

Building an Open-Source based audio streaming platform



Maxime Bugeia

2 February 2020

We stream audio!



radiofrance

French public broadcasting service

radiofrance



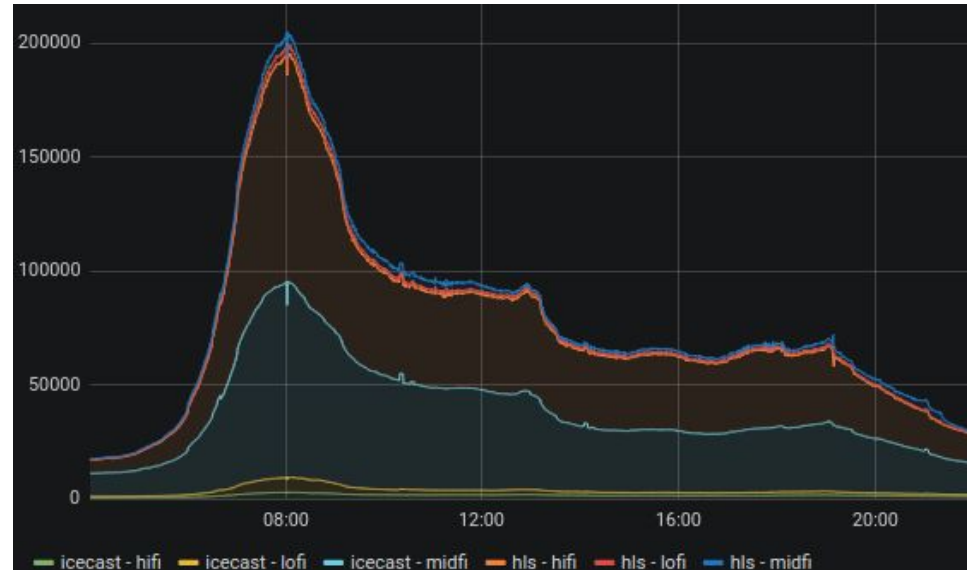
6 national channels

47 local channels

23 music webradios

2 Million listeners per day

200k simultaneous listeners

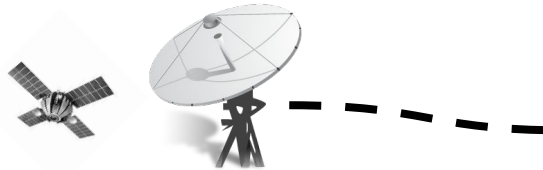


Before



Black Box
3rd party

Icecast



After



Our own cloud and
open source based
infrastructure

Icecast

HLS



Audio streaming

ICECAST

Streaming server

Master/Relay Architecture

Single bitrate

No cache

Long persistent TCP
connections

HLS

Segmented audio/video files

Sliding Playlist (m3u8)

Adaptive bitrate

Better mobile experience

Cacheable content

Just “static” files

Audio to listeners



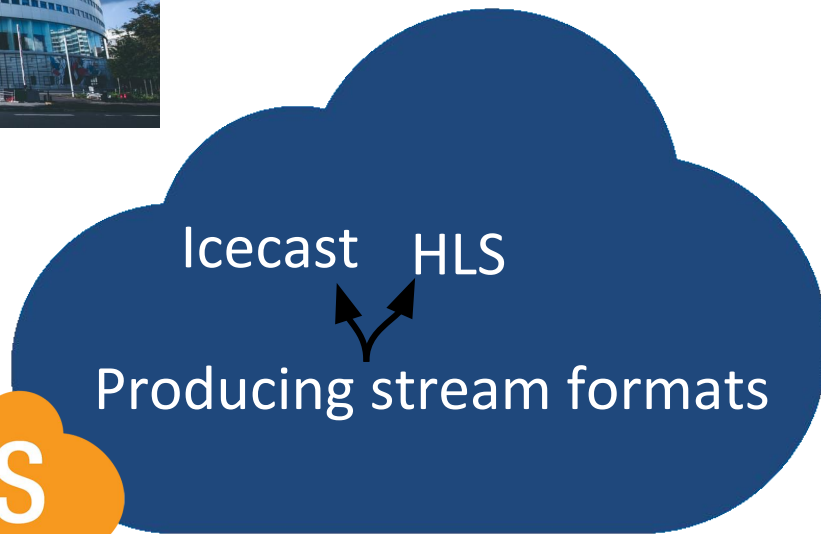
Getting
audio to
the cloud



Audio to listeners



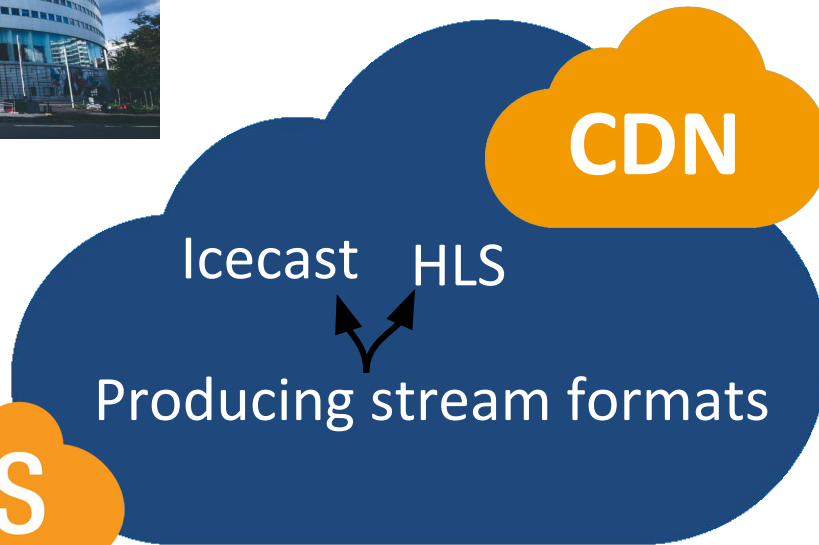
Getting
audio to
the cloud



Audio to listeners



Getting
audio to
the cloud



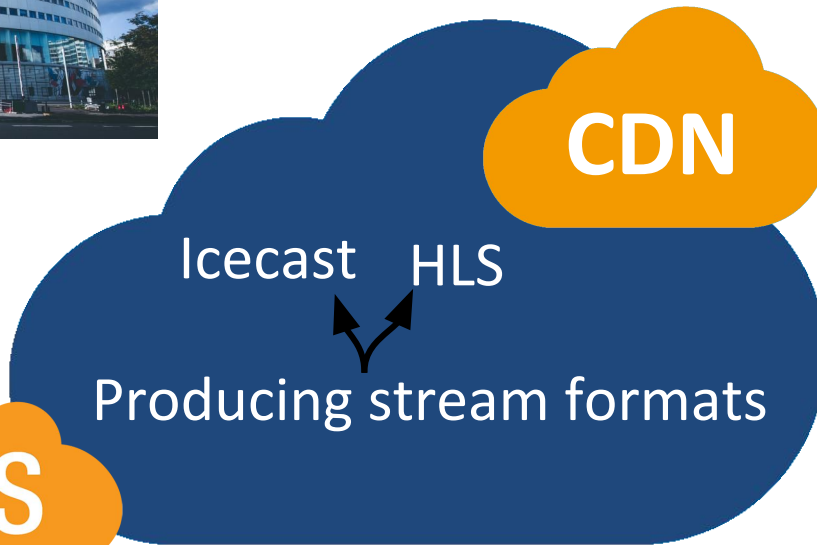
Deliver content
to listeners



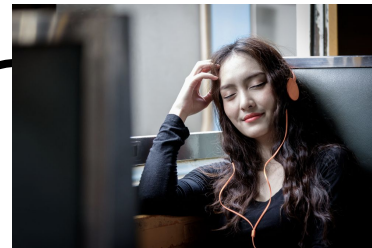
Audio to listeners



Getting
audio to
the cloud



Deliver content
to listeners

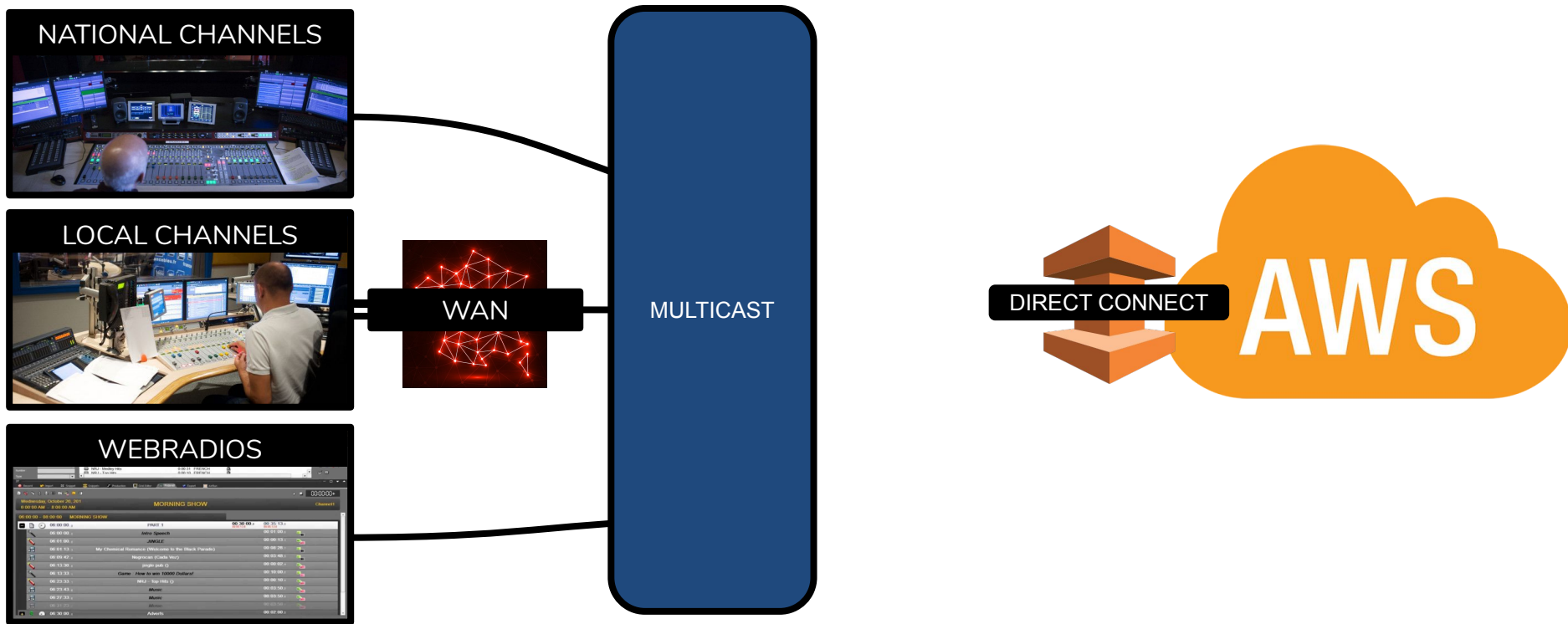


Operate the platform

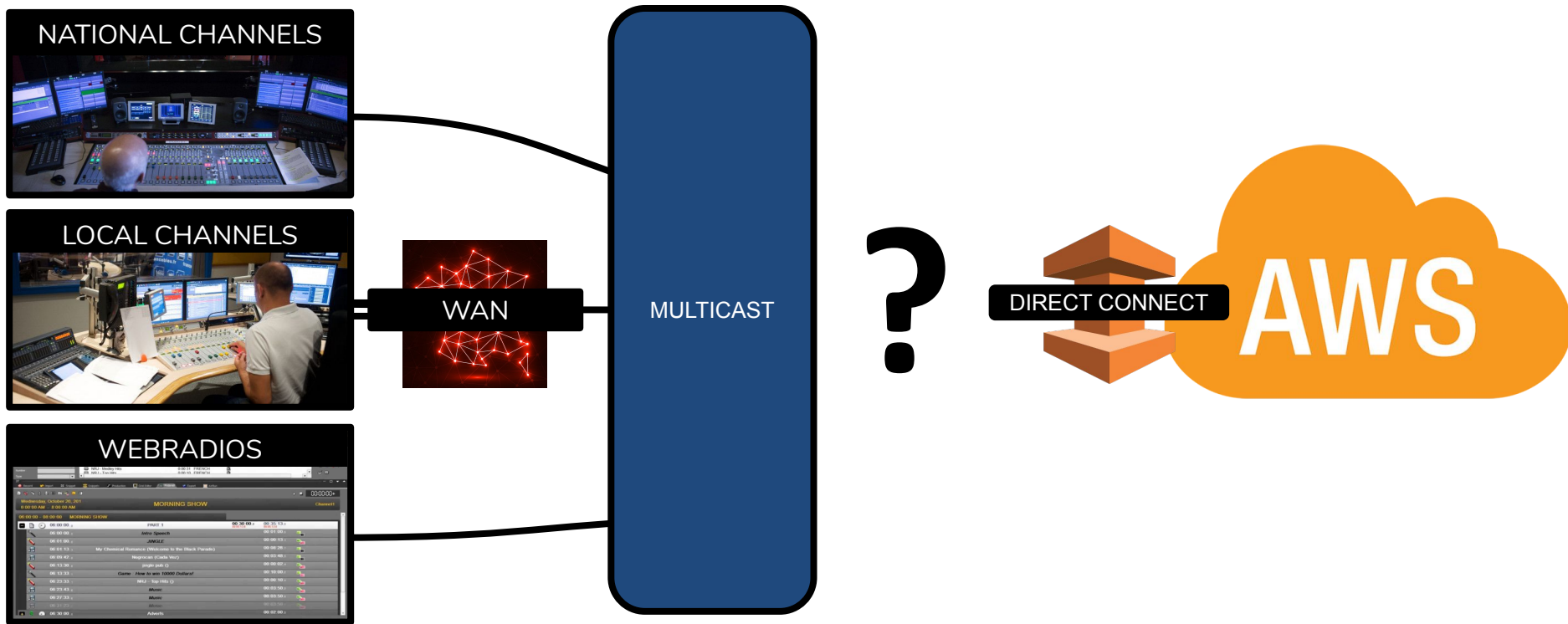
Getting audio to the cloud!



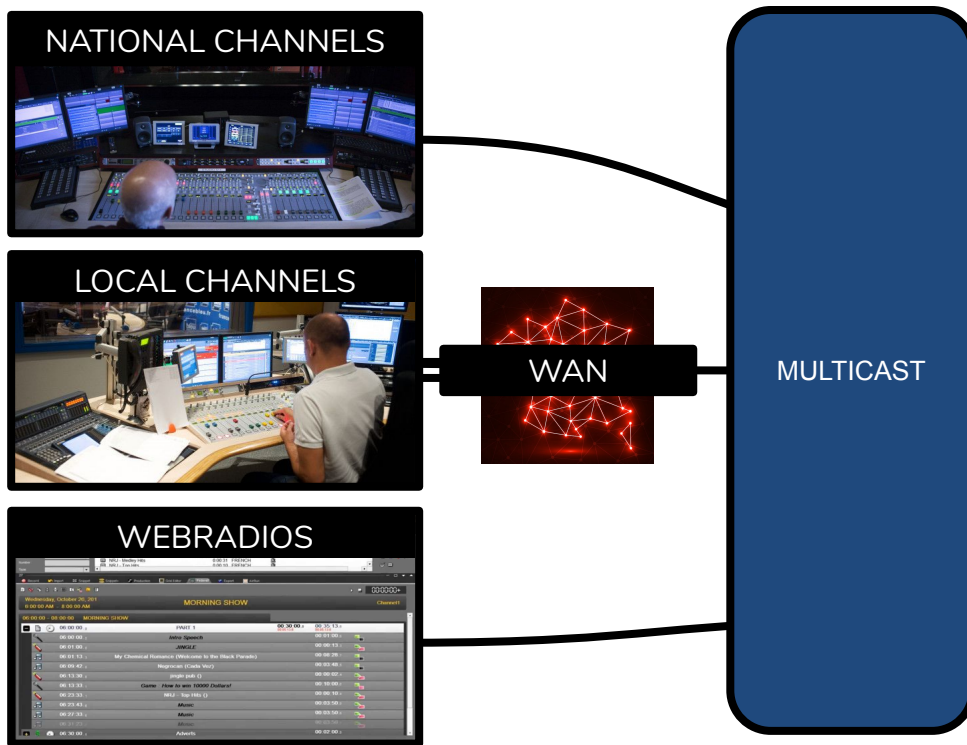
Transport: multicast problem



Transport: multicast problem



Transport: multicast problem



?



No multicast in AWS!

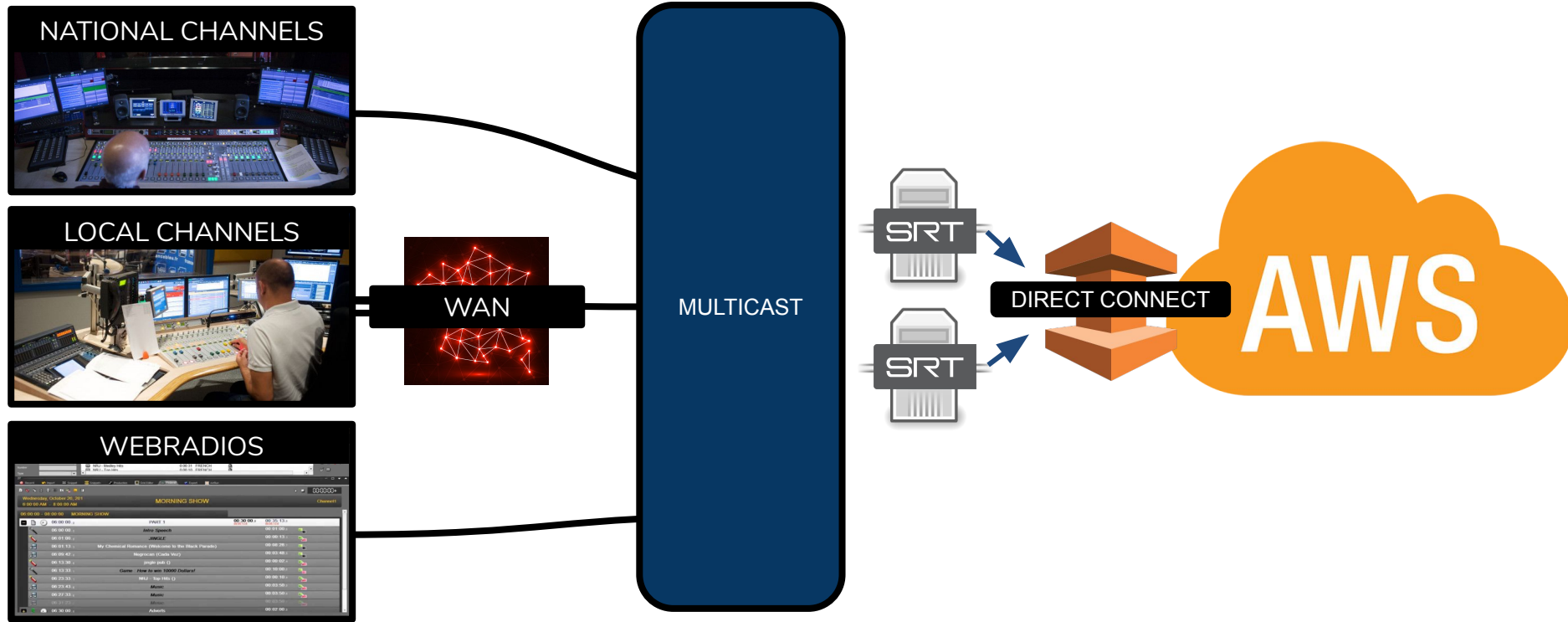
Transport: SRT to the rescue

S R T

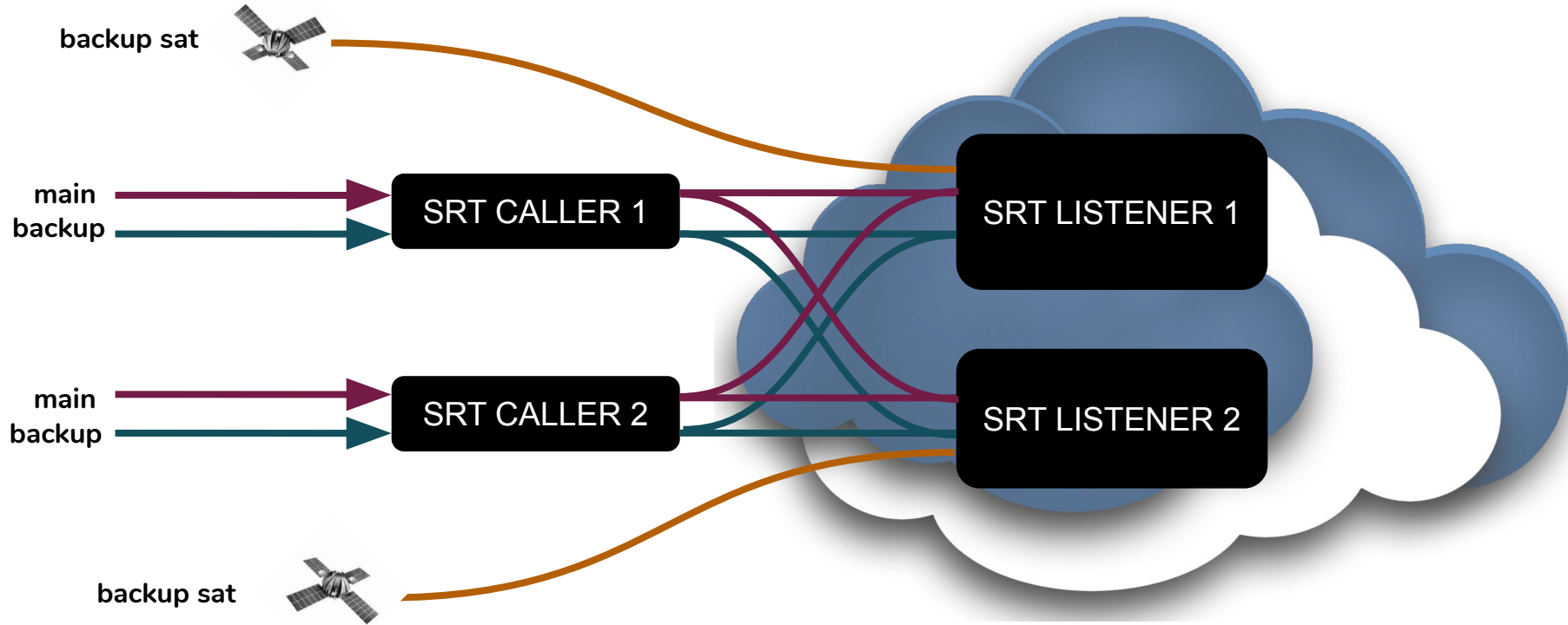


Secure | **R**eliable | **T**ransport

Transport: to the cloud!

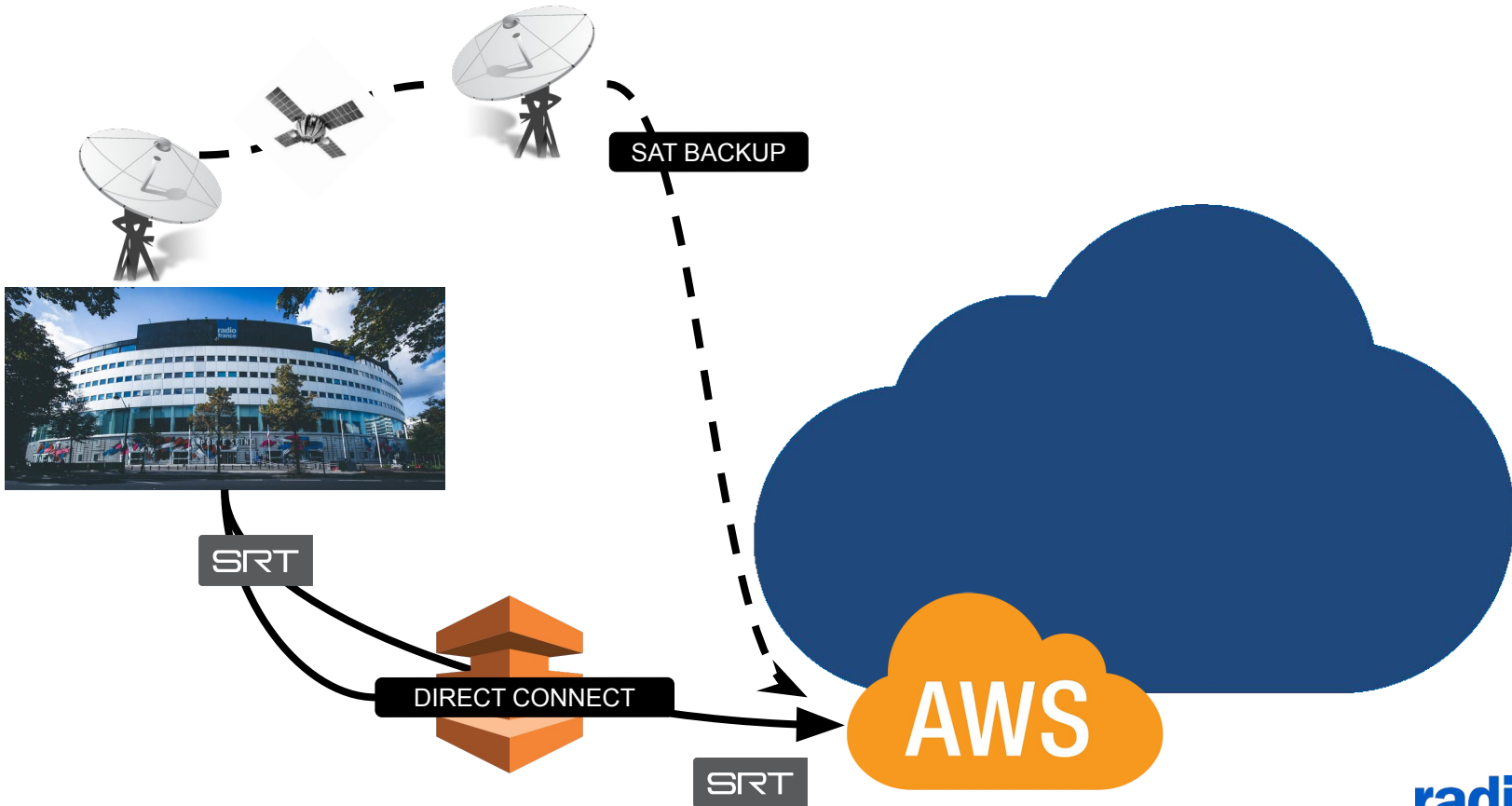


Transport: resiliency



5 inputs per channel

Audio to cloud



Producing stream formats!



One program to rule them all

Liquidsoap



Open Source Audio & Video Streaming Language

Collaboration with Radio France

Receive

SRT listener

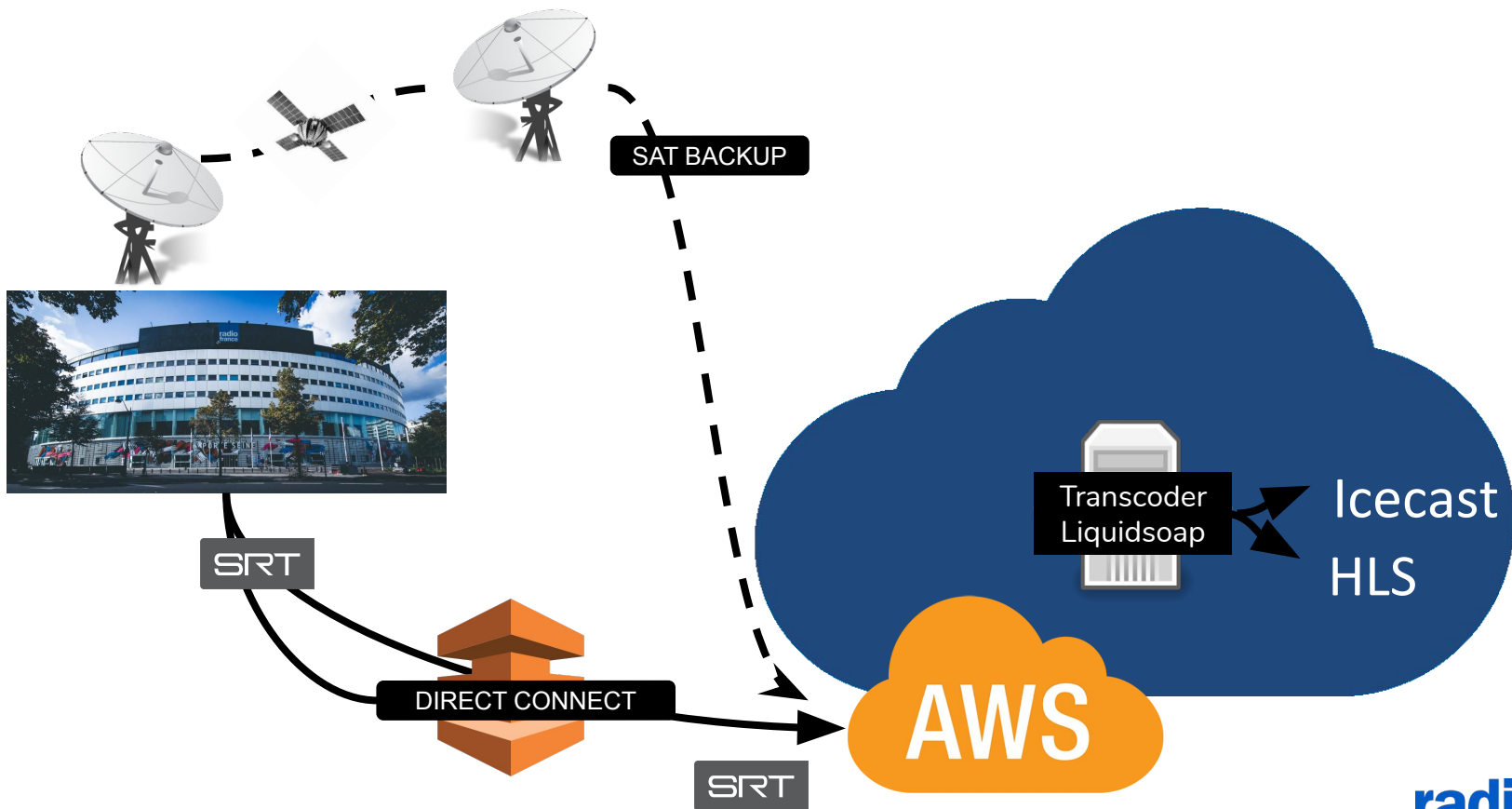
Transcode

Output AAC
Output MP3
Produce HLS
Produce Icecast

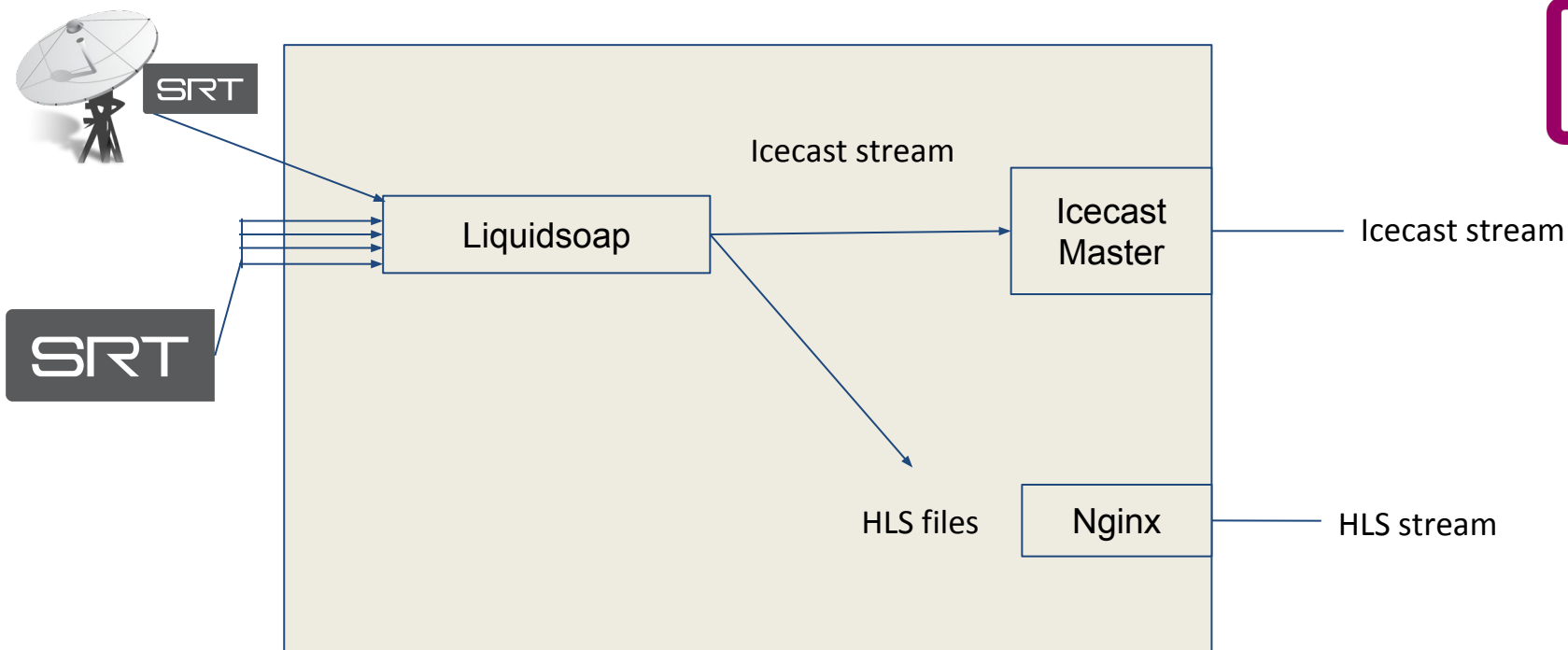
Control

Switch between sources
Fallback logic
Expose metrics

Producing the streams



Transcoder Architecture



Radio as Code



Define inputs

```
main_caller1 = buffer(fallible=true, input.srt(port=10000))
main_caller2 = buffer(fallible=true, input.srt(port=10001))
backup_caller1 = buffer(fallible=true, input.srt(port=10002))
backup_caller2 = buffer(fallible=true, input.srt(port=10003))
override_caller1 = buffer(fallible=true, input.srt(port=10004))
override_caller2 = buffer(fallible=true, input.srt(port=10005))
sat_sat1 = buffer(fallible=true, input.srt(port=10006))
safe_blank = blank()
```

Radio as Code



Live: what we want to play

```
live = switch(  
  [ (is_playing("main_caller1"),main_caller1),  
    (is_playing("main_caller2"),main_caller2),  
    (is_playing("backup_caller1"),backup_caller1),  
    (is_playing("backup_caller2"),backup_caller2),  
    (is_playing("override_caller1"),override_caller1),  
    (is_playing("override_caller2"),override_caller2),  
    (is_playing("sat_sat1"),sat_sat1) ] )
```



Fallback logic

```
radio_prod = fallback(  
    [ live,  
      main_caller1,  
      main_caller2,  
      backup_caller1,  
      backup_caller2,  
      sat_sat1,  
      safe_blank ])
```

Radio as Code



Fallback logic

```
radio_prod = fallback(  
    [ live,  
      main_caller1,  
      main_caller2,  
      backup_caller1,  
      backup_caller2,  
      sat_sat1,  
      safe_blank ])
```

Never fail!



Output HLS

```
output.file.hls(on_file_change=on_file_change,  
                playlist="fip.m3u8",  
                segment_duration=2.0,  
                segment_name=segment_name,  
                "/var/hls/fip",  
                [("fip_aac_midfi", aac_midfi_mpegts),  
                ("fip_aac_hifi", aac_hifi_mpegts)],  
                radio_prod)
```



Source that never fails

Radio as Code



Output HLS

```
output.file.hls(on_file_change=on_file_change,  
                playlist="fip.m3u8",  
                segment_duration=2.0,  
                segment_name=segment_name,  
                "/var/hls/fip",  
                [("fip_aac_midfi", aac_midfi_mpegts),  
                ("fip_aac_hifi", aac_hifi_mpegts)],  
                radio_prod)
```

```
aac_midfi_mpegts = %ffmpeg(format="mpegts",  
                            ar=48000,  
                            b="96k",  
                            codec="libfdk_aac",  
                            profile="aac_low")  
  
aac_hifi_mpegts = %ffmpeg(format="mpegts",  
                           ar=48000,  
                           b="192k",  
                           codec="libfdk_aac",  
                           profile="aac_low")
```

Radio as Code

Output HLS

```
def on_file_change(~state, fname) =  
  if state == "closed" and file.extension(fname) != '.m3u8' then  
    upload_segment()  
  end
```

```
output.file.hls(on_file_change=on_file_change,  
  playlist="fip.m3u8",  
  segment_duration=2.0,  
  segment_name=segment_name,  
  "/var/hls/fip",  
  [("fip_aac_midfi", aac_midfi_mpegts),  
   ("fip_aac_hifi", aac_hifi_mpegts)],  
  radio_prod)
```

```
def segment_name(~position, ~extname, stream_name) =  
  timestamp = int_of_float(gettimeofday())  
  "#{stream_name}_2_#{position}_#{timestamp}.ts"  
end
```

Radio as Code



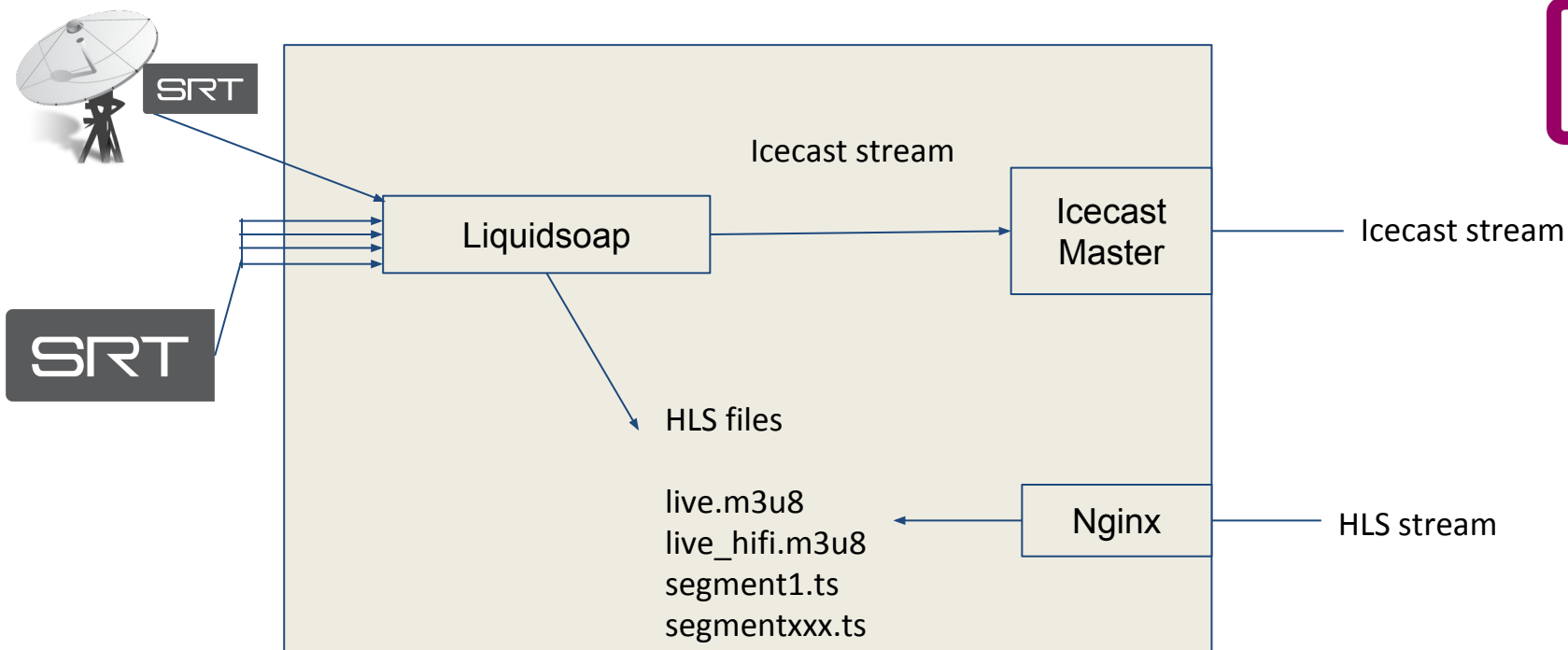
Output Icecast

```
output.icecast(fallible=true,  
               host="localhost",  
               port=8000,  
               password="icecastpassword",  
               mount="fip-hifi.aac",  
               aac_hifi,  
               radio_prod)
```

Source that never fails

```
aac_hifi = %fdkaac(channels=2,  
                   samplerate=48000,  
                   bandwidth="auto",  
                   bitrate=192,  
                   afterburner=true,  
                   aot="mpeg4_aac_lc",  
                   transmux="adts",  
                   sbr_mode=false)
```

Transcoders Architecture



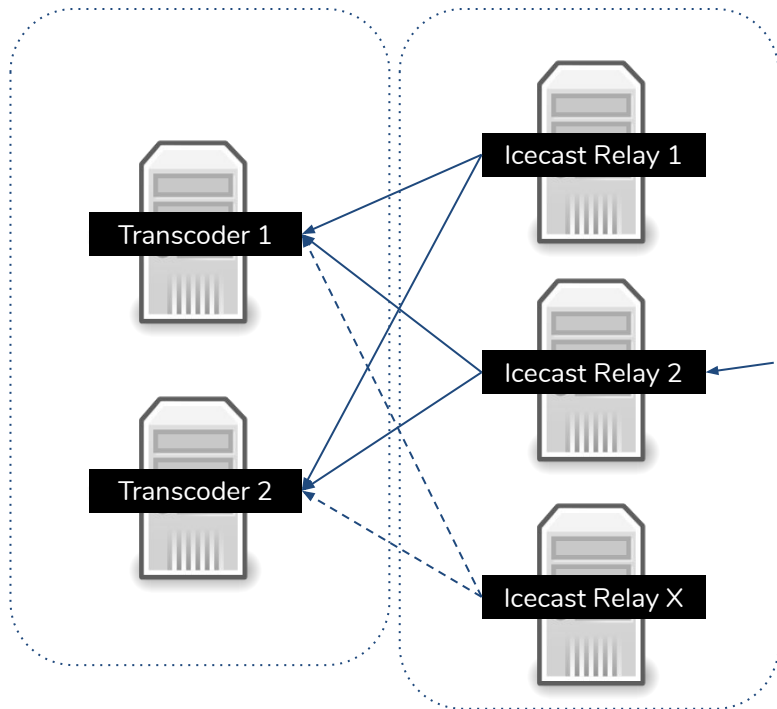
Scaling to 200k listeners!



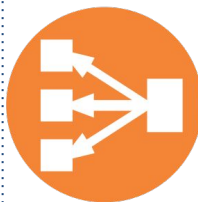
Scale: Icecast

Icecast Master

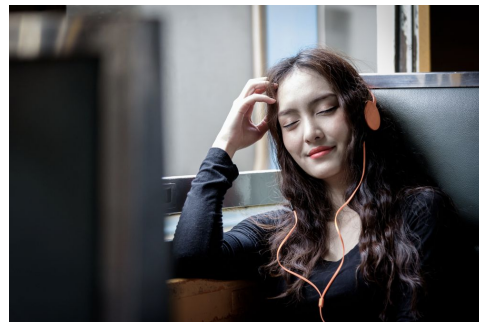
Icecast Relay



HTTP :80
HTTPS :443



AWS NLB

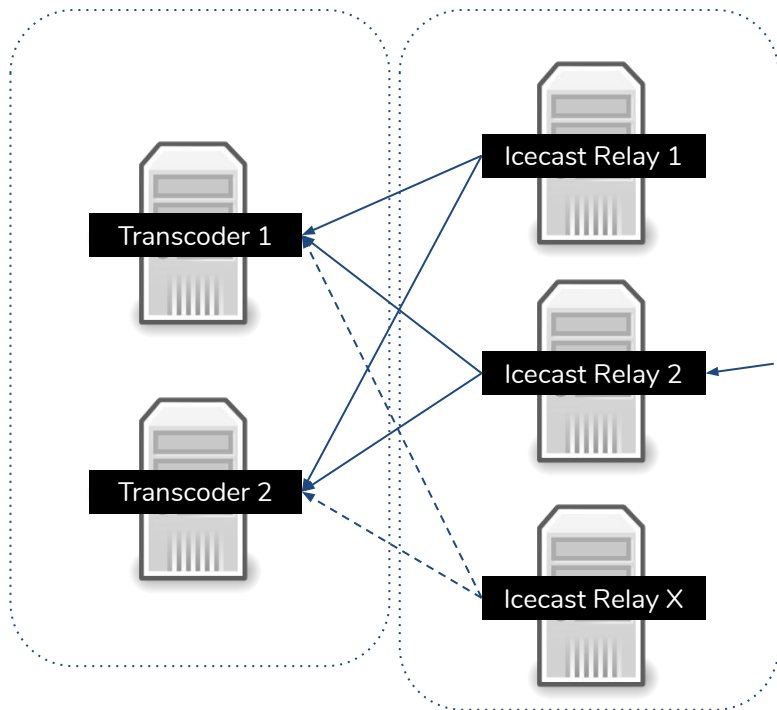


<https://icecast.radiofrance.fr/fip-hifi.aac>

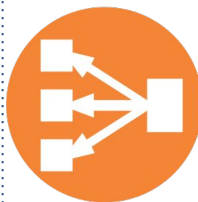
Scale: Icecast

Icecast Master

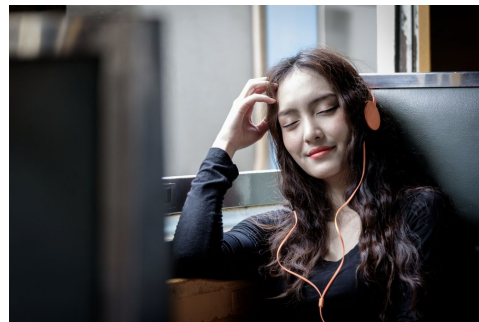
Icecast Relay



HTTP :80
HTTPS :443



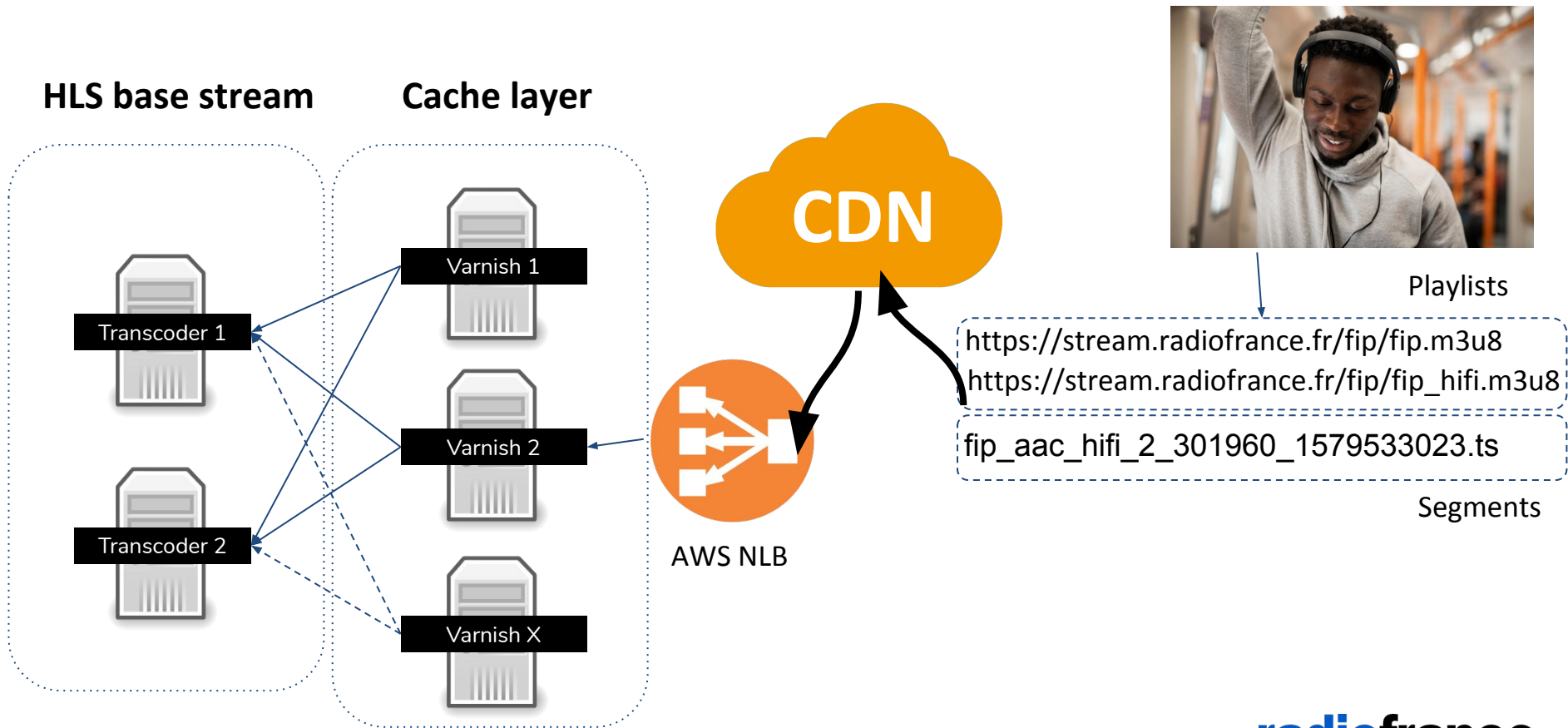
AWS NLB



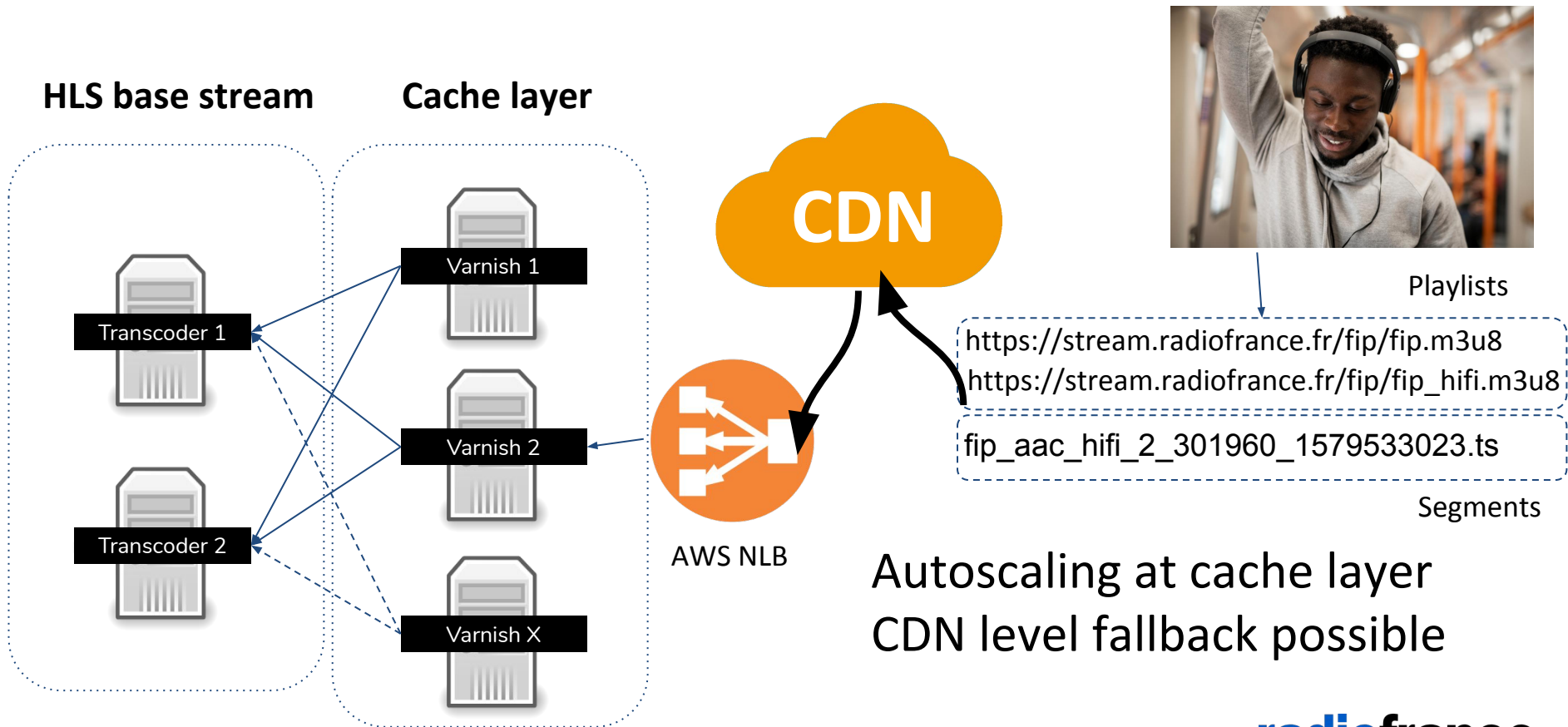
<https://icecast.radiofrance.fr/fip-hifi.aac>

No cache
No autoscaling
Icecast Relay automatic fallback

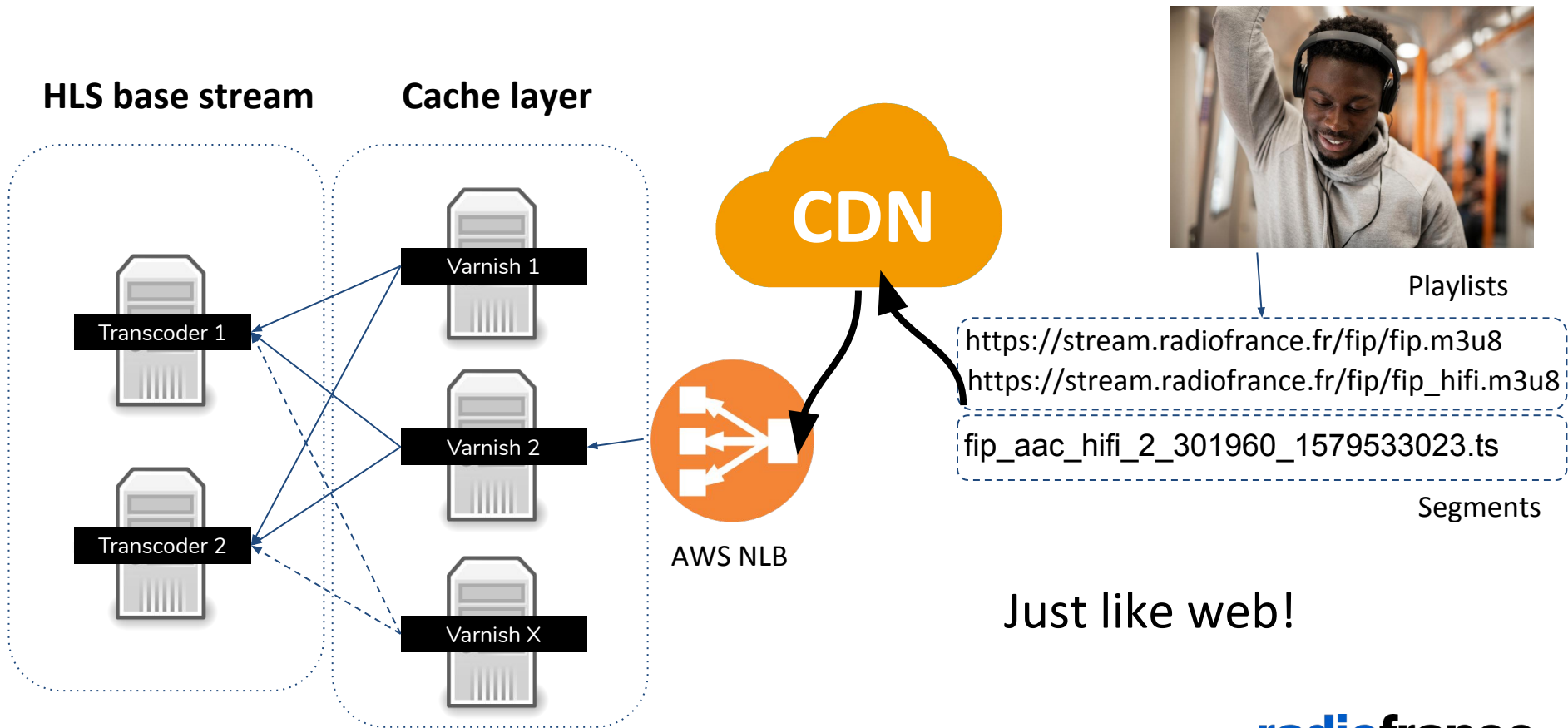
Scale: HLS



Scale: HLS



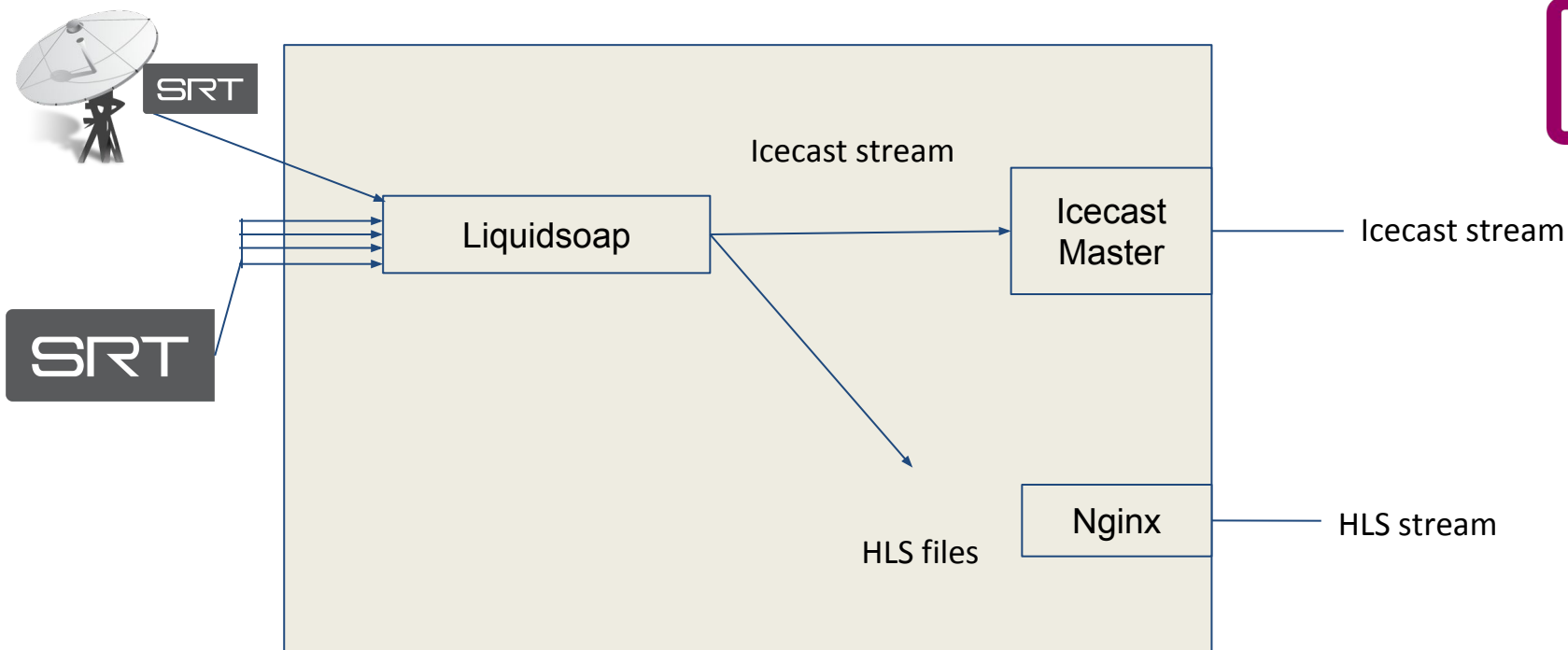
Scale: HLS



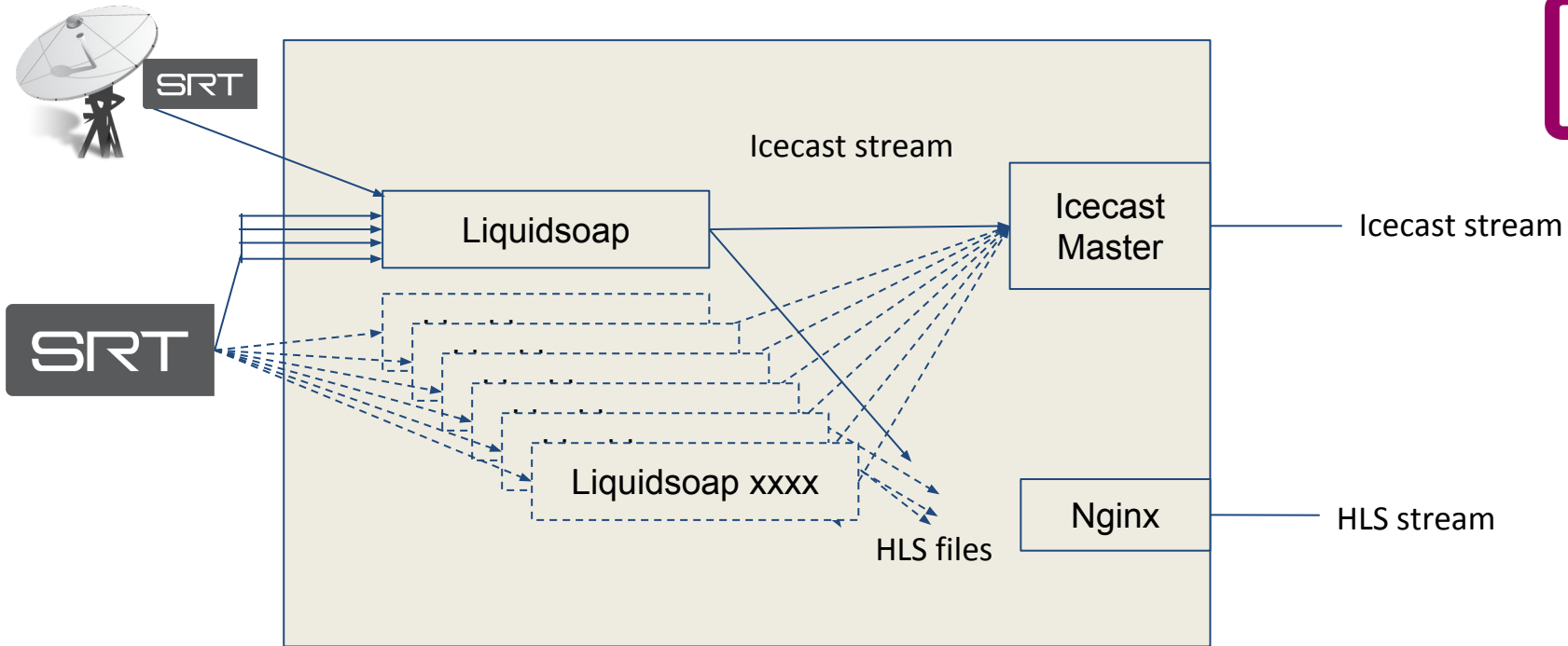
Operate



Transcoder Architecture



Transcoder Architecture



One Liquidsoap per channel


```

fipelectro:
  inputs:
    - name: main
      source: rtp://228.0.8.18:5000
      srtports:
        caller1: 10770
        caller2: 10771
    - name: backup
      source: rtp://228.0.8.18:5000
      srtports:
        caller1: 10772
        caller2: 10773
    - name: override
      isautofallback: false
      source: udp://228.0.4.110:2008
      srtports:
        caller1: 10774
        caller2: 10775
  outputs:
    hls:
      aac:
        - lofi
        - midfi
        - hifi
      icecast:
        aac:
          - hifi
        mp3:
          - midfi
    telnetserverport: 5077
    prometheusserverport: 6077
    httpserverport: 7077

```

Automate

Declarative everything

Generating:

- SRT callers
- Liquidsoap scripts
- Icecast master & relay conf



A N S I B L E

Monitor

Export metrics

Collect

Visualize

Node Exporter

Liquidsoap

Icecast exporter

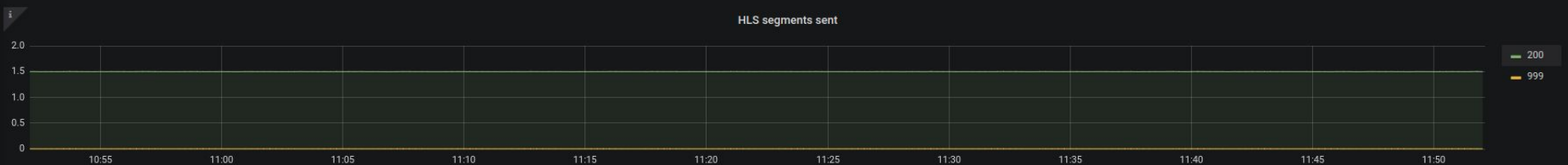
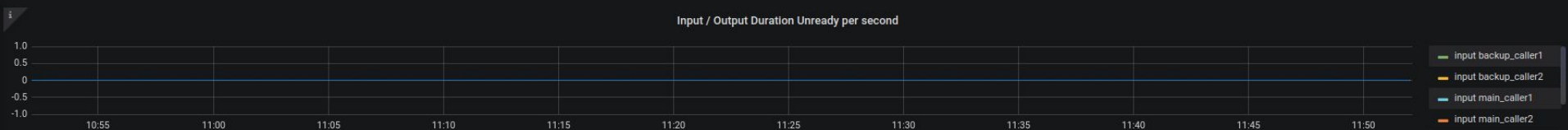
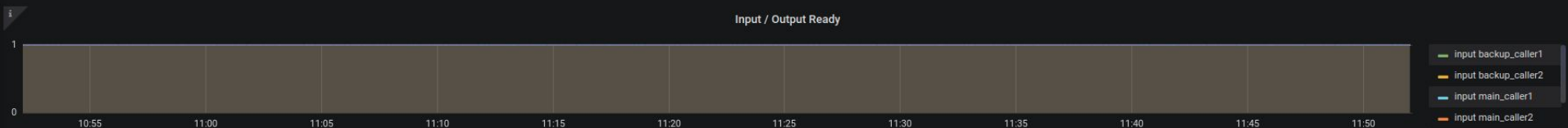


Prometheus

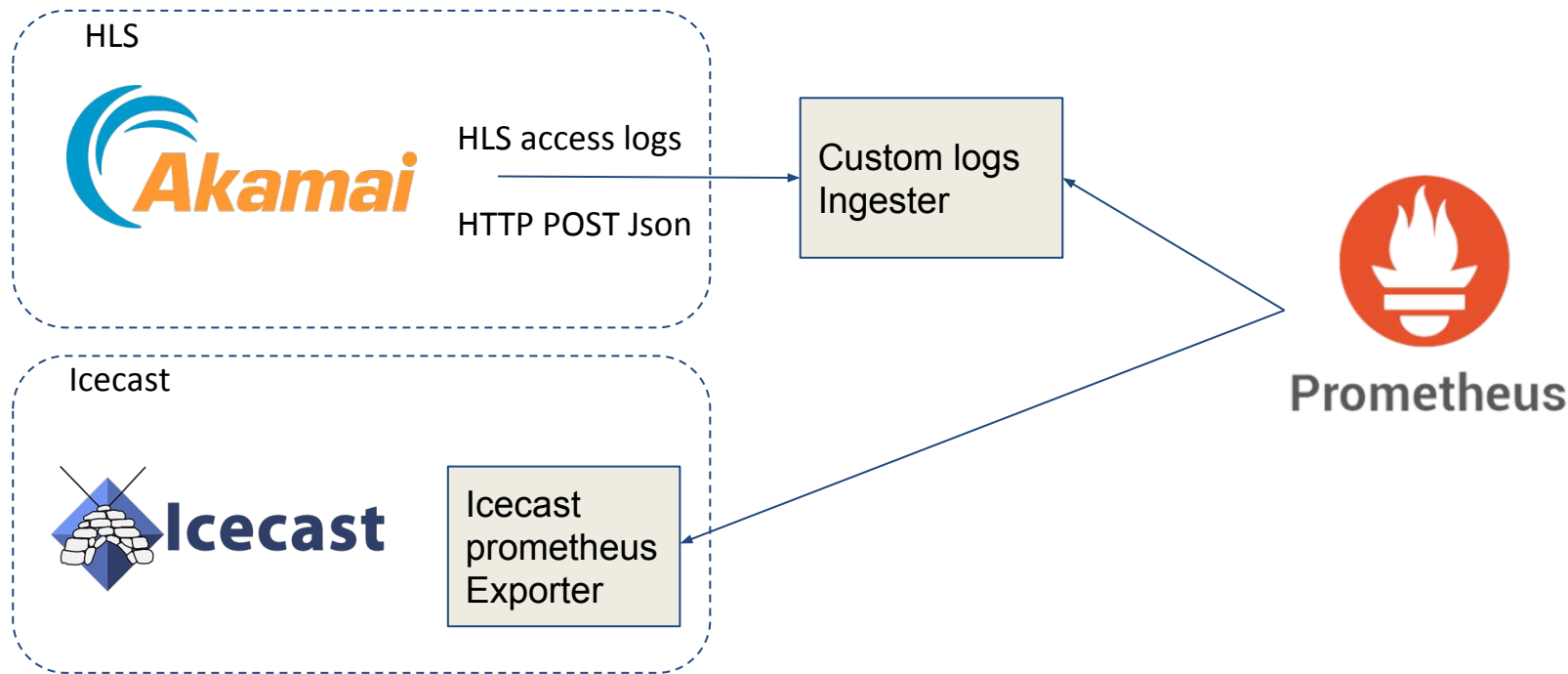


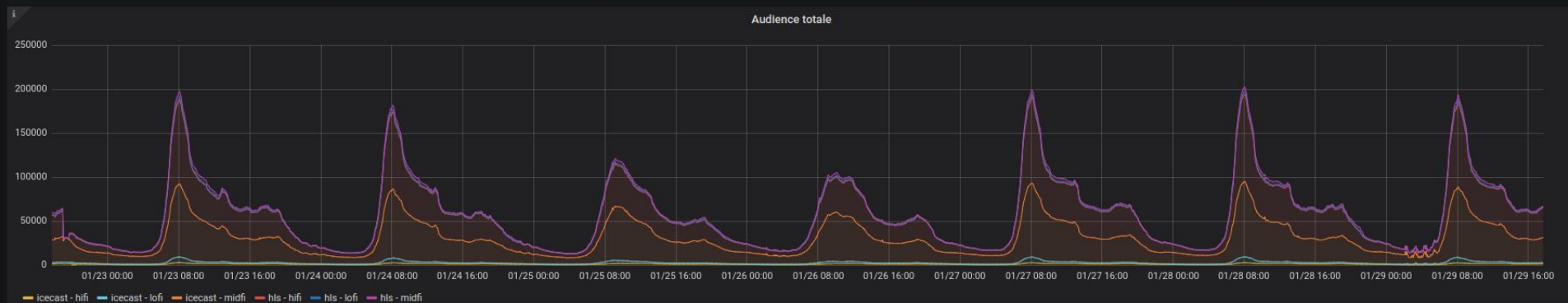
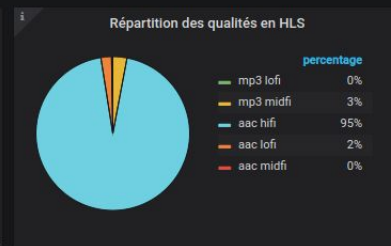
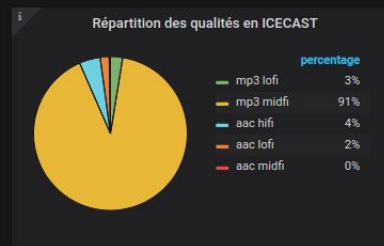
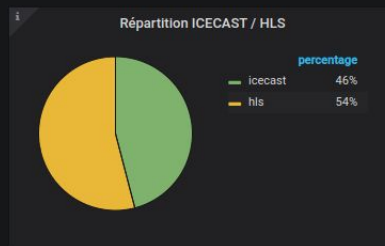
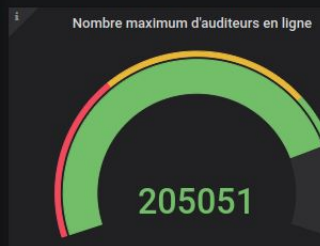
Grafana

Transcoder transcoder2 Radio franceinter Caller All

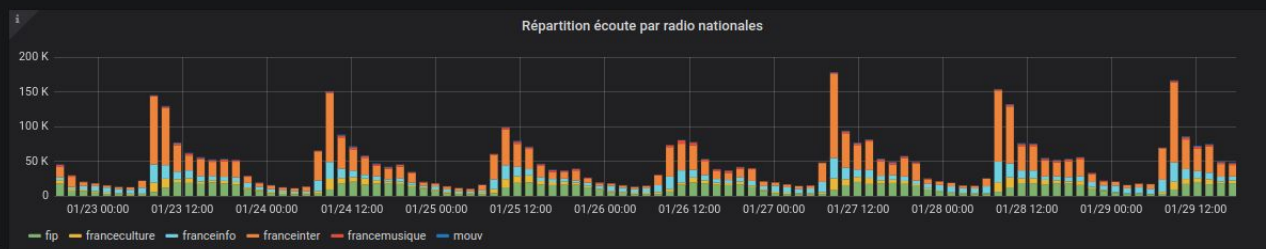


Monitor: real time listeners





Radios Nationales



Learn more

<https://github.com/mbugeia/srt2hls>

Receive SRT stream

Produce HLS

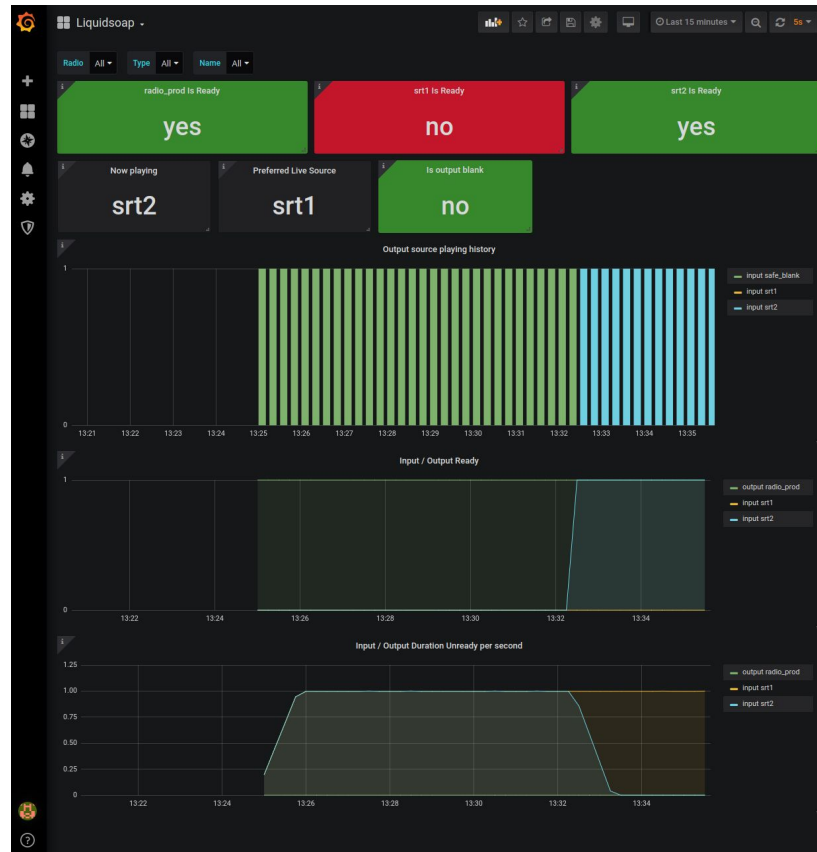
Prometheus Metrics

HTTP API

Command and fallback logic

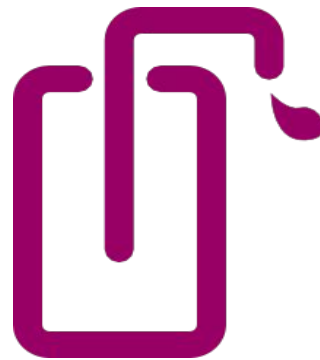
<https://github.com/Haivision/srt>

<https://www.liquidsoap.info/>



Learn more

Special thanks to the
Liquidsoap team!




<https://www.liquidsoap.info/>



Questions?

HLS / DASH Compatibility

Dynamic Adaptive Streaming over HTTP (MPEG-DASH) - OTHER

Usage % of all users  ?
Global 2.04%

HTTP-based media streaming communications protocol, an alternative to HTTP Live Streaming (HLS).

Current aligned		Usage relative		Date relative		Apply filters		Show all		?					
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser
		2-20													
		21-22													
6-10	12-18	23-71	4-78	3.1-12.1	10-63	3.2-13.1		2.1-4.4.4	12-12.1				4-9.2		
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1
		73-74	80-82	TP		13.3									

HTTP Live Streaming (HLS) - UNOFF

Usage % of all users  ?
Global 60.2%

HTTP-based media streaming communications protocol

Current aligned		Usage relative		Date relative		Apply filters		Show all		?					
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Balc Brow.
				3.1 - 5.1				2.1 - 2.3							
6 - 10	12 - 18	2 - 71	4 - 78	6 - 12.1	10 - 63	3.2 - 13.1		3 - 4.4.4	12 - 12.1				4 - 9.2		
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1
		73 - 74	80 - 82	TP		13.3									

mp3 / aac Compatibility

MP3 audio format - OTHER

Usage

% of all users

Global

96.74% + 0.13% = 96.87%

Popular lossy audio compression format

Current aligned Usage relative Date relative Apply filters Show all ?

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Balc Brow
		2-3													
6-8		3.5-21		3.1-3.2	10-12.1	3.2		2.1-2.2							
9-10	12-18	22-71	4-78	4-12.1	15-63	4-13.1		2.3-4.4.4	12-12.1				4-9.2		
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1
		73-74	80-82	TP		13.3									

AAC audio file format - OTHER

Usage

% of all users

Global

92.11% + 4.59% = 96.7%

Advanced Audio Coding format, designed to be the successor format to MP3, with generally better sound quality.

Current aligned Usage relative Date relative Apply filters Show all ?

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Balc Brow
			4-9												
6-8		2-21	10-11	3.1-3.2	10-12.1	3.2		2.1-2.3							
9-10	12-18	22-71	12-78	4-12.1	15-63	4-13.1		3-4.4.4	12-12.1				4-9.2		
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1
		73-74	80-82	TP		13.3									

ogg / flac Compatibility

Ogg Vorbis audio format - OTHER

Usage: % of all users
Global 78.52%

Vorbis is a free and open source audio format, most commonly used with the Ogg container.

Current aligned		Usage relative		Date relative		Apply filters		Show all		?						
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baid Brow	
	12-16	2-3			10.1			2.1-2.2								
6-10	17-18	3.5-71	4-78	3.1-12.1	11.5-63	3.2-13.1		2.3-4.4.4	12-12.1				4-9.2			
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1	
		73-74	80-82	TP		13.3										

FLAC audio format - OTHER

Usage: % of all users
Global 88.76% + 3.77% = 92.53%

Popular lossless audio compression format

Current aligned	Usage relative	Date relative	Apply filters	Show all	?										
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser
			4-43												
			44-46												
	12-15	2-50	47-55	3.1-10.1	10-41	3.2-10.3		2.1-2.3	12				4		
6-10	16-18	51-71	56-78	11-12.1	42-63	11-13.1		3-4.4.4	12.1				5-9.2		
11	79	72	79	13	64	13.2	all	76	46	79	68	12.12	10.1	1.2	7.1
		73-74	80-82	TP		13.3									

Real HLS diffusion

