Modeling, Identifying, and Simulating Dynamic Adaptive Streaming over HTTP

Andrew Reed, Jay Aikat
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599, USA
{reed, aikat}@cs.unc.edu

***Abstract*— As HTTP-based streaming video applications have grown to become a major source of Internet traffic, and as the new ISO standard Dynamic Adaptive Streaming over HTTP (DASH) gains industry acceptance, researchers need the ability to both study real-world viewing data and simulate realistic DASH streams. The first effort is complicated by the fact that network researchers are often restricted to anonymized, header-only traces. The second effort is difficult since researchers are currently limited to three undesirable options for generating DASH traffic: (1) encode and store movies, which is both labor- and storage-intensive, (2) parameterize DASH traffic, which is open to criticism, or (3) stream movies from a service such as Netflix, which is prohibitively expensive if a researcher wants to simulate many simultaneous viewers.**

In this paper we present our work towards developing a model for DASH traffic and show how the model can be applied to identify DASH streams from anonymized, header-only traces using a combination of Hadoop and Hive. We then describe how the insight gained from our work will be used in a simulator that can recreate the DASH payloads of real movies using a format that requires only a few kilobytes of storage per movie.

Keywords—Internet; Streaming media; Data analysis; Computer simulation; Content distribution networks

I. Introduction

A. DASH Design Overview

DASH movies are first segmented into equal length time slices and encoded at various bitrates, or quality levels. These video segments are then served from content distribution network (CDN) nodes over HTTP. During playback, DASH clients continually gauge the available bandwidth to determine which quality level of each segment should be requested. If bandwidth is limited, DASH clients will request segments of lesser quality, resulting in smaller application data units (ADUs). When more bandwidth is available, DASH clients will instead request higher quality segments, resulting in larger ADUs.

B. DASH Traffic Properties

Our model of DASH traffic is based on the following four properties which follow from DASH's design:

1. Outbound ADUs, which represent the HTTP GETs for successive video segments, are sent at regular intervals roughly corresponding to the length of each segment.
2. The sizes of the outbound ADUs exhibit low variance due to the URL naming convention for segments.
3. Since video segments represent a constant length of the movie, the maximum size for any inbound ADU will be constrained by the bitrate, or quality level, supported by a client's available bandwidth (i.e. envision the inbound ADU sizes as the values of a waveform where the quality level *clips* the wave). A video service's highest offered bitrate will therefore impose an upper limit on the size of all inbound ADUs.
4. Since new video segments are requested as buffered segments are consumed, the average inbound data rate will be roughly equivalent to the bitrate of the movie.

In Section II we describe how we measured these properties to obtain data for our DASH traffic model. In Sections III and IV we present our preliminary results and show how the model can be used for both the identification and the simulation of DASH streams.

II. Methodology

A. Traffic Capture

We used adudump to trace the network traffic of a Windows 7 virtual machine while the VM streamed a single Netflix movie at a time. adudump reconstructs and logs ADUs in real-time by analyzing only the IP and TCP headers as described in [1]. For each ADU, adudump records the timestamp, sender's IP.port, receiver's IP.port, and size.

B. Experimental Design

Netflix allows users to select one of four quality levels which serve to cap the maximum bitrate for an account: (1) Best - High Definition, (2) Best - Standard Definition, (3) Better, and (4) Good. We captured four movies at each quality level using the three supported web browsers (Internet Explorer, Firefox, and Chrome), for a total of 48 traces. The lengths of the movies used were 7 min, 17 min, 46 min, and 110 min. In addition to the 48 Netflix-only traces, we traced over 45 hours' worth of Internet activity containing no Netflix traffic.

C. Data Analysis

To gather data for our model, we wrote a Hadoop program called "DASH Detective" which analyzes adudump traces and reports the following statistics for each HTTP connection that lasted at least 30 seconds:

- *Duration*: Length of the connection.
- *Average ADU Out*: The average size of all ADUs sent from the local IP and port to the remote HTTP server.
- *ADU Out Standard Deviation*: Standard deviation for the above.
- *Average Interval*: The average time between ADUs sent from the local IP and port to the remote HTTP server.
- *Interval Standard Deviation*: Standard deviation for the above.
- *Max ADU In*: The largest ADU sent by the remote HTTP server to the local IP and port. Think of this value as the *clipping level* for a video stream.
- *Average Data Rate*: The average data rate for the ADUs received from the remote HTTP server (calculated using 10 second interval averages).
- *Data Rate Standard Deviation*: The standard deviation of the 10 second interval averages.

III. Preliminary Results

Table 1 lists the results for the Netflix streams. As an indication of our model's strength, these streams could have been identified, with no false positives from the non-Netflix traffic, by using a Hive query based on the ranges in Table 1.

Table 2 lists the observed ranges for *Max ADU In* and *Average Data Rate* when the streams are grouped by quality level. The results for *Max ADU In* represent the ranges in which the segment sizes of a video stream were *clipped*. Notice that these ranges do not overlap. The same cannot be said of the *Average Data Rates*, as these results are sometimes inflated by aggressive buffering which will cause the transfer of video segments to complete much sooner than the actual movie.

For each of the movies, we extracted all of the inbound segments from the Internet Explorer traces and summed the total data received for each quality level. Table 3 shows the data received for the lower levels relative to *Best HD*. These percentages only apply to the total size of the movie; individual segments varied in a range around these percentages. Notice the similarity in the results for the 17-, 46-, and 110-min movies. This seems to indicate that, for all but the shortest movies, the segment sizes for lower quality levels of a movie can be approximated as a constant percentage less than the highest quality level.

TABLE I. DASH Detective Results

Statistic	Min	Max
Average ADU Out (B)	433	570
ADU Out Standard Deviation (B)	1	10
Average Interval (s)	1	4
Interval Standard Deviation (s)	2	3
Max ADU In (B)	481,107	3,275,999
Average Data Rate (Kb/s)	469	3,095
Data Rate Standard Deviation (Kb/s)	174	2,145

TABLE II. Max ADU In and Avg Data Rate by Quality Level

Quality Level	Max ADU In (B)		Avg Data Rate (Kb/s)	
	Min	*Max*	*Min*	*Max*
Best HD	2,210,954	3,275,999	2,038	3,095
Best SD	1,642,325	1,885,911	1,620	2,089
Better	891,322	1,086,177	622	1,551
Good	481,107	580,330	469	1,385

TABLE III. Total Data Received by Quality Level

Movie	Best HD	Best SD	Better	Good
7 min	102,434,048 (B)	68.3%	54.6%	32.2%
17 min	367,076,893 (B)	59.8%	36.1%	19.5%
46 min	1,055,834,938 (B)	59.5%	36.6%	20.7%
110 min	2,249,302,778 (B)	59.7%	40.8%	19.4%

IV. Future Work

These results point toward the effectiveness of our model as a means to identify DASH traffic from anonymized, header-only traces and then estimate the quality levels of captured DASH streams. Since our analysis uses Hadoop and Hive, it is capable of scaling to a trace of any size. Indeed, our next step is to analyze campus-wide UNC traces in order to refine our Hive query ranges and to better gauge the model's accuracy.

We are also developing a DASH traffic simulator that will store captured movies as individual text files ordering on 7KB for every hour of video. Each text file will only need to list the sequence of video segment sizes for the highest quality level of a movie. Dummy traffic can then be generated by the distributed clients and CDN nodes using either the percentages from Table 3 or custom percentages to approximate the various quality levels. A researcher can easily generate these text files by capturing the HTTP GETs sent during a high quality stream with a browser plug-in (e.g. HttpFox for Firefox) and then use a script to calculate the sequence of segment sizes from the byte ranges in the URLs. Since these text files are only representations of actual movies, they can be shared among the research community without violating copyright law. Our distributed client will be designed so that researchers can either implement their own strategies for CDN selection, buffering, etc., or use the default methods that we provide.

References

[1] J. Terrell, K. Jeffay, F. Smith, J. Gogan, and J. Keller, "Passive, Streaming Inference of TCP Connection Structure for Network Server Management," in Traffic Monitoring and Analysis: First International Workshop, 2009.