Modeling, Identifying, and Simulating Dynamic Adaptive Streaming over HTTP (DASH)

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Research Problem

Dynamic Adaptive Streaming over HTTP (DASH) operates by serving movie segments, each encoded at multiple bitrates, from standard web servers. During playback, successive movie segments are retrieved by a DASH application using HTTP GETs. Thus, while DASH provides industry with an effective means to serve content to users, its use presents network researchers with the following problems:

Identification

Since DASH video appears as standard HTTP traffic:

- It is hard for researchers to isolate and study DASH streams, especially if they are restricted to anonymized, header-only traces. This also means that DASH traffic has the potential to skew studies of "normal" browsing activity if it is not removed from a trace.
- It is difficult to apply QoS to DASH streams from unknown IP addresses without applying the same QoS to all HTTP traffic.

Question #1

Can DASH streams be distinguished from other HTTP traffic both **quickly** and **effectively** given either a **live, high-speed interface** or an **anonymized, header-only trace**?

Question #2

Can the movie represented by a DASH stream be **fingerprinted** and **identified** by the **sizes** of the stream's **video segments**?

Simulation

Since DASH segment sizes and buffering activity will vary based on network conditions, captured DASH streams should not be replayed in network simulations. Thus, to generate DASH traffic, researchers are currently presented with these three options:

1. Encode and store movies on an HTTP server and use real DASH clients to stream and play DASH movies.
   - Labor- and storage-intensive. This approach is impractical for an environment such as GENI due to storage and time constraints.
2. Parameterize DASH traffic.
   - Open to criticism.
3. Stream movies from a service such as Netflix or Amazon Instant Video.
   - Expensive at scale.
   - Introduces uncontrollable conditions to experiments.

Question #3

Can a DASH infrastructure be **replicated**, and can DASH streams be **emulated**, in a manner that requires **little storage** and **low CPU utilization**?

Preliminary Results

Overall Results (Table 1)

DASH Traffic Model Measurement	Min	Max
Average ADU Out (bytes)	433	570
ADU Out Standard Deviation (bytes)	1	10
Average Interval (seconds)	1	4
Interval Standard Deviation (seconds)	2	3
Max ADU In (bytes)	481,107	3,275,999
Average Data Rate (Kbits/s)	469	3,095
Data Rate Standard Deviation (Kbits/s)	174	2,145

- *Table 1* shows the Min/Max for each measurement across all of the captured DASH connections.
- *Table 2* points to the possibility of identifying the quality level of a captured DASH stream.
- *Table 3* indicates that lower quality levels can be approximated by constant percentages.
- *Fig. 1* suggests that individual movies can be reproduced/fingerprinted by their segment sizes.

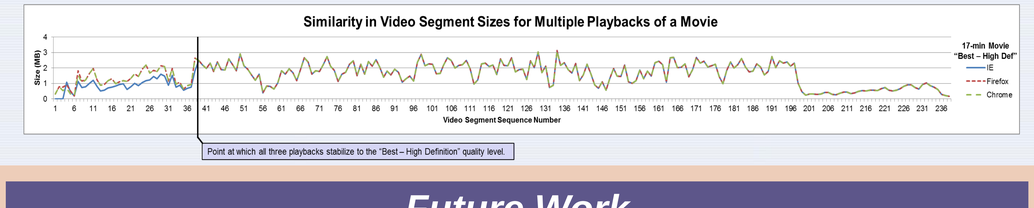
Determining a Stream's Quality Level (Table 2)

	Max ADU In (B)		Avg Data Rate (Kb/s)	
Quality Level	*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

Approximating Lower Quality Levels (Table 3)

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%

Reproducing / Identifying a Specific Movie (Fig. 1)



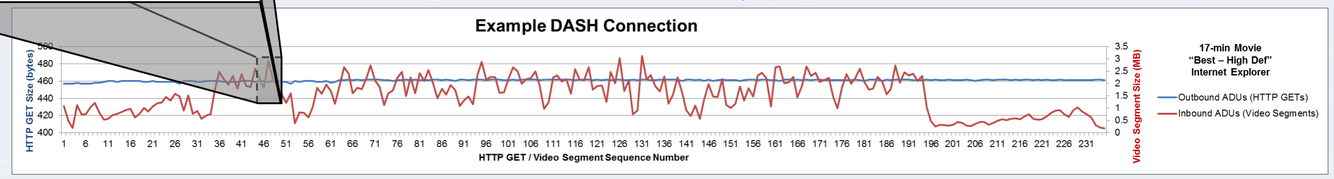
DASH Traffic Properties

adudump Trace

```
ADU: 1371320907.341094 192.168.19.137.49251 > 205.128.73.253.80 460
ADU: 1371320911.000346 192.168.19.137.49251 < 205.128.73.253.80 2217469
ADU: 1371320911.050769 192.168.19.137.49251 > 205.128.73.253.80 460
ADU: 1371320913.971590 192.168.19.137.49251 < 205.128.73.253.80 1819596
ADU: 1371320916.689535 192.168.19.137.49251 > 205.128.73.253.80 459
ADU: 1371320920.596368 192.168.19.137.49251 < 205.128.73.253.80 2912520
ADU: 1371320920.643804 192.168.19.137.49251 > 205.128.73.253.80 460
ADU: 1371320924.395510 192.168.19.137.49251 < 205.128.73.253.80 2152791
ADU: 1371320924.444516 192.168.19.137.49251 > 205.128.73.253.80 460
ADU: 1371320927.407864 192.168.19.137.49251 < 205.128.73.253.80 1920372
```

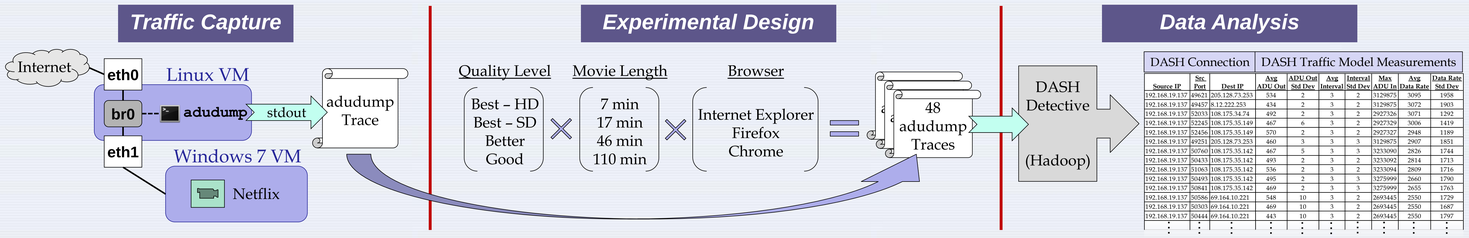
Our model of DASH traffic is based on the following four properties which follow from DASH's design:

1. **Outbound ADUs* 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. **The maximum size for a segment is capped** by the bitrate supported by the client's available bandwidth.
4. **The average inbound data rate** will be roughly equivalent to the **bitrate of the movie**.



* Application Data Unit

Gathering Baseline Data for the DASH Traffic Model



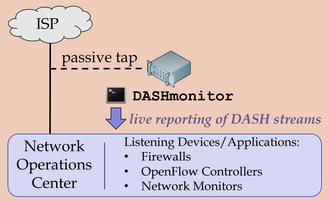
DASH Connection				DASH Traffic Model Measurements									
Source IP	Src Port	Dest IP		Avg ADU Out	ADU Out Std Dev	Avg Interval	Interval Std Dev	Max ADU In	Avg Data Rate	Data Rate Std Dev
192.168.19.137	49621	205.128.73.253		534	2	3	2	3129875	3095	1958
192.168.19.137	49457	8.12.222.253		434	2	3	2	3129675	3072	1903
192.168.19.137	52033	108.175.34.74		492	2	3	2	2927326	3071	1292
192.168.19.137	52243	108.175.35.149		467	6	3	2	2927329	3006	1419
192.168.19.137	52456	108.175.35.149		570	2	3	2	2927327	2948	1189
192.168.19.137	49253	205.128.73.253		460	3	3	3	3129875	2907	1851
192.168.19.137	50760	108.175.35.142		467	5	3	3	3233090	2826	1744
192.168.19.137	50433	108.175.35.142		493	2	3	2	3233092	2814	1713
192.168.19.137	51063	108.175.35.142		536	2	3	2	3233094	2809	1716
192.168.19.137	50493	108.175.35.142		495	2	3	3	3275999	2660	1790
192.168.19.137	50841	108.175.35.142		469	2	3	3	3275999	2655	1763
192.168.19.137	50586	69.164.10.221		548	10	3	2	2693443	2550	1729
192.168.19.137	50303	69.164.10.221		469	10	3	2	2693443	2550	1687
192.168.19.137	50444	69.164.10.221		443	10	3	2	2693443	2550	1707
⋮	⋮	⋮		⋮	⋮	⋮	⋮	⋮	⋮	⋮

Future Work

Identification

Our next step is to analyze campus-wide UNC traces to better gauge the model's ability to discriminate between DASH and non-DASH traffic.

Once verified, we will incorporate the model into a program that can monitor a network link for **active DASH streams**. The monitor will be designed so that it can be easily extended by the networking community to **report** DASH streams to other devices (e.g. **OpenFlow controllers**) for routing, logging, QoS, etc.




```
admin@passive-tap ~ $ adudump /dev/dag0 | java DASHmonitor
NEW 192.168.1.101 49621 205.128.73.253
NEW 192.168.1.143 52033 8.12.222.253
END 192.168.1.101 49621 205.128.73.253
 ⋮       ⋮       ⋮       ⋮
```

DASH streams are reported as they are identified.

Simulation

We are developing a lightweight **DASH emulator** that can replicate a **Netflix-like** architecture and generate **adaptive traffic** from any number of distributed clients streaming **real DASH movies**.

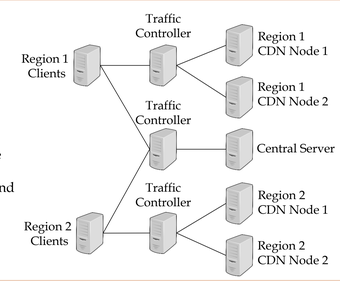
Design goals:

- The *central server* and *CDN nodes* serve all content as static files from standard web servers.
- Each movie is stored on the *central server* as a text file ordering on 7KB per hour of video.
- *CDN nodes* store a single 10-20MB dummy file from which the clients request byte ranges.

geni Exploring Networks of the Future

A Netflix-like architecture such as this could be written as a GENI **rspec** and shared among network researchers.

For more info on GENI: http://www.geni.net/



Andrew Reed, Jay Aikat

Department of Computer Science
University of North Carolina – Chapel Hill

21st IEEE International Conference on Network Protocols (ICNP), Göttingen, Germany, October 2013

Research supported by: The National Science Foundation (NSF) and the Global Environment for Network Innovations (GENI)