableRecording HTTP/1.1
Host: test.flashphoner.com:8081
Content-Type: application/json

{
    "ids": [
        "test",
        "test-HLS-ABR-STREAM"
    ]
}
```

Response example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

Return codes

Code	Reason
200	ОК
400	Bad request
404	Not found

Параметры

Имя параметра	Описание	Пример
name	Имя потока, опубликов анного на сервере	test
hIsId	Имя потока, опубликов анного на сервере	test
profileName	Имя профиля качества, для которого запрашив ается статистика	v_test]
offset	Смещение от начала сп иска потоков в ответе на запрос статистики	0
size	Максимальное количес тво потоков в ответе н а запрос статистики	10
ids	Список имен HLS-поток ов, для которых включ ается/останавливается запись сегментов на ди ск	["test", "test-HLS- ABR-STREAM"]
state	Статус HLS нарезки	(ACTIVE)
logs	Сообщения о проблема x c LL HLS потоком	

Известные ограничения

1. Если HLS нарезка потока запущена при помощи REST запроса /hls/startup, и нет активных HLS подписчиков, нарезка остановится по истечении интервала в секундах

hls_manager_provider_timeout=300

По умолчанию, интервал составляет 5 минут. То же касается автоматически созданных HLS нарезок при установленной настройке

```
hls_auto_start=true
```

2. Если HLS нарезка потока останавливается при помощи REST запроса /hls/terminate и есть активные HLS подписчики, то нарезка будет перезапущена. При этом активные HLS подписчики должны повторно подключиться к потоку.

Отображение сообщений о проблемах с LL HLS потоком

В сборке 5.2.1709 добавлена возможность вывода сообщений о проблемах с LL HLS потоком в ответ на REST API запрос /hls/find_all:

По умолчанию, для каждого потока выводится до 50 последних сообщений. Это значение может быть изменено при помощи настройки

```
hls_metrics_log_size=50
```

Диагностируются следующие проблемы:

- Fps changed from x to y скачок FPS потока более 10 %
- Segment x does not start with keyframe интервал между ключевыми кадрами потока превышает длительность одного сегмента
- Playback speed changed to x скорость проигрывания потока изменилась
- Segment interval is too big слишком большой интервал между сегментами
- Video resolution changed from x to y разрешение потока изменилось

• Gap{from=x, to=y, duration=z} - пауза в потоке, в плейлист вставлен тэг EXT-X-GAP

Любая из этих проблем означает ухудшение качества исходной публикации, которая проигрывается по HLS, и может приводить к фризам, рассинхронизации звука и видео и остановке проигрывания в некоторых браузерах. В таких случаях следует принять меры по снижению помех на канале публикации, либо изменить способ публикации, например, с WebRTC на RTMP или SRT как более помехоустойчивые.

Отображение статистики HLS потока

В сборке 5.2.1777 добавлена возможность получения статистики HLS потока при помощи REST API

Данные о потоке в целом

В ответ на запрос /hls/find_all возвращается список всех HLS потоков на сервере с основной информацией о них

```
[ {
    "id": "test",
    "streamName": "test",
    "status": "ACTIVE",
    "waitingSize": 0,
    "profiles": [
      "a_test",
      "v_test"
    "playlist": "#EXTM3U\n#EXT-X-VERSION:9\n#EXT-X-INDEPENDENT-
SEGMENTS\n#EXT-X-MEDIA:TYPE=AUDIO,URI=\"a_test/a_test.m3u8\",GROUP-
ID=\"audio\", NAME=\"none\", DEFAULT=YES, AUTOSELECT=YES, CHANNELS=\"2\"\n#EXT-X-
STREAM-
INF:BANDWIDTH=1761281, CODECS=\"avc1.640028, mp4a.40.2\", RESOLUTION=1280x720, FRAME
RATE=29.0, AUDIO=\"audio\"\nv_test/v_test.m3u8\n",
    "createdDate": 1697605114475,
    "logs": []
}]
```

Здесь:

- id идентификатор HLS потока
- streamName имя исходного потока, который нарезается на HLS сегменты
- waitingSize количество HTTP запросов, ожидающих ответа
- profiles список аудио и видео профилей
- subscribers число HLS подписчиков
- playlist содержимое HLS манифеста
- createdDate дата создания в виде целого числа

• logs - список сообщений о проблемах с HLS потоком

При большом количестве потоков на сервере выдачу запроса можно ограничивать параметрами

```
{
    "offset":0,
    "size":10
}
```

Здесь:

- offset с какого элемента выводить список, по умолчанию 0
- size максимальное количество выводимых элементов, по умолчанию 10

Данные об аудио и видео профилях нарезки

В ответ на запрос /hls/profiles возвращается статистика аудио или видео профиля нарезки:

```
"name": "v_test",
    "appKey": "defaultApp",
   "sessionId": "test-HLS",
   "mediaSessionId": "81b8b278-612e-4b72-9153-454be9df0a34-test-HLS",
   "name": "test".
    "published": false,
    "hasVideo": false,
    "status": "PLAYING",
    "sdp": "v=0\r\no=- 1988962254 1988962254 IN IP4 0.0.0.0\r\nc=IN IP4
0.0.0.0\r\nt=0 0\r\na=sdplang:en\r\nm=video 0 RTP/AVP 112\r\na=rtpmap:112
H264/90000\r\na=fmtp:112 packetization-mode=1; profile-level-
id=42001f\r\na=recvonly\r\n",
    "videoCodec": "H264",
    "record": false,
    "width": 1280,
   "height": 720,
   "bitrate": 0,
   "minBitrate": 0,
    "parentMediaSessionId": "f3401d2e-7e9a-4e18-a353-d323c947ac94",
    "history": false,
    "gop": 0,
    "fps": 0,
    "audioBitrate": 0,
    "codecImpl": "",
    "transport": "UDP",
    "createDate": 1697605114875,
    "mediaType": "play",
    "audioState": {
```

```
"muted": false
    "muted": false
  "mediaProvider": "HLS"
"keyFrameReceived": true,
"videoProfile": {
 "type": "video",
  "width": 1280,
  "height": 720,
 "fps": 29,
 "bitrate": 1720,
  "quality": 0,
  "audioGroupId": "audio"
"metrics": {
 "minFPS": 29.962547,
 "avgFPS": 30.000088,
 "maxFPS": 30.04292,
  "countGaps": 0,
  "resolutionChanges": 0,
  "startPts": 375400,
 "currentPts": 375400
"subscribers": 1
```

Здесь:

- name имя профиля
- stream информация о потоке профиля, состав полей аналогичен ответу на запрос /stream/find
- keyFrameReceived получен ли хотя бы один ключевой кадр из исходного потока
- audioProfile, videoProfile исходные данные аудио или видео профиля:
 - type тип: видео или аудио
 - width заданная ширина картинки видео
 - height заданная высота картинки видео
 - fps частота кадров видео
 - bitrate заданный битрейт профиля
 - codec заданный кодек профиля
 - quality заданное качество видео профиля
 - audioGroupId идентификатор аудио профиля, заданный в видео профиле
 - rate частота дискретизации аудио профиля

- channels количество каналов аудио профиля
- groupId идентификатор аудио профиля для привязки видео
- metrics текущие метрики профиля:
 - minFPS минимальный FPS
 - avgFPS средний FPS
 - maxFPS максимальный FPS
 - countGaps количество пауз, вставленных в поток
 - resolutionChanges изменения разрешения видео
 - queueSize размер очереди кадров потока
 - startPts стартовая метка времени MPEG
 - currentPts текущая метка времени MPEG
- subscribers количество HLS подписчиков на данный профиль

Данные о подписчиках на HLS поток

В ответ на запрос /hls/subscribers возвращается статистика подписчиков на HLS поток:

Здесь:

- id идентификатор подписчика
- ір ІР адрес подписчика

- port исходящий порт подписчика
- userAgent данные заголовка User-Agent, переданные подписчиком
- active подписчик активен
- metrics текущие метрики подписчика:
 - profileTime время, в течение которого подписчик запрашивал данный профиль, указанное для каждого профиля
 - requestsNumber количество запросов от подписчика
 - requestStatuses количество статусов ответов на запросы подписчика, указанное для каждого запроса
 - profileSwitches число переходов подписчика с одного HLS ABR профиля на другой
 - maxResponseTime максимальное время ответа на запрос
 - minResponseTime минимальное время ответа на запрос
 - avgResponseTime среднее время ответа на запрос

Особенности отображения количества подписчиков и количества соединений HLS

В ответ на запросы /hls/find_all, /hls/profiles, /hls/subscribers возвращаются текущие количество и состав HLS подписчиков с точностью до вкладки браузера. Однако количество сетевых соединений для загрузки HLS, которое отображается на странице статистики сервера

curl -s 'http://localhost:8081/?action=stat¶ms=connections_hls'

может отличаться от количества подписчиков. В большинстве случаев, HLS подписчики используют HTTP 2 протокол, и все вкладки одного браузера, которые получают HLS потоки с одного WCS сервера, будут использовать одно соединение.

При этом количество соединений, отображаемое параметром connections_hls, соответствует количеству установленных соединений на HLS порт, отображаемых командой netstat:

```
sudo netstat -np | grep ESTABLISHED | grep java | grep 8445
```

Здесь 8445 - HTTPS HLS порт WCS сервера

Отображение HLS клиентов, подключенных к серверу

В сборке 5.2.1968 добавлена возможность выводить список HLS клиентов, подключенных в данный момент к серверу. Этот список возвращается в ответ на запрос /hls/connections:

Здесь:

- ір адрес клиента
- port исходящий порт клиента
- userAgent заголовок User-Agent , присланный клиентом

Для HTTPS клиентов все вкладки одного и того же браузера на одном и том же ПК будут представлены одной позицией в списке, посколько HTTP/2 предусматривает использование одного и того же TCP соедиения для всех HTTPS соединений.

Количество клиентов в списке будет соответствовать параметру статистики

```
curl -s 'http://wcs:8081/?action=stat&params=connections_hls'
```

Запись сегментов на диск для отладки

В сборке 5.2.1913 добавлена возможность записать сегменты и плейлисты определенного потока, который уже играют подписчики, на диск в целях отладки. Запись производится в каталог, указанный настройкой

```
hls_debug_dir=hls-debug
```

Запись сегментов определенного потока запускается при помощи REST API запроса /hls/enableRecording

```
POST /rest-api/hls/enableRecording HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "ids": [
        "test",
        "test-HLS-ABR-STREAM"
    ]
}
```

Для остановки записи используется REST API запрос /hls/disableRecording

```
POST /rest-api/hls/disableRecording HTTP/1.1
Host: localhost:8081
Content-Type: application/json
    "ids": [
       "test",
       "test-HLS-ABR-STREAM"
```

С записанными сегментами можно работать различными инструментами, предназначенными для разбора и проигрывания медиафайлов: ffmpeq, ffprobe и т.п. Кроме того, записанный поток можно проиграть с сервера по URL с суффиксом 🖪 DEBUG

```
https://wcs:8445/test-DEBUG/test-DEBUG.m3u8
```

Если поток с определенным именем был записан, а затем опубликован новый поток с таким же именем, запись нового потока будет сделана с удалением предыдущей.

Поддержка HLS ABR

Для потоков с видео (только видео или видео+аудио) WCS поддерживает HLS ABR в CDN (в этом случае качества кодируются на выделенном Transcoder сервере) и на одном узле.



Warning

Для потоков только с аудио HLS ABR не работает, при попытке запросить HLS ABR манифест для такого потока сервер вернет 404 Not found!

Устаревшая реализация HLS ABR



Warning

Используйте эту реализацию только в случае, если на сервер установлена сборка из диапазона 5.2.484 - 5.2.582

В сборке 5.2.484 добавленаподдержка HLS ABR плейлистов. Использование этой возможности включается при помощи настройки

hls_master_playlist_enabled=true

Имя основного плейлиста указывается при помощи настройки

```
hls_manifest_file=index.m3u8
```

Браузер должен запросить основной плейлист по URL

```
https://wcs_address:8445/streamName/index.m3u8
```

Здесь

- wcs_address адрес WCS сервера
- streamName имя потока на сервере
- index.m3u8 имя основного плейлиста

При запросе основного плейлиста сервер проверяет наличие потоков, транскодируемых из указанного потока согласно профилям транскодинга, перечисленным в файле настроек cdn_profiles.yml, например:

```
profiles:
 -720p:
   video:
     height: 720
     bitrate: 1000
     codec: h264
 -480p:
   video:
     height: 480
     bitrate: 1000
     codec: h264
 -240p:
   video:
     height: 240
     bitrate: 400
     codec: h264
```

Все транскодированные потоки по профилям, которые в момент запроса опубликованы на сервере, попадают в основной плейлист, например:

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
../streamName-720p/streamName-720p.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=852x480, CODECS="avc1.42e01f, mp4a.40.2"
../streamName-480p/streamName-480p.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=400000, RESOLUTION=426x240, CODECS="avc1.42e01f, mp4a.40.2"
../streamName-240p/streamName-240p.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=2500000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
../streamName/streamName.m3u8
```

Затем браузер, в зависимости от пропускной способности канала, переключается между потоками, перечисленными в основном плейлисте.

В момент, когда браузер запрашивает основной плейлист, потоки уже должны быть опубликованы на сервере и нарезаться на HLS сегменты

Чтобы обеспечить наличие потоков, необходимо:

- 1. На отдельно стоящем сервере:
- 2. 1.1. Периодически проверять, транскодируются ли потоки к указанным параметрам, и запускать транскодинг при необходимости при помощи REST API

```
curl -s -X POST -d "{\"uri\":\"transcoder://tcode_test-
240p\",\"remoteStreamName\":\"test\",\"localStreamName\":\"test-
240p\",\"encoder\":{\"width\":320,\"height\":240}}"
http://localhost:8081/rest-api/transcoder/startup
curl -s -X POST -d "{\"uri\":\"transcoder://tcode_test-
480p\",\"remoteStreamName\":\"test\",\"localStreamName\":\"test-
480p\",\"encoder\":{\"width\":640,\"height\":480}}"
http://localhost:8081/rest-api/transcoder/startup
curl -s -X POST -d "{\"uri\":\"transcoder://tcode_test-
720p\",\"remoteStreamName\":\"test\",\"localStreamName\":\"test-
720p\",\"encoder\":{\"width\":1280,\"height\":720}}"
http://localhost:8081/rest-api/transcoder/startup
```

3. 1.2. Периодически запускать HLS потоки для включения в основной плейлист, например

```
curl -s -X POST -d "{\"name\":\"test\"}" http://localhost:8081/rest-
api/hls/startup
sleep 1
curl -s -X POST -d "{\"name\":\"test-240p\"}" http://localhost:8081/rest-
api/hls/startup
sleep 1
curl -s -X POST -d "{\"name\":\"test-480p\"}" http://localhost:8081/rest-
api/hls/startup
sleep 1
curl -s -X POST -d "{\"name\":\"test-480p\"}" http://localhost:8081/rest-
api/hls/startup
sleep 1
curl -s -X POST -d "{\"name\":\"test-720p\"}" http://localhost:8081/rest-
api/hls/startup
```

4. На Edge сервере в CDN периодически запрашивать HLS потоки по профилям, например

```
curl -s http://localhost:8082/test/test.m3u8
sleep 1
curl -s http://localhost:8082/test-240p/test-240p.m3u8
sleep 1
curl -s http://localhost:8082/test-480p/test-480p.m3u8
sleep 1
curl -s http://localhost:8082/test-720p/test-720p.m3u8
sleep 1
```

Актуальная реализация HLS ABR



Attention

Используйте эту реализацию в сборках 5.2.585 и новее

В сборке 5.2.585 реализация HLS ABR существенно изменена. Как и прежде, HLS ABR может использоваться только в CDN, при этом все транскодированные потоки для вариантов в ABR манифесте Edge забирает одновременно в пределах одной медиасессии, чтобы варианты одного потока были синхронизированы друг с другом. Это требует совместной настройки Transcoder и Edge узлов и накладывает ряд ограничений. Рассмотрим их ниже.

Настройки Transcoder узлов

Для того, чтобы все варианты одного потока были синхронизированы между собой, на Transcoder узлах должно быть включено выравнивание кодирования

transcoder_align_encoders=true

Кроме того, должен быть включен FPS фильтр

video_filter_enable_fps=true
video_filter_fps=25

Ключевые фреймы в вариантах потока должны быть синхронизированы. Например, при 25 кадрах в секунду будем отправлять ключевой фрейм каждые 2 секунды

video_filter_fps_gop_synchronization=50

Hастройки HLS Edge узлов

Ha HLS Edge узлах необходимо отключить использование прелоадера и транскодирование потоков

hls_preloader_enabled=false
hls_player_width=0
hls_player_height=0

Необходимо также настроить профили транскодирования в файле cdn_profiles.yml

profiles: -240p: audio: codec : 1

codec : mpeg4-generic

rate : 48000

```
video:
   height : 240
   bitrate : 300
   gop : 50
   codec : h264
-480p:
   video:
   height : 480
   bitrate : 600
   gop : 50
   codec : h264
-720p:
   video:
   height : 720
   bitrate : 1000
   gop : 50
   codec : h264
```

Обратите внимание, что параметры звука можно указать для первого профиля, т.к. для всех профилей эти параметры должны быть идентичными и будут применены по первому из профилей.

Затем необходимо включить HLS ABR

```
hls_abr_enabled=true
```

Использование

Клиент должен запрашивать плейлист HLS ABR, указывая имя потока с суффиксом

```
https://server:8445/test_0-HLS-ABR-STREAM/test_0-HLS-ABR-STREAM.m3u8
```

Суффикс задается при помощи настройки

```
hls_abr_stream_name_suffix=-HLS-ABR-STREAM
```

Плейлист содержит ссылки на плейлисты вариантов потока, между которыми клиент может переключаться

```
#EXT-X-STREAM-
INF:BANDWIDTH=614400, RESOLUTION=852x480, CODECS="avc1.42e01f, mp4a.40.2"
-480p/-480p.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=1024000, RESOLUTION=1278x720, CODECS="avc1.42e01f, mp4a.40.2"
-720p/-720p.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=307200, RESOLUTION=426x240, CODECS="avc1.42e01f, mp4a.40.2"
-240p/-240p.m3u8
```

Если запросить поток без суффикса, будет играть HLS без поддержки ABR.

Предотвращение транскодирования к более высоким разрешениям

Если транскодирование к более высоким разрешениями отключено на HLS ABR Edge сервере

```
cdn_strict_transcoding_boundaries=true
```

то варианты, соответствующие профилям с более высокими разрешениями, для данного потока не будут нарезаться и не будут доступны плееру.

Известные ограничения

- 1. HLS Edge может быть использован только для воспроизведения HLS потоков, клиентские сессии с использованием других протоколов работать не будут.
- 2. Не работают такие функции, как запись, снятие снапшотов, микширование, захват потоков с другого сервера и прочие функции обработки потоков

HLS ABR на одном узле

В большинстве случаев, для проигрывания HLS ABR целесообразно использовать CDN, поскольку такое решение лучше масштабируется по вычислительной мощности. Однако, начиная со сборки 5.2.1582, поддерживается и транскодирование с нарезкой плейлистов по заданным качествам HLS ABR на одном узле

```
hls_abr_enabled=true
hls_abr_with_cdn=false
```

В этом случае необходимо отключить прелоадер и транскодирование к определенному разрешению, поскольку поток будет транскодироваться по заданным профилям

```
hls_preloader_enabled=false
hls_player_width=0
hls_player_height=0
```

Также необходимо включить выравнивание FPS при транскодинге

```
transcoder_align_encoders=true
video_filter_enable_fps=true
video_filter_fps=30
video_filter_fps_gop_synchronization=60
```

Профили транскодинга настраиваются в файле

/usr/local/FlashphonerWebCallServer/conf/hls_abr_profiles.yml

```
profiles:
 -180p:
     codec : opus
     rate: 48000
   video:
     height : 180
     bitrate : 300
     codec : h264
     codecImpl : OPENH264
     gop : 60
     fps : 30
 -240p:
     codec : opus
     rate : 48000
     height: 240
     bitrate : 500
     codec : h264
     codecImpl : OPENH264
     gop : 60
     fps : 30
 -480p:
   audio:
     codec : opus
     rate : 48000
     height: 480
     bitrate : 1000
     codec : h264
     codecImpl : OPENH264
     gop : 60
      fps : 30
     codec : opus
     rate : 48000
     height: 720
     bitrate: 1500
     codec : h264
     codecImpl : OPENH264
     gop : 60
      fps : 30
```

Для HLS ABR поддерживается и Low Latency HLS

```
hls_ll_enabled=true
hls_new_http_stack=true
```



Warning

При использовании HLS ABR на одном сервере, на этом сервере будет работать транскодинг к нескольким качествам для каждого опубликованного потока. Это потребует большого количества ядер CPU и оперативной памяти.

Предотвращение транскодирования к более высоким разрешениям

Начиная со сборки 5.2.1611, если поток, опубликованный на сервере, имеет разрешение по высоте меньшее, чем какой-то из вариантов, перечисленных в hls_abr_profiles.yml, то все варианты с большими разрешениями не будут кодироваться и не попадут в манифест, например, при публикации 960х540 манифест будет таким:

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=300000, RESOLUTION=320x180, CODECS="avc1.42e01f, mp4a.40.2"
180/180.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=500000, RESOLUTION=428x240, CODECS="avc1.42e01f, mp4a.40.2"
240/240.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=848x480, CODECS="avc1.42e01f, mp4a.40.2"
480/480.m3u8
```

поскольку транскодирования вверх до 1280х720 не будет.

Если разрешение потока, опубликованного на сервере, ниже минимального профиля, такой поток будет транскодирован к минимальному профилю, и в плейлисте будет только этот профиль

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=300000, RESOLUTION=320x180, CODECS="avc1.42e01f, mp4a.40.2"
180/180.m3u8
```

Если в списке профилей есть такой, для которого не указаны ни высота, ни ширина, по этому профилю поток будет транскодироваться с оригинальным разрешением и заданными FPS и GOP, и этот вариант будет всегда включаться в плейлист:

```
profiles:
 original:
   video:
      codec : h264
      codecImpl : OPENH264
      gop : 60
```

Очередность качеств в манифесте HLS ABR потока

До сборки 5.2.1606, качества в манифесте HLS ABR потока сортировались в алфавитном порядке именования профилей, например, при таком cdn_profiles.yml или hls_abr_profiles.yml

```
profiles:
 360:
   video:
     height: 360
     bitrate : 1000
     codec : h264
     codecImpl : OPENH264
     gop : 60
     fps : 30
   video:
     height: 720
     bitrate : 2000
     codec : h264
     codecImpl : OPENH264
     gop : 60
     fps : 30
 1080:
   video:
     height: 1080
     bitrate : 2500
     codec : h264
     codecImpl : OPENH264
     gop : 60
      fps : 30
```

манифест будет таким

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=2500000, RESOLUTION=1920x1080, CODECS="avc1.42e01f, mp4a.40.2"
1080/1080.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=640x360, CODECS="avc1.42e01f, mp4a.40.2"
360/360.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=2000000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
720/720.m3u8
```

Начиная со сборки 5.2.1606, порядок следования качеств в манифесте соответствует порядку перечисления профилей в cdn_profiles.yml или hls_abr_profiles.yml

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=1000000,RESOLUTION=640x360,CODECS="avc1.42e01f,mp4a.40.2"
360/360.m3u8
```

```
#EXT-X-STREAM-
INF:BANDWIDTH=2000000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
720/720.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=2500000, RESOLUTION=1920x1080, CODECS="avc1.42e01f, mp4a.40.2"
1080/1080.m3u8
```

При этом, если в настройке встречаются два профиля с одинаковыми названиями, сервер будет использовать только последний из профилей с одинаковыми именами.

Транскодирование максимального качества только при наличии Вфреймов в исходном потоке

Для того, чтобы снизить нагрузку на сервер при кодировании видео, в сборке 5.2.1840 добавлена возможность транскодировать максимальное ABR качество (которое обычно соответствует оригинальному разрешению потока) только при наличии В-фреймов в потоке. Эта возможность включается настройкой

```
h264_b_frames_force_transcoding=true
```

При этом сервер проверяет наличие В-фреймов в исходном потоке, анализируя заданное количество фреймов (по умолчанию 10)

```
frame_cnt_to_determine_their_type=10
```

Если в потоке есть В-фреймы, максимальное ABR качество будет транскодироваться, и будет доступно плееру в HLS манифесте

```
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=640x360, CODECS="avc1.42e01f, mp4a.40.2"
360/360.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=20000000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
720/720.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=25000000, RESOLUTION=1920x1080, CODECS="avc1.42e01f, mp4a.40.2"
1080/1080.m3u8
```

Если в потоке нет B-фреймов, максимальное ABR качество не будет транскодироваться

```
#EXTM3U
#EXT-X-STREAM-
INF:BANDWIDTH=1000000, RESOLUTION=640x360, CODECS="avc1.42e01f, mp4a.40.2"
360/360.m3u8
#EXT-X-STREAM-
INF:BANDWIDTH=2000000, RESOLUTION=1280x720, CODECS="avc1.42e01f, mp4a.40.2"
720/720.m3u8
```

В этом случае оригинальное качество необходимо запросить отдельно с клиента.

Начиная со сборки 5.2.1916, эта возможность доступна и для HLS ABR в CDN. Для этого все серверы в CDN должны быть обновлены до сборки 5.2.1916 или новее, и на Edge сервере указаны следующие настройки

cdn_strict_transcoding_boundaries=true
h264_b_frames_force_transcoding=true

Максимальный размер плейлиста

Максимальный размер плейлиста в сегментах задается настройкой

hls_list_size=8

По умолчанию размер HLS плейлиста равен 8 сегментам. Отметим, что, когда нарезка HLS только стартовала, количество сегментов в первых плейлистах будет меньше заданного.

Хранение сегментов HLS

Использование диска

В сборках до 5.2.1713 HLS-сегменты по умолчанию записываются на диск сервера, в каталог /usr/local/FlashphonerWebCallServer/hls. Начиная со сборки 5.2.687, каталог для сохранения сегментов можно изменить при помощи параметра

hls_dir=/usr/local/FlashphonerWebCallServer/hls

(Расположение прелоадера настраивается отдельно при помощи параметра hls_preloader_dir.)

На диске хранится количество сегментов, соответствующее заданному размеру плейлиста, по умолчанию 10

hls_list_size=10

Чем меньше количество сегментов в плейлисте, тем меньше задержка при воспроизведении. Однако при коротком плейлисте подписчики с недостаточной пропускной способностью каналов могут запрашивать сегменты, которых уже нет в плейлисте и на диске. В связи с этим, в сборке 5.2.581 добавлена возможность хранить некоторое число сегментов на диске после их ухода из плейлиста. Эта возможность включается настройкой

```
hls_hold_segments_before_delete=true
```

По умолчанию, будет храниться 5 последних сегментов

```
hls_hold_segments_size=5
```

Например, если плейлист содержит 3 сегмента

```
#EXTM3U
#EXT-X-VERSION:8
#EXT-X-TARGETDURATION:11
#EXT-X-MEDIA-SEQUENCE:15
#EXT-X-DISCONTINUITY-SEQUENCE:1
#EXTINF:3.415,
test_017.ts
#EXTINF:10.417,
test_018.ts
#EXTINF:9.084,
test_019.ts
```

на диске будут храниться 3 текущих сегмента из плейлиста и 5 предшествующих

```
test_012.ts
test_013.ts
test_014.ts
test_015.ts
test_016.ts
test_017.ts
test_018.ts
test_018.ts
```

Использование оперативной памяти

При больших нагрузках на сервер, например, если он выделен для раздачи потоков по HLS, чтение сегментов с диска для отправки подписчикам может давать задержки. В этом случае необходимо включить хранение HLS сегментов в памяти

```
hls_store_segment_in_memory=true
```

Для отправки подписчикам сегменты будут считываться из памяти. Необходимо отметить, что в этом случае потребуется больше памяти под Java heap для хранения сегментов.

В сборке 5.2.1713 хранение сегментов в оперативной памяти включено по умолчанию.

Отладочные логи для HLS-сессии

Для отчета об ошибке можно, используя CLI, включить сбор отладочных логов для HLS-сессий

```
update node-setting --value true hls_enable_session_debug
```

Следует учесть, что файл настроек flashphoner.properties будет перезаписан после этой команды.

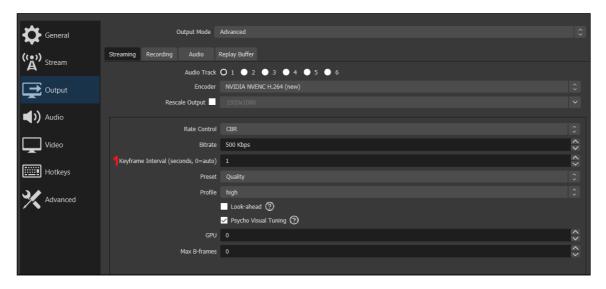
Поддержка Low Latency HLS

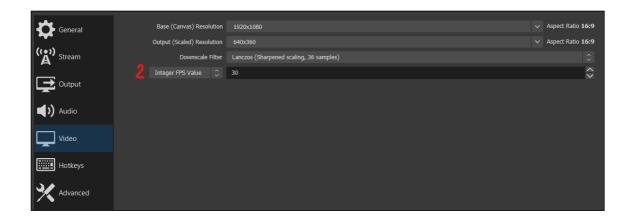
Начиная со сборки 5.2.1181, поддерживается Low Latency HLS (LL HLS). Эта возможность включается при помощи настройки

```
hls_ll_enabled=true
hls_new_http_stack=true
```

В этом случае плееры, которые поддерживают LL HLS (например, HLS.JS), будут играть дополнительные HLS сегменты и давать меньшую задержку по сравнению с плеерами, которые их не играют (например, VideoJS).

Для того, чтобы LL HLS проигрывался корректно, необходимо, как и для обычного HLS, обеспечить стабильный FPS публикуемого потока и стабильный интервал между ключевыми кадрами. Таким образом, исходный поток рекомендуется публиковать как RTMP с параметрами





Рекомендованные настройки сервера для LL HLS

Начиная со сборки 5.2.1345, рекомендуются следующие настройки для проигрывания Low latency HLS:

```
hls_ll_enabled=true
hls_auto_start=true
hls_preloader_enabled=false
hls_player_width=848
hls_player_height=480
video_filter_enable_fps=true
video_filter_fps=30
video_encoder_h264_gop=60
```

Начиная со сборки 5.2.1965, для работы LL HLS по сравнению с настройками по умолчанию достаточно настройки

hls_ll_enabled=true

Использование HTTP/2 и HTTP/1

Согласно спецификации, LL HLS должен воспроизводиться с использованием HTTP/2, то есть через безопасное соединение

https://wsc:8445/test/test.m3u8

В WCS также возможно использование HTTP/1 через небезопасное соединение

http://wsc:8082/test/test.m3u8

Отметим, что LL HLS через HTTP/1 работает во всех основных браузерах, кроме Safari, поэтому не рекомендуется использовать эту возможность в промышленной эксплуатации.

Каталог для нарезки сегментов LL HLS

По умолчанию, сегменты LL HLS помещаются в подкаталоги с именами потоков в каталог

11_hls_dir=/usr/local/FlashphonerWebCallServer/11-hls

При изменении местоположения, например

11_hls_dir=/opt/ll-hls

необходимо назначить права доступа командой

/usr/local/FlashphonerWebCallServer/bin/webcallserver set-permissions

и перезапустить WCS, чтобы применить изменения.

Прелоадер для LL HLS

Начиная со сборки 5.2.1729, для LL HLS, как и для обычного HLS, может использоваться прелоадер

hls_preloader_enabled=true

Файлы прелоадера LL HLS по умолчанию помещаются в каталог

ll_hls_preloader_dir=/usr/local/FlashphonerWebCallServer/ll-hls/.preloader

Расположение может быть изменено, например

11_hls_preloader_dir=/opt/preloader

По умолчанию, LL HLS прелоадер состоит из следующих файлов, по одному на каждое соотношение сторон видео потоков

16x9.mp4 2x1.mp4

4x3.mp4

Если соотношение сторон публикуемого потока неизвестно, используется прелоадер с соотношением 16:9. Если файлы прелоадера отсутствуют, нарезка LL HLS будет начинаться без прелоадера, аналогично настройке

hls_preloader_enabled=false

Настройка собственного прелоадера

При необходимости, для LL HLS может быть настроен собственный прелоадер. Для этого необходимо подготовить файлы в трех основных аспектах 16:9, 4:3 и 2:1 согласно следующим требованиям:

- контейнер МР4, кодек видео Н264, кодек аудио ААС
- файлы должны позволять немедленное проигрывание (MP4 атом moov должен быть перед атомом mdat)
- файлы не должны содержать В-фреймов
- длительность файла должна быть около 1 минуты
- файл должен иметь ровный FPS
- интервал между ключевыми кадрами должен быть около 2 секунд

Предполагается, что исходные файлы уже записаны в нужном аспекте, например, при помощи OBS Studio или подготовлены в видеоредакторе. Пример команды для преобразования файла под указанные требования:

```
ffmpeg -i 16x9-source.mp4 -bf 0 -acodec aac -vcodec h264 -preset ultrafast -g 60 -strict -2 -r 30 -ar 48000 -movflags faststart -ss 00:00:00 -t 00:01:00 16x9.mp4
```

Затем подготовленными файлами необходимо заменить файлы прелоадера по умолчанию и перезапустить WCS.

Для восстановления прелоадера по умолчанию достаточно удалить файлы собственного прелоадера и перезапустить WCS.

Общие настройки для LL и не LL HLS

Начиная со сборки 5.2.1965, все настройки с префиксом hls_ применяются одинаково к LL и не LL HLS.

Следующие настройки применяются только к LL HLS и не применяются к не LL HLS

Параметр	Описание	Значение по умолчани ю
Il_hls_max_number_of_pa rent_segments_containin g_partials	Максимальное число ч астичных сегментов дл я одного основного сег мента	5
II_hls_part_hold_back_co unt	Значение атрибута PART -HOLD-BACK в плейлист е	6
II_hls_partial_time_max	Максимальный размер	400

одного частичного сегм ента в миллисекундах

Поддержка m4s контейнера

В сборке 5.2.1626 добавлена поддержка m4s контейнера для нарезки HLS сегментов, а в сборке 5.2.1632 поддержка данного контейнера включена и для HLS ABR

11_hls_fragmented_mp4=true

Начиная со сборки 5.2.1724, контейнер m4s поддерживается и для HLS ABR в CDN.

При необходимости, можно переключиться на использование ts контейнера

11_hls_fragmented_mp4=false

Использование общего сетевого стека для обычного и Low Latency HLS

В сборке 5.2.1749 добавлена настройка, разрешающая использование унифицированного сетевого стека для обычного HLS и Low latency HLS. Эта настройка включена по умолчанию:

use_new_hls=true

При этом:

- по умолчанию используется контейнер m4s для записи сегментов
- настройки с префиксом hls применяются и к обычному HLS, и к LL HLS
- настройки с префиксом 11_hls применяются к LL HLS и к контейнеру m4s

Warning

Начиная со сборки 5.2.1793, данная настройка удалена. Для доставки HLS и LL HLS сегментов используется унифицированный сетевой стек.

Настройка URL манифеста

Начиная со сборки 5.2.1852, можно задать шаблоны URL, по которому должен запрашиваться основной плейлист (манифест) потока. По умолчанию используются следующие шаблоны:

 $\label{lem:label} $$hls_path_template=\{streamName\}/\{streamName\}.m3u8$$ $hls_abr_path_template=\{streamName\}_{abrSuffix}/_{streamName}_{abrSuffix}.m3u8$$ $hls_abr_path_template=\{streamName}_{abrSuffix}.m3u8$$ $hls_abr_path_template$

Здесь:

- streamName имя потока, опубликованного на сервере
- abrSuffix суффикс для ABR потока, заданный настройкой hls_abr_stream_name_suffix

В этом случае для получения манифеста HLS потока используется URL

https://wcs:8445/stream/stream.m3u8

а для HLS ABR потока

https://wcs:8445/stream-HLS-ABR-STREAM/stream-HLS-ABR-STREAM.m3u8

Если необходимо, например, задать фиксированное имя манифеста и при этом различать ABR и не ABR потоки, можно задать следующие шаблоны

hls_path_template={streamName}/playlist.m3u8
hls_abr_path_template={streamName}/playlist{abrSuffix}.m3u8

В этом случае для получения манифеста HLS потока будет использоваться URL

https://wcs:8445/stream/playlist.m3u8

а для HLS ABR потока

https://wcs:8445/stream/playlist-HLS-ABR-STREAM.m3u8

Остановка HLS провайдера при остановке публикации

В сборке 5.2.1920 добавлена настройка, управляющая остановкой HLS провайдера при остановке публикации исходного потока. По умолчанию в этом случае HLS провайдер останавливается с задержкой

hls_delayed_shutdown=true

Задержка зависит от размера плейлиста

hls_list_size=8

Таким образом, при остановке исходного потока тем подписчикам, которые уже его играют, дается возможность доиграть уже нарезанные сегменты.

При необходимости, задержку можно отключить

hls_delayed_shutdown=false

В этом случае HLS провайдер будет остановлен стразу при остановке исходной публикации.

Управление нарезкой HLS сегментов

По умолчанию сегменты нарезаются либо по получению очередного ключевого кадра, либо по достижении сегментом размера, заданного настройкой (по умолчанию 2000 мс)

hls_time_min=2000

В сборке 5.2.1974 добавлена возможность включить нарезку только по получению ключевых кадров

hls_always_start_segment_with_key_frame=true

При этом размер сегмента может превышать hls_time_min, однако минимальный размер сегмента может быть и меньше этого значения. Для того, чтобы ограничить минимальную длительность сегментов, используется настройка

hls_keep_min_segment_duration=true

В этом случае размер сегмента будет не меньше, чем hls_time_min.

Известные проблемы

1. Невосстанавливаемый фриз HLS потока при воспроизведении в iOS Safari через CDN



🤴 Симптомы

Через одну минуту после начала публикации изображение останавливается, звук продолжает воспроизводиться



а) включить транскодинг на сервере при помощи настройки в файле flashphoner.properties

disable_streaming_proxy=true

b) если включение транскодинга нежелательно, установить в файле flashphoner.properties

hls_discontinuity_enabled=true

В этом случае возможны щелчки в аудиопотоке, но изображение останавливаться не будет.

2. Прекращение записи сегментов HLS при воспроизведении потока, опубликованного в браузере Firefox



🚺 Симптомы

Через несколько минут после начала воспроизведения прекращается запись HLSсегментов, при этом директория потока в директории hls не удаляется, в логе сервера продолжают появляться сообщения

INFO HLSStreamManager - HLSStreamProviderKeepaliveThread-80 Remove hls channel

Для восстановления публикующая сторона должна заново опубликовать поток.



Решение

Использовать другой браузер для публикации потока, который предполагается воспроизводить по HLS

3. При воспроизведении HLS в Safari в iOS 12.4 не играет видео для первого подписчика



Симптомы

При подключении первого подписчика к потоку по HLS в Safari в iOS 12.4 видео не воспроизводится, если HLS-подписчики уже есть, видео играет нормально



Установить минимальное количество сегментов в плейлисте HLS не менее 2

4. При воспроизведении RTMP-потока как HLS в Safari в iOS 12.4 не играет видео для любого подписчика, если активна настройка

HLS может не играть в iOS Safari 12.4 даже с настройкой

hls_auto_start=true



Симптомы

При подключении подписчиков к RTMP потоку по HLS в Safari в iOS 12.4 видео не воспроизводится

Решение

Gри публикации файла со звуковой дорожкой стерео, использовать моно звук, например, для ffmpeg указать настройку

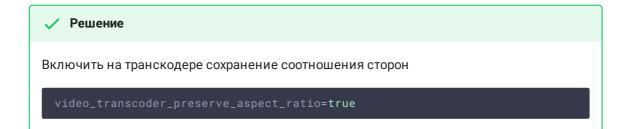
-acodec aac -ac 1

5. Если по HLS проигрывается поток, транскодированный в CDN, и если при этом изменяется соотношение сторон потока, HLS прелоадер отображается в соответствии с соотношением сторон оригинального потока



🥶 Симптомы

При заказе транскодированного потока с указанием профиля в имени, например test-640х480р, отображается прелоадер 16:9, если исходный поток опубликован в разрешении 1280х720



6. Если в исходном потоке содержатся В-фреймы, в некоторых плеерах могут наблюдаться подергивания



Сильно дергается картинка при проигрывании HLS, возможен эффект низкого FPS

Решение

Обновить WCS до сборки 5.2.863, в которой решена данная проблема

7. При проигрывании LL HLS в браузере Safari, при первом подключении к потоку может пропадать звук



Поток играет по LL HLS, но нет звука



Обновить WCS до сборки 5.2.1345 или новее, где эта проблема решена

8. Браузер Chrome на Ubuntu 22.04 может давать ошибку CORS при загрузке плейлистов по HTTPS



Браузер Chrome на Ubuntu 22.04 играет HLS по HTTPS нормально, затем выдает ошибку CORS при загрузке очередного плейлиста



Не отправлять из браузера Chrome HTTP запросы на тот же сайт, с которого проигрывается HLS по HTTPS

9. При проигрывании LL HLS ABR в iOS Safari 16 второй и последующие подписчики могут играть поток с большой задержкой.



🥶 Симптомы

Первый iOS Safari 16 подписчик подключается к потоку и играет нормально, последующие подписчики играют с большой задержкой относительно публикации



Решение

Обновить WCS до сборки 5.2.1677, чтобы использовать контейнер m4s по умолчанию для LL HLS, и ожидать возможного исправления в iOS Safari 17

10. HLS ABR не играет при использовании контейнера m4s



🥶 Симптомы

Нарезка HLS ABR не стартует, в серверном логе сообщение

02:18:01,957 ERROR HlsAbrStreamProvider - HLS-HTTPS-pool-5-thread-1 Failed to check stream null java.lang.NullPointerException



Обновить WCS до сборки 5.2.1677, где эта проблема решена

11. VLC требует, чтобы LL HLS манифест включал как минимум 4-6 сегментов при подключении первого подписчика



VLC играет LL HLS в контейнере m4s с рассинхронизацией звука и видео, либо дает фризы при переключении качества для LL HLS ABR

✓ Решение

Обновить WCS до сборки 5.2.1677 и увеличить минимальный размер манифеста

hls_min_list_size=6

12. HLS поток только с аудио в контейнере ts проигрывается с заметными щелчками в браузере Safari

6 Симптомы

HLS поток только с аудио щелкает при проигрывании с использованием native HTML5 плеера в браузере Safari

Решение

Начиная со сборки 5.2.1690, использовать контейнер m4s для потоков только с аудио

13. При использовании HLS ABR под нагрузкой может возникать утечка ресурсов кодирования

© Симптомы

При использовании HLS ABR, при большой нагрузке на CPU сервера (например, общее число профилей кодирования для всех опубликованных потоков превышает возможности CPU), после остановки публикаций ресурсы кодирования могут не освобождаться, это видно в статистике, например

streams_hls=0
...
native_resources.video_encoders=5

Решение

Обновить WCS до сборки 5.2.1947 и установить следующий параметр

handler_async_disconnect=false