Comparing Just-in-Time Packaging with CDN Storage for VoD and nDVR Applications

A model for real-time segmentation and delivery of multi-format HTTP streaming protocols

First presented at SCTE Canadian Summit, March 2012

By Yuval Fisher, CTO, RGB Networks

1. Introduction – Adaptive HTTP Streaming

Multiscreen video delivery makes use of HTTP streaming based on protocols defined by Apple (HTTP Live Streaming or HLS [HLS]), Microsoft (Silverlight Smooth Streaming or MSS [MSS]), and Adobe (HTTP Dynamic Streaming or HDS [HDS]). These protocols have spawned a new component in the video delivery chain – the packager (sometimes also called a segmenter or fragmentor). This component is used to create the segmented video files that are delivered over HTTP to clients that then stitch the segments together to form a contiguous video stream. The packager is sometimes integrated into the encoder/transcoder that first creates the digital encoding of the video, but often it is a separate component.

Adding to the work of the packager, an MPEG standard called DASH [DASH], YAA (yet another acronym) for dynamic adaptive streaming over HTTP, attempts to unify the other protocols under one open standard umbrella. In the near term, and possibly for longer, if HLS, MSS and HDS do not disappear, DASH adds more formats that service providers may need to address. In fact, DASH has several profiles that have very different underlying delivery formats, so that it may be necessary for packagers to serve not just HLS, MSS and HDS, but an MPEG-2 TS DASH profile and a base media file format DASH profile as well.

As multiscreen video services mature, the packager component has become more complex, leading to more complex features and use cases. In this paper, we focus on one specific use case: just-in-time packaging (JITP), which has applications for video-on-demand (VoD) and network digital video recorder (nDVR) applications, including catch-up and restart TV. In all of these applications, each client makes a separate request to view video content from its beginning, so that unlike broadcast video, the sessions are independent. In legacy delivery, a video streamer is used to stream the video to the client. In HTTP streaming, two options are possible: either the assets are stored in an HTTP-ready format, so that clients can make HTTP requests for video segments directly. Or, assets can be stored in a canonical (or mezzanine) format which is then converted to HTTP segments as the client makes requests for them – just-in-time. The first option is more disk memory intensive, while the second is more computationally intensive.

In the sections below, we review the packaging ecosystem, touching on various modes in which a packager can function. We then analyze the tradeoff between memory, computation and core network bandwidth when JITP is used. We propose a model and demonstrate that in most cases, JITP is considerably less expensive than pre-storage of multiple HTTP streamable formats. Lastly, we touch upon a related option of just-in-time (JIT) transcoding, in which the digital representation/format of assets is converted (transcoded) when a request is made.

Video-on-Demand Use Cases

There are multiple VoD use cases which offer a tradeoff between storage and computation. These use cases are all examples of VoD for which there is no standard terminology – so we define them below:

Standard Video-on-Demand

In this use case, typically "high-value" content, such as movies or television programs, is available on demand to the user. Users connect to the service, request an asset and receive a stream. VoD libraries can be very large (e.g. 150,000 hours), but not all assets are viewed with the same frequency. The assets that are viewed often are known as "short tail" content, while the content this is viewed far less is called "long tail." If every asset request is served from a central location, both long tail and short tail content is treated the same way, but this requires sufficient core network bandwidth to handle all the requests. In many cases, the bandwidth requirement makes distributed storage of assets closer to the edge attractive. It then becomes a complex optimization problem to know how much edge storage is optimal. Operators gather (but don't often share) data on what percentage of users access the short-tail content or what percentage of long tail content is viewed at all.

Network Digital Video Recording

In this use case, users specify programs that they want recorded on storage devices owned and managed centrally by the service provider. Typically, such content is available for 30 days and can be viewed on demand at any time via streaming to the client device.

The so-called "Cablevision ruling" (see [Cablevision]) stipulates that U.S. operators must store a unique copy of each asset for each user, mirroring the consumer's option to store the content at home per standard copyright fair-use statutes. In the U.S., where the Cablevision ruling holds, popular shows will be stored many times, with one copy stored per user. If users have the option to view back that content on devices that make use of different transport formats, then the content would potentially need to be stored in each format, resulting in very high storage requirements and cost.

Even in locations where the Cablevision ruling does not hold, storage requirements are high. For example, storing 30 days of content for 100 channels, assuming every show available is stored by someone, requires 260TB of storage (assuming 8Mbps bitrate for the content). Multiplying this by three to five formats results in significant storage requirements.

Restart TV/Catch-up TV

In this use case, users record programs as they are being watched. The user has the option of pausing the live stream, restarting it from the beginning or catching up from a paused program to the live broadcast as it is happening. The infrastructure for delivering this use case is very similar to standard nDVR, but programs are not stored as long – typically just while the program is broadcast or for a small number of hours or days after. Nevertheless, when delivery occurs over different formats, storage for each format is required—potentially storage per user when the Cablevision ruling applies.

Mitigating High Storage Requirements

One way to minimize storage requirements is to create the delivery format as needed for each client. The underlying video and audio codecs are identical for all the delivery formats discussed, so that only a muxing operation is needed. However, modeling the computational demand required depends on each of these use cases. For standard VoD, concurrency rates are relatively low at around 5%. That is, at any moment, it is expected that a peak of about 5% of subscribers will be utilizing the standard VoD service. For nDVR, however, a significant proportion of recorded programs will be viewed – perhaps around 50%. The concurrency of users using nDVR at any time may be closer to the VoD concurrency, but specific estimates are not available from service providers. The restart/catch-up TV concurrency is also difficult to estimate. In this paper, we assume a 5% concurrency and don't distinguish between the use cases further.

2. The Packaging Ecosystem

For context, we review four different modes of packaging:

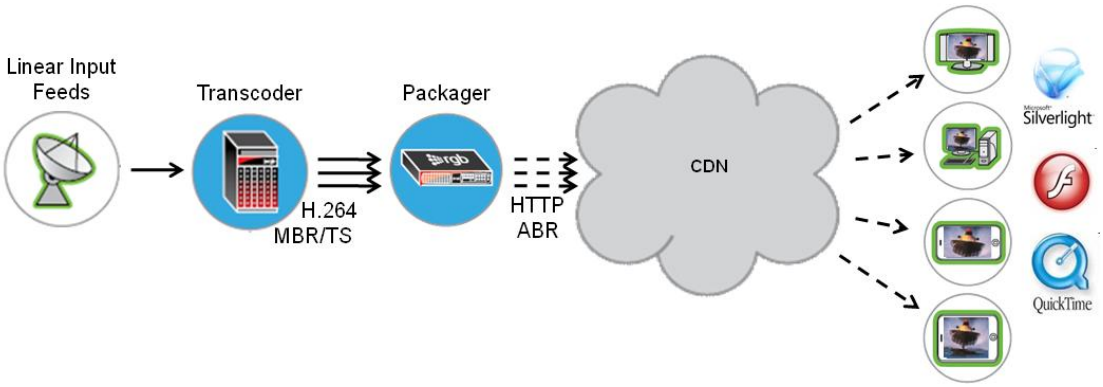
- Linear packaging
- File-based packaging
- Just-in-time packaging
- Edge packaging

All of these modes create the manifests and video file fragments needed for HTTP streaming and optionally encrypt the output, but each mode addresses different use cases.

Linear Packaging

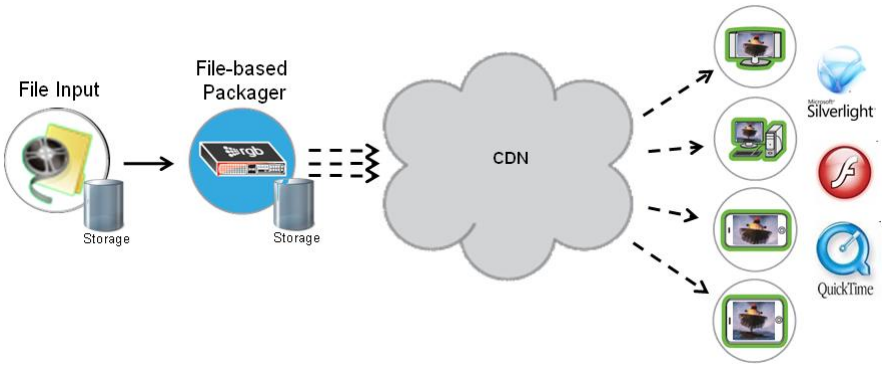
In linear packaging, live video is converted to a stream, typically an MPEG-2 Transport Stream (TS) that is ingested by a packager. The packager segments the stream into files and either serves as

the origin of a content distribution network (CDN) or pushes the segments, or chunks, and manifests into such an origin. The CDN is responsible for delivering the chunks to the client and for duplicating chunks at the edge so that not all client requests are fed back into the core network and to the packager. This is shown in the figure below: the output of the transcoder is a collection of H.264/AAC-LC encoded video/audio streams carried in a multi-bitrate (MBR) collection of MPEG-2 SPTS streams. The output of the packager is a collection of adaptive bitrate (ABR) HTTP streams which are delivered into the CDN.



File-based Packaging

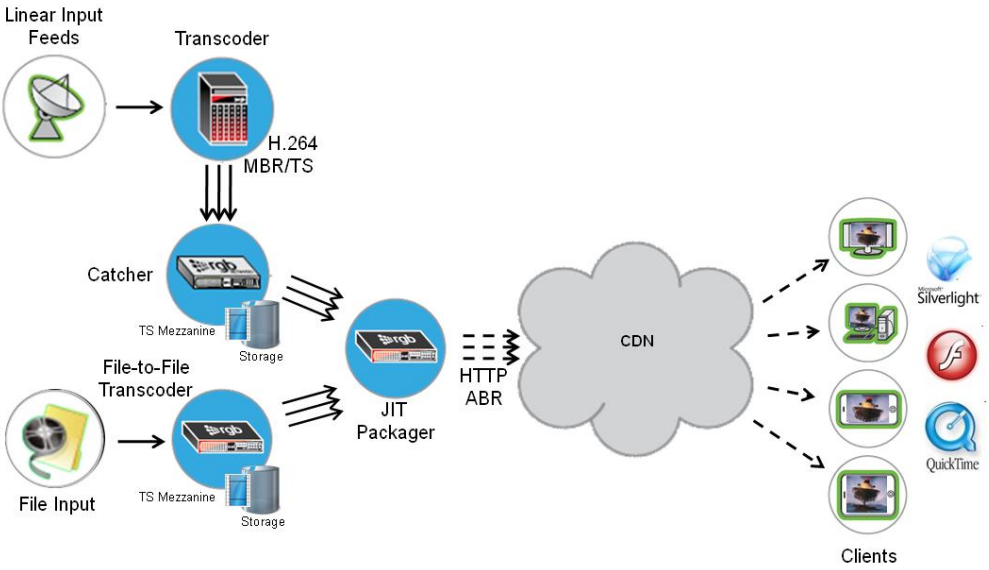
File-based packaging, shown below, is used to convert a file from one format into an HTTP deliverable format, for example from a TS file format suitable for legacy TS streamers into an HLS segmented format. With offline packaging, all the segments (and playlists) are created and stored for subsequent delivery.



Just-in-time Packaging

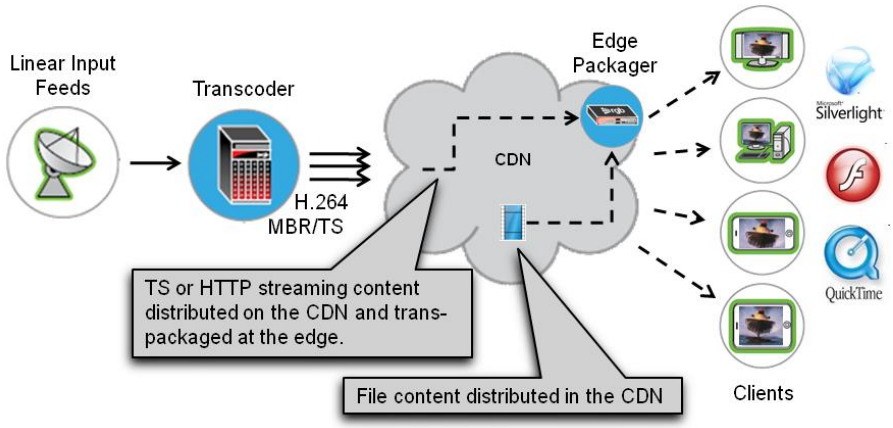
In a typical JITP use case, shown below, live content is first transcoded into MBR outputs and captured by a “catcher” component that converts the live streams into files in a chosen mezzanine format. Alternatively, file assets, rather than live streams, are transcoded into a mezzanine format which uses H.264/AAC for the video/audio codecs and a pre-selected container format. MPEG-2 TS container format is a natural choice for the mezzanine files, since it can contain the signaling (e.g. SCTE 35 cues or other PID data) present in the original signals.

When clients connect to the JIT packager, it extracts the requested chunks (or computes the manifests) from the mezzanine files and delivers them to the clients.



Edge Packaging

Edge packaging comes in various flavors, including JIT, off-line and linear. In this use case, the packager is located at the edge of the CDN rather than at its core. The packager function is almost the same as when it is located at the network's core. For example, one feature that may be found in an edge packager that would not be needed in a core-located packager is the ability to trans-package from one ABR format to another.



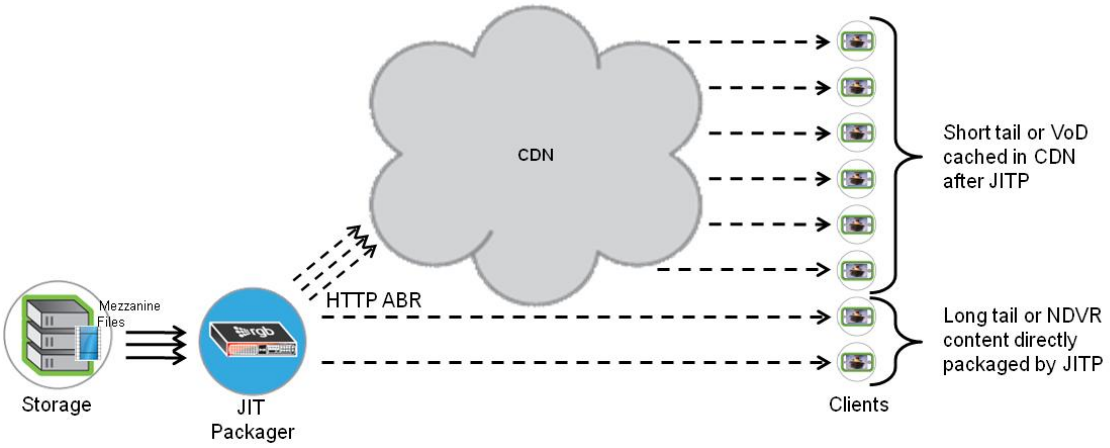
3. Just-in-Time Packaging

There are a number of reasons why JITP may be a better alternative to pre-positioning assets in all final delivery formats:

- *Storage cost savings:* When multiple HTTP streaming formats are used, every asset must be stored in multiple formats, with associated storage costs.
- *"Cablevision ruling" nDVR copies:* Service providers must store a separate copy of every asset that users record for nDVR playback. Storing a single mezzanine format reduces storage requirements.
- *Format future-proofing:* The HTTP streaming protocols in use today are still evolving, and doing JITP of mezzanine format assets eliminates the need to re-package VoD libraries when these formats change.
- *Single work-flow:* Using JITP for VoD with a caching CDN can automatically lead to caching of short tail assets in the CDN and the use of JITP for un-cached long-tail assets.

JITP Use Cases

The use cases for JITP range between two extremes; at one extreme, only JITP is used with no storage of assets in an HTTP-deliverable format, and at the other extreme, all assets are stored in multiple formats and no JITP is used. The middle use-case is shown in the figure below. Here, most clients receive their content from the CDN, which caches the short tail content. Long tail content is served directly from the JIT packager. The CDN must differentiate between long and short tail content by amassing statistics on requested chunks, caching frequently requested chunks and sending cache misses, the long tail, directly to the JIT packager.

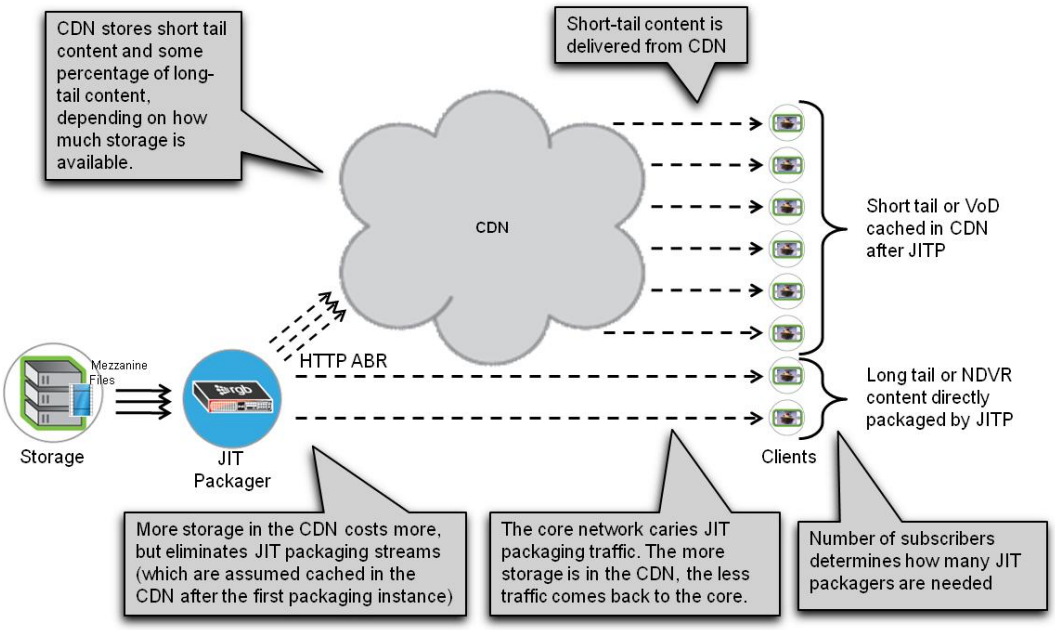


4. Modeling Storage and Bandwidth Capacity vs. JITP

The model used to estimate the cost of memory converted into equivalent JITP cost per stream is explained in the figure below. The model looks at an instantaneous snapshot of JITP, core network traffic and CDN storage.

The modeling assumptions for JITP are as follows:

- Use of JITP offsets the need to store content in multiple HTTP streaming formats. At any instant, the storage in the CDN can hold some percentage of the total available content. The more long tail can be stored in the CDN, the less core traffic is needed to JIT packagers. At the extreme edge of the model, the CDN can cache all the content, and no JITP is needed. The instantaneous model ignores the offline (or JIT) packaging that was used to fill the CDN cache, but this is not a weakness, since it adequately captures the instantaneous balance of storage to JITP capacity.
- In this paper we assume the long tail content has a flat distribution – that is, anything that's not considered "short tail" will be viewed by someone at some point. For very large libraries, this may not be true, as some assets are basically never viewed.
- We assume that all VoD use cases have the same concurrency, thus the model is most apt for standard VoD.
- JITP potentially increases core network usage, as clients need access to the core if JITP is done there. Note that using edge JIT packagers means that mezzanine formats would need to be stored multiple times at the edge, which diminishes the value of JITP; thus, the model we examine here assumes JITP at the core only.
- Note that a relatively small number of file-based packagers can convert even large libraries into multiple formats. For example, six packagers can convert a 100,000 hour library into three formats in about four days. Thus, the cost of off-line packaging (or the equivalent JIT workflow) for feeding CDN caches is ignored in the model.



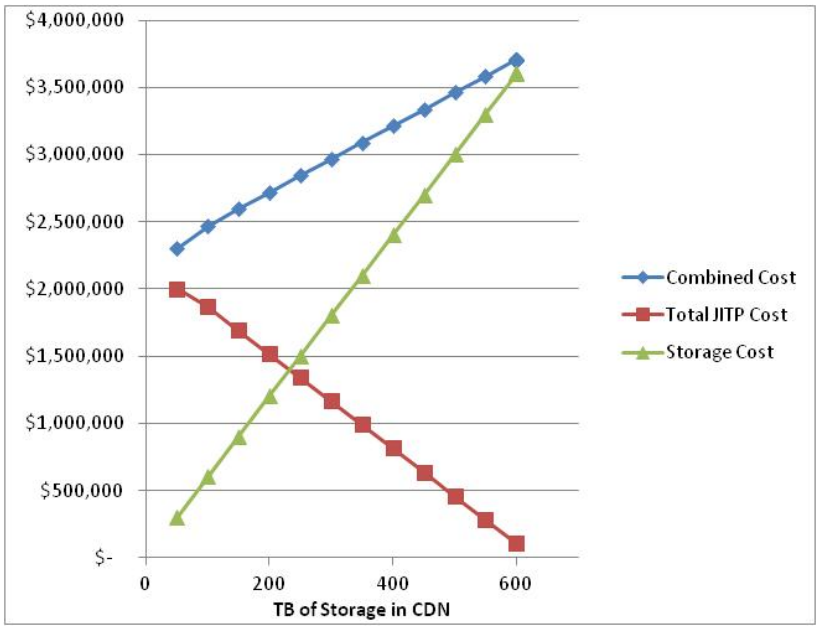
The model to measure these tradeoffs has the following variables:

Variable	Details	Model Value(s)
Number of subscribers	As the number of subscribers increases, core network usage and the number of JIT packagers increases.	100K-10M
Peak concurrency	This is the peak percentage of users that use the service at the same time.	5%
Core storage capacity	This is the core storage in the CDN. When large, more client requests (long and short tail) can be served from the CDN, thus requiring less JITP at any instant.	0TB-750TB
Cost per TB of storage	The cost of professional storage ranges widely depending on negotiated bulk purchase agreements. It falls somewhere between US$1,000 - $10,000 per terabyte.	US$1,000
VoD library size	A larger VoD library requires more storage. Library sizes vary widely between different providers. We assume the library size is in the range of 5,000 to 100,000 hours.	5K-100K hours
Cumulative MBR bitrate	Adaptive HTTP streaming uses multiple profiles with different bitrates (and resolutions). The cumulative bitrate of all the profiles determines the storage needed for each HTTP streaming format. In this model, we assume a middle value, corresponding roughly to six profiles at bitrates of 3Mbps, 1.5MBps, 1Mbps, 750kbps, 500kbps and 250kbps.	7Mbps
Highest MBR bitrate	This is used to calculate the peak core network usage.	3Mbps
Number of MBR formats	There are (currently) three viable options: Apple HLS, Adobe HDS and Microsoft Smooth Streaming.	3

Amount of short tail content	In the model, we assume that short tail content is cached at the CDN edge. This eliminates a significant proportion of network traffic into the core at a relatively inexpensive storage cost.	500-5,000 hours
Average asset duration	This is used to convert the library size into a number of unique assets.	1 hour
Percentage of short tail stream requests	Most viewers watch a small percentage of the content – the short tail.	90%
CDN edge storage scaling factor	This is a multiplicative factor that accounts for the fact that a tiered CDN has storage both at the core and at the edge. Storing content on the CDN means that it is potentially stored in the core, mid-tiers, and the edge. Thus a TB of content would require 3TB to be fully stored on the CDN.	3
JITP stream cost	This is the cost per stream of JITP. This is sometimes used as input to the model and sometimes derived as an output. At the low end, hardware costs and concurrency rates provide a lower bound on this cost at about $15/stream.	$15-$5K
Long tail behavior	This isn't a variable, but an assumed behavior in the model. Requests that hit the long tail are assumed to be distributed evenly within the long tail content.	Evenly distributed

Results

Various results and cross sections from this model are described below. The assumptions for the results of various models are shown in the tables below.



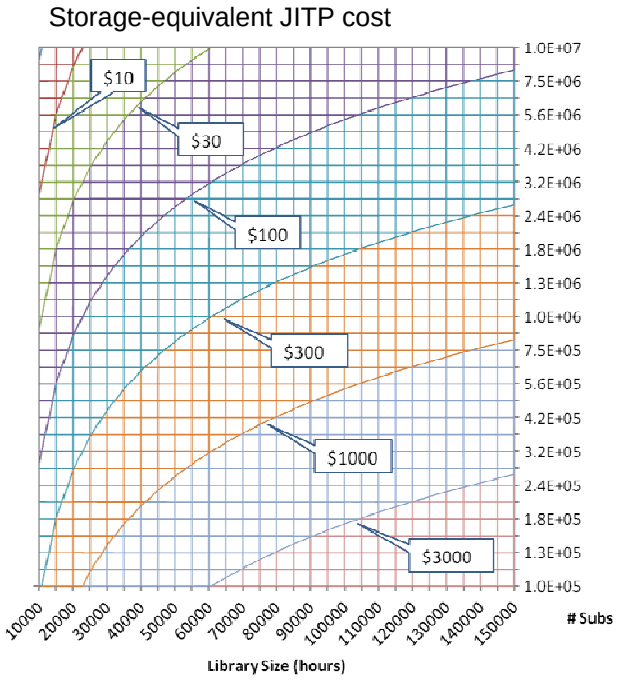
Assumption	Model A
Library size (hours)	50,000
Short tail (hours)	5,000
Average Asset duration (hours)	1
MBR bitrate (Mbps)	7
Number of subscribers	1M
Peak concurrency	5%
Long tail stats: % of people hitting short tail	90%
Cost of storage ($/TB)	US$2,000
CDN Storage tiered multiplicative factor	3
Number of ABR formats	4
Cost per JITP stream	$400

The figure above shows the combined cost of JITP and storage as a function of the amount of storage in the CDN, assuming a per-stream JITP cost of $400 and 1M subscribers. As the CDN

storage increases, more of the content can be delivered from CDN caches rather than requiring JITP– thus the associated cost of JITP falls. With the configuration of model A in the table above, it is cheaper to have as little storage as possible and as much JITP as possible – more storage just costs more overall. However, at higher JITP costs or larger total subscriber count or smaller library sizes, the slope of the "combined cost" curve reverses, and it becomes more economical to use only storage and avoid JITP.

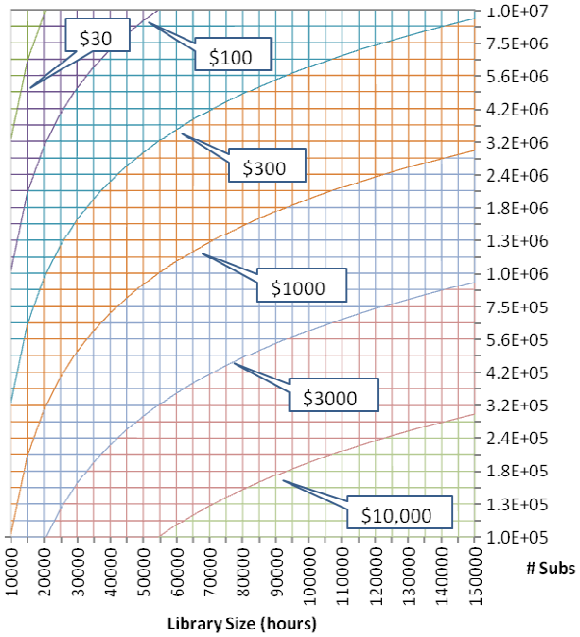
The figure below shows bands of the storage-equivalent cost per stream of doing JITP rather than storing every asset in the library in multiple formats as a function of the library size and number of subscribers. The figure shows that except for a band (in the upper left corner) of high subscriber count and low library size, memory is equivalent to a JITP cost of over $300 per stream. Thus, for example, a service provider with about 750K users and a library of 130K hours paying $500 per JITP stream would spend half as much on JITP as compared with storage (since the graph shows the memory-equivalent JITP cost per stream to be about $1000).

With slightly less conservative estimates, as shown in model C, where four formats are delivered (e.g. HLS, HDS, MSS, and one DASH profile) and the memory cost is US$2000/TB, the area of subscriber-count/library-size where JITP can offer significant factors of savings over storage increase dramatically.



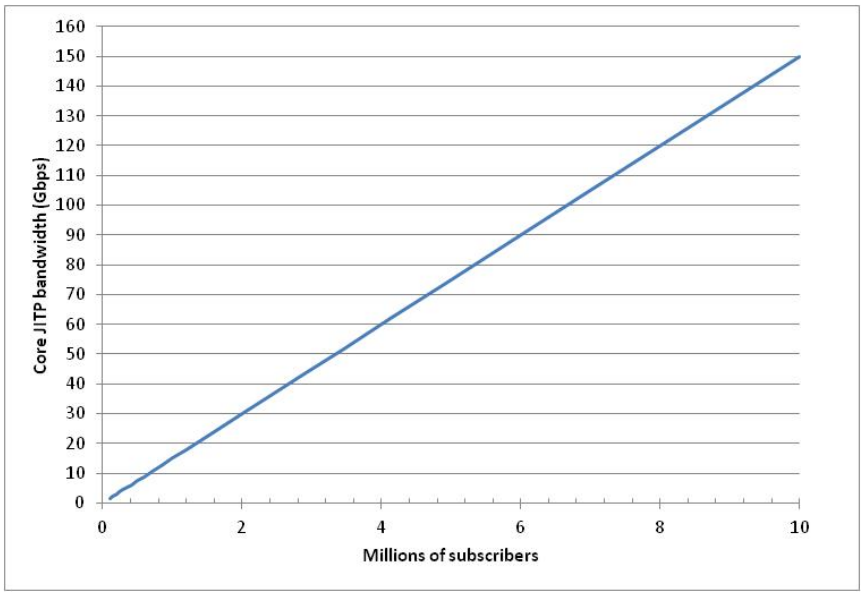
Assumption	Model B
Library size (hours)	10K-150K
Short tail (hours)	5,000
Average asset duration (hours)	1
MBR bitrate (Mbps)	7
Number of subscribers	100K-10M
Peak concurrency	5%
Long tail stats: % of people hitting short tail	90%
Cost of storage ($/TB)	US $1,000
CDN storage tiered multiplicative factor	3
Number of ABR formats	3

Storage-equivalent JITP cost



Assumption	Model C
Library size (hours)	10K-150K
Short tail (hours)	5,000
Average asset duration (hours)	1
MBR bitrate (Mbps)	7
Number of subscribers	100K-10M
Peak concurrency	5%
Long tail stats: % of people hitting short tail	90%
Cost of storage ($/TB)	US $2,000
CDN storage tiered multiplicative factor	3
Number of ABR formats	4

The core bandwidth used for JITP can be estimated by assuming all users receiving JITP streams are receiving the highest bandwidth available, assume here to be 3Mbps. This is shown in the figure below. The figure shows that even for the 10M subscriber use case, the core network bandwidth is a manageable 150Gbps. Note that most of the "edge" bandwidth is covered by the CDN's caching of short tail content.



5. Conclusion

When memory-equivalent JITP costs are high, it may in fact be beneficial to utilize just-in-time transcoding as well. In that case, VoD or nDVR assets could be stored in an MPEG-2 encoded mezzanine format. These assets can then be used to serve traditional MPEG-2 set-top boxes and TVs. When a client makes a request, the MPEG-2 asset is transcoded on demand and subsequently packaged just-in-time. The advantage of doing this is that the mezzanine (MPEG-2) format can serve more services and doesn't have to be stored in a separate multi-bitrate format, thus saving more storage space. However, since the density of JIT transcoding is significantly less than JIT packaging, this use case can only be supported when the JIT packaging costs in the model above are significantly higher, so that some of the cost can be used for JIT transcoding.

Either way, JITP can represent a significant cost savings over storage. While storage prices have declined, they have not declined rapidly for high-capacity, high-availability applications. Moreover, computation costs are declining as well, so that it is plausible to expect that JITP will continue to rival storage for at least a few years. But perhaps more importantly, JITP gives service providers future-proof flexibility amid a rapidly shifting technology landscape. The ability to shift to new variants of existing transport formats or to altogether new formats such as DASH gives service providers ample reason to implement both VoD and nDVR functionality using JITP.

6. References

[Cablevision] 2nd Circuit Court ruling on network DVR
http://www.ca2.uscourts.gov/decisions/isysquery/339edb6b-4e83-47b5-8caa-4864e5504e8f/1/doc/07-1480-cv_opn.pdf

[HLS] HTTP Live Streaming, R. Pantos, http://tools.ietf.org/html/draft-pantos-http-live-streaming-06

[MSS] IIS Smooth Streaming Transport Protocol,
http://www.iis.net/community/files/media/smoothspecs/%5BMS-SMTH%5D.pdf

[HDS] HTTP Dynamic Streaming on the Adobe Flash Platform,
http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming_wp_ue.pdf

[DASH] ISO MPEG 23009-1 Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats

Abbreviations & Acronyms

AAC	Advanced Audio Coding
ABR	Adaptive Bitrate
CDN	Content Delivery Network
DASH	Dynamic Adaptive Streaming over HTTP
HDS	HTTP Dynamic Streaming
HLS	HTTP Live Streaming
HTTP	Hypertext Transfer Protocol
JIT	Just-in-Time
JITP	Just-in-Time Packaging
MBR	Multi-bitrate
MPEG	Moving Picture Experts Group
MSS	Microsoft Smooth Streaming
nDVR	Network Digital Video Recorder
PID	Program ID
TB	Terabit
TS	Transport Stream
VoD	Video-on-Demand