Agenda

• Simple live streaming
• Real (complex) live streaming blocks
• Some challenges
  ◦ Ingest
  ◦ Personalized live streams (SSAI)
  ◦ Monitoring
  ◦ Latency
  ◦ Redundancy
• Future
• Conclusions
WHAT IS LIVE STREAMING?
Simple live streaming
Simple live streaming demo

WebServer (nodeJS):
https://gist.github.com/jordicenzano/a943bda656ad1e0a0568fbaad311af05

RTMP + transcode + HLS (ffmpeg):
https://gist.github.com/jordicenzano/c5516b5854eb046bd1c27076fe4ba917

#!/usr/bin/env bash
http-server ~ -p 8080 --cors -c -1

#!/usr/bin/env bash
ffmpeg -hide_banner \
-listen 1 -i "rtmp://0.0.0.0:1935/live/stream" \
-c:v libx264 -b:v 900k -g 60 -profile:v baseline -preset veryfast \ 
-c:a aac -b:a 48k \ 
-hls_flags delete_segments -hls_time 6 -hls_segment_filename ~/480p_%05d.ts ~/480p.m3u8

RTMP + transcode + HLS (ffmpeg): https://gist.github.com/jordicenzano/c5516b5854eb046bd1c27076fe4ba917
WebServer (nodeJS): https://gist.github.com/jordicenzano/a943bda656ad1e0a0568fbaad311af05
WHAT IS REALLY LIVE STREAMING?
Live streaming blocks

On site / premises

Live encoder

Ingest

Transcode

RTMP

h264

AAC

TS + ?

(h264+AAC)

Nxhr264

NxAAC

Packetizer/
Segmenter

Manifest
Generator

TS

fMP4

HLS

DASH

HTTP[+]

HTTP[+]

CDN

Origin

Viewer

Monitoring
(analytics)
LIVE INGEST
Live ingest

- Data loss
- Latency
- Jitter
- Reordering
- Corruption
- Duplication

From: https://en.wikipedia.org/wiki/Internet_service_provider
Live ingest options: RTMP

**Pros**
- Reliable (TCP based)
- Popular

**Cons** (for live streaming)
- Only 1 video + 1 audio + data
- Connection oriented
- Retries consume BW
- Slow packet loss recovery
- **No real time guaranteed**
Live ingest options

• Pros
  ◦ Can transport **everything**
  ◦ Not connection oriented
  ◦ Designed for live
  ◦ Very common in the broadcast world

• Cons
  ◦ Not reliable
  ◦ Complexity
Live ingest RTMP vs SRT

• Comparison RTMP vs SRT
  ◦ 5% packet loss
  ◦ 50ms delay
  ◦ 11 Mbps BW
  ◦ Test stream
    • 8Mbps VBR (9Mbps max)
    • h264 1 keyframe every 3s

Source code: https://github.com/jordicenzano/live-streaming-ingest-advance-d-simulation
PERSONALIZATION
(SSAI: Server Side ad insertion)
Live streaming

Live SSAI
Live stream personalization

- Considerations
  - One manifest per viewer = 1:1
    - Same for content
    - Different for adbreaks
  - Track every viewer in a "DB"
  - Ultimate solution: Edge?
Live streaming SSAI

- SCTE 35
- Manifest Generator
- Origin
- CDN
- Sessions "DB"
- Ad-123c.mp4
- Ad-123c-0001.ts
- SCTE35 info
  - Session info
  - Job info

- FreeWheel
  - Ad Servers
  - SPOTX
  - DoubleClick
Live streaming SSAI

...[
#EXTINF:8.0,
A-0000.ts
#EXTINF:6.40,
A-0001.ts
#AD-BREAK
#EXTINF:8.0,
A-0002.ts
#EXTINF:4.0,
A-0003.ts
...

Packetizer/
Segmenter

Manifest
Generator

Ads
adaptor

media

FreeWheel
Ad Servers
SPOTX
DoubleClick

...
#EXTINF:8.0,
A-0000.ts
#EXTINF:6.40,
A-0001.ts
#EXT-X-DISCONTINUITY
#EXTINF:8.0,
ad-123c-0001.ts
#EXT-X-DISCONTINUITY
#EXTINF:4.0,
A-0003.ts
...
Live streaming SSAI impressions

- Live streaming with 1M CCU
- 6 ads / adbreak
- 20 imp / ad
- 120 M imp / adbreak
MONITORING
Monitoring

- Considerations
  - Measure all you can
    - Impact of logging / metrics
  - All real time
    - Metrics
    - Logs
  - Aggregate and alarm wisely
  - Automate

- Graphs and data visualizations showing metrics and performance indicators.
LATENCY

Latency ≠ BAD
HLS Latency sources

Encoder 1 x TD Player (3 x TD)

Target Duration = 6s

4.5 x TD
4.5 x 6 = 27s
Low latency “simple” demo

- LHLS (periscope approach)
  - Chunk transfer + advanced segments

Source code:
https://github.com/jordicenzano/go-ts-segmenter
https://github.com/mjneil/go-chunked-streaming-server (From Matthew Neil)
Redundancy blocks

#EXTM3U
#Example of HLS REDUNDANT_STREAMS
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=200000, RESOLUTION=720x480
http://ALPHA.mycompany.com/lo/prog_index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=200000, RESOLUTION=720x480
http://BETA.mycompany.com/lo/prog_index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=500000, RESOLUTION=1920x1080
http://ALPHA.mycompany.com/md/prog_index.m3u8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=500000, RESOLUTION=1920x1080
http://BETA.mycompany.com/md/prog_index.m3u8
Redundancy blocks: Full sync approach
Redundancy blocks: non-sync approach

```plaintext
...#EXTINF:4.000000,
from-A-0000.ts
#EXTINF:4.000000,
from-A-0001.ts
#EXT-X-DISCONTINUITY
#EXTINF:4.000000,
from-B-0001.ts
....
```

A
- Live encoder
- Ingest
- Transcode
- Packetizer/Segmenter
- Origin A.m3u8
- CDN
- Manifest Generator
- HA Generator
- HA.m3u8

B
- Live encoder
- Ingest
- Transcode
- Packetizer/Segmenter
- Origin B.m3u8
- CDN
- Manifest Generator
- HA Generator
- HA.m3u8
Future

- Do more at the edge with multi CDN
- Personalized streams
  - Cohorts approach (hotstar 2019)
- Low latency deployments
  - ABR
- Codecs
  - More, faster, deeper pixels
  - Context aware encoding
Conclusions: Experience based

- Everything fails, and we are live
  - Plan B, plan C, ,,,, plan Z
  - Better hot - hot

- Be sure the involved 3rd party are ready
  - CDNs, AdServers

- Load test (not easy)
  - There are limits everywhere
  - Individually
  - End to end

- Enjoy the challenge 😁
Thanks!

Jordi Cenzano
cenzano@brightcove.com
jordicenzano