Media Streaming

A Benchmark for Media Streaming and Cloud Computing

USER'S MANUAL

May 27th 2013

Revision Sheet

Release No.	Date	Revision Description
Rev. 1.0	27/05/2013	Media streaming v1.0

CONTENTS

1. ***Revision Sheet………………………………………………………………..1***
2. ***Contents……………………………………………..…………………………2***
3. ***Introduction…………………………………….……………………….3***
4. ***Install the Media Streaming…………………….………………….3***

    4.1 Prerequisite Software Components……………………………………………………..3

    4.2 Installing the Client…………………………………………………………………………4

    4.3 Installing the Server………………………………………………………………………..6

5. ***Generating the Data Set……………………………..…………….8***
6. ***Running the Benchmark…………………………………..………..10***

    6.1 From the command line………………………………………………………………10

    6.2 From the GUI…………………………………………………………………………….12

7. ***Quality of Service………………………………………..…………..13***

Introduction:

The media streaming benchmark consists of two main components: A client and a server. The client component emulates real world clients; sending requests to stress a streaming server. The length and quality of videos requested by the client can be controlled by the client interface with the default setup closely following real world observations. The client system is used to send requests to a media streaming server that supports the RTSP protocol, like the Darwin Streaming Server used in our case.

Install the Media Streaming:

Prerequisite Software Components:

1. **Darwin Streaming Server**
2. **Faban** workload generator kit
3. Stream driver, RTSP client code, and operation scripts (with the benchmark package)
4. **cURL** RTSP library

Download the Media Streaming Benchmark

Installing the Client:

1. The client is built on top of the Faban workload generator. Therefore, the first step to install the client is to download Faban. You can obtain Faban from the Media Streaming Benchmark **package** or Faban's **website**.

   untar Faban in the desired directory. Call this directory: *faban-streaming*. (rename the directory as faban-streaming: mv faban faban-streaming)

2. Copy the *streaming* directory from the downloaded package to the, just created, *faban-streaming* directory
   cp -r streaming faban-streaming

   The streaming directory contains the streaming benchmark driver and the RTSP client code, in addition to the necessary scripts that will ease the deployment and run of the benchmark.

3. Make sure that the Java JDK is installed and the JAVA_HOME environment variable is properly set.
4. Start the Faban master: From Faban main directory (*faban-streaming*):
   cd faban-streaming

   ./master/bin/startup.sh

5. In the streaming directory, make the following necessary modifications to *build.properties.*
   - Change *faban.home* to the path of the main Faban directory (e.g., /opt/faban-streaming).
   - Make sure *faban.url* is set to *http://localhost:9980/*.
   - Change ant.home to the location of you ant installation.
6. From the streaming directory, run: *sh scripts/new_depoly.sh*.

This script will compile the stream driver using the default values of the benchmark. Note that this step requires the Apache Ant tool installed.

7. Prepare the RTSP client
   - The code is provided in the same streaming directory, *faban-streaming/streaming/rtspclientfinal.c*, you need to download the cURL version that works on your platform.
   - unpack the downloaded cURL distribution.
   - To install cURL, in the curl directory:
     - mkdir curlinst
     - ./configure --prefix=/opt/curl-7.24.0/curlinst/ --enable-rtsp
     - make
     - make install
   - Now, compile the RTSP client code (from the /path/to/faban-streaming/streaming)

```
gcc-I/opt/curl-7.24.0/curlinst/include   -L/opt/curl-7.24.0/curlinst/lib/
-lcurl rtspclientfinal.c -o rtspclient.o
```

   - Make a preliminary test:

```
./rtspclient.o
```

If you receive the following message:

“error while loading shared libraries: libcurl.so.4: cannot open shared object file: No such file or directory“

The way to solve the problem:

```
ln -s/opt/curl-7.24.0/curlinst/lib/libcurl.so.4 usr/lib64/libcurl.so.4
```

If the installation was done correctly, you should receive the following message: "ERROR: enter a valid URL" simply because it expects a parameter indicating where to find the video.

Installing the Server:

Here, we provide the steps necessary to install the Darwin Streaming Server on a Linux machine. Our test was carried out on Ubuntu10.10, but the instructions should apply to other distributions without problems.

1. Download and unpack the Darwin Streaming Server (on a different machine).
2. Download the following two patches:
   - **Patch one**:

     http://dss.macosforge.org/trac/raw-attachment/ticket/6/dss-6.0.3.patch

   - **Patch two:**

     http://dss.macosforge.org/trac/raw-attachment/ticket/6/dss-hh-20080728-1.patch

3. Apply the patches:

```
patch -p1 < dss-6.0.3.patch
patch -p1 < dss-hh-20080728-1.patch
```

4. Replace the Install script with this **one** **(**http://dss.macosforge.org/trac/raw-attachment/ticket/6/Install**)** (if you are not using the benchmark package) and then change its permissions:

```
chmod +x Install
```

5. The Darwin streaming server requires a qtss group and qtss user in that group to start:

```
sudo addgroup qtss
sudo adduser --system --no-create-home --ingroup qtss qtss
```

6. To scale Darwin Streaming Server beyond 1024 connections, change the #define __FD_SETSIZE parameter in the following file:

   /usr/include/bits/typesizes.h set it to 65536 (64K for example).

.

7. Build the server:

```
./Buildit
```

8. Install the server:

```
./Install
```

9. To tune the server, all the configuration parameters are found in:

   /etc/streaming/streamingserver.xml

   You need to change the following parameters:

   - *real_rtsp_timeout* to 700
   - *rtp_timeout* to 700
   - *maximum_connections* to -1 (infinite)
   - *maximum_bandwidth* to -1 (infinite)
   - *movie_folder* to the path of your movies directory
   - *run_num_threads* should be set to a value higher than the number of cores you want to utilize (e.g., 8 was enough to saturate 4 cores)
   - *overbuffer_rate* to 1.2

10. Check the limis imposed by the operating system on the maximum number of open file: (e.g., ulimit in Linux). Set this value to unlimited or to a sufficient number for the number of streams your are simulating.
11. Now the server is ready. To start the server:
    /usr/local/sbin/DarwinStreamingServer -dDS 1
    The server should print the main statistics every second!

Generating the Data Set:

The streaming benchmark dataset consists of two main folders. The first, called streamingVideos_10, is a 1.6 GB folder and includes all the following video categories and lengths:

Duration: 1 minute to 10 minutes at the increments of 1 minute

Rates:

Dial-up: 42 Kbps

Low: 102 Kbps

Medium: 290 Kbps

Hi: 790 Kbps

LAN: 1500 Kbps

The second directory, called stressVideos, consists of three videos: 5-minute Dialup, 5-minute Medium and 5-minute Hi. The intent of these videos is to emulate real world situations where a small subset of the videos gets accessed most of the time.

To generate a large data set, you need to replicate the streamingVideos_10 folder the number of times until the total dataset is sufficient for you. The replication process should use the same name but changing the number at end (streamingVideos_11, streamingVideos_12, etc.) and the numbers should be contiguous with no gaps.

In the case of the Darwin streaming server, the data set should be under the */usr/local/movies*directory in the server machine. Otherwise, you can point the server to use your directory of choice by editing */etc/streaming/streamingserver.xml* and modifying the corresponding option.

The benchmark comes with a simple script to help the process of dataset generation. To generate the dataset under the */usr/local/movies* directory:

- Copy the *streamingVideos_10* directory to */usr/local/movies* directory.
- sh copy_streaming.sh X, where X is the number of directories you want to generate.

  Note: This process can be time consuming depending on the size of the generated data set.

Running the Benchmark:

Before running the benchmark you should do the following:

1. export JAVA_HOME=/usr/lib/jvm/java-6-sun (or point to the JDK on your machine)
2. Make sure to start the master process: From the /opt/*faban-streaming* directory:
   sh master/bin/startup.sh

From the command line:

The benchmark comes with a set of scripts to ease running the benchmark. The scripts necessary to run the benchmark are located under the *faban-streaming/streaming* (client directory) directory under the directory *scripts*.
The main script in this folder is *run-test.sh*. This script generates 10 Java processes, called agents, each of which will generate an equal number of requests to the server machine. For example, if the total number of requests is 100, each agent will generate 10 requests.
The *run-test.sh* script by itself reads the default configuration found in */opt/faban-streaming/streaming/deploy/run.xml* and starts generating requests based on the settings in that file.
To use this script, first you need to set the correct paths as follows, you need to be in*/opt/faban-streaming/streaming* and:

- *vim scripts/run-test.sh* and replace all the */home/username/* to point to the path to /opt/*faban-streaming*
- Modify the deploy/run.xml to set mainly the address of the server, code directory and output directory. A full description of the parameters is provided in the next paragraph.
- *sh scripts/run-test.sh*

The following is a description of the main parameters in the run.xml file that you may need to modify to customize your run:

- Host IP address: specifies the address of the machine where the client agents will be running. This is set to *localhost* by default.
- The scale: defines the number of clients simultaneously simulated.
- The ramp up, steady state and ramp down time.
  Please note that the benchmark is designed to ramp up the clients based on the number of such emulated clients. Therefore, the ramp up time is not critical but should be greater than or equal to 5 seconds. The

important parameter for a successful run is the ramp down time. This value should be set to *2* duration of longest video* you are willing to use.

- The number of agents: how many agent processes to control the total number of clients.
  Please note that this number affects the speed of preparing the client threads. Along with this parameter, you should change the *run-test.sh* script by including separate commands for all the agents. For example if you want to use 15 agents instead of 10, you should add 10 extra lines in the *run-test.sh*.
- Whether to start the agents concurrently or not: selecting yes starts all the agents at the same time, each creating the number of clients assigned to that agent.
- The mix of videos (duration and bitrate) you want your clients to use. For example, to stream only short-duration and high bitrate videos you need to set the GetShortHi parameter to 100 and the rest to zero. If you want the requested videos to be 30% short and high and 70% medium duration and low bitrate, you should specify the GetShortHi to 30 and the GetMediumLow to 70 and the rest to zeros, and so on.
- IP Address of the server
- The port number the streaming server uses to listen to client requests. Darwin Streaming Server listens to 4 ports by default, one of them is 554 (specified by default). You can check and set other ports by looking at (and editing) */etc/streaming/streamingserver.xml*in the server machine. You can also force the server to listen to a certain port by using the*-p* option in the Linux command line when you start the server. For example:
  /usr/local/sbin/DarwinStreamingServer -dDS 1 -p 3330 to listen to port 3330
- Data set scale: corresponds to the number of video directories that were generated (as part of the data set). If you want to generate ~3 GB of video data set you need to use two directories. To ensure that your requests actually request videos from both directories, you should specify this number as 2.

  Please note that this number has no effect on the requests to *stressVideos*. For example, if you specify in the request mix that you need 100% of requests to *stressHi*, the dataset scale becomes irrelevant and will not be used, however, it will still be used for all other requests if their percentage in the mix is greater than 0.

- Fix duration: is an optional parameter that is provided to give more control over the user run. It limits the duration of each requested video to this number. For example, if you want to use a mix of videos with different durations and bit rates but want to test only 1 minute of each video, you should specify this number as 1. A negative number means ignoring this parameter (by default it is -1 meaning that it is ignored).

- Protocol code: is the directory where the RTSP client resides. The default case is under*/path/to/faban-streaming/streaming*.

To facilitate the configuration process, the benchmark comes with few scripts to allow the rapid change and run of basic configurations. You can use the *start-run.sh*script in the */path/to/faban-streaming/streaming/scripts* as follows:

From */path/to/faban-streaming/streaming*

sh scripts/start-run.sh <NUM-CLIENTS> <MIX> <SERVER-IP> <DATASET-SCALE>

where:

- NUM-CLIENTS: is the same as scale (number of simultaneous clients).
- SERVER-IP: is the IP address of the server.
- DATASET-SCALE: the same as the data set scale described earlier.
- MIX:
  -dist: a selected mix of durations and bitrates.
  -shorthi = 100% short duration high bitrate.
  Similarly, you can use: shortmedium, shortlow, longlow, longmedium, longhi, mediumlow,mediummedium, medium hi, stresshi, stressmedium, stresslow

From the GUI:

1. To run the benchmark from the the gui, start by deploying the benchmark: open*http://localhost:9980/deploy* in a browser.

   Username: deployer and password is: adminadmin
   Then select *$FABAN_HOME/benchmarks/streaming/stream.jar* and click on *deploy*. You should see a message that says that the benchmark was deployed successfully!

2. Open *http://localhost:9980/* in a browser. You will see three main tabs: The first defines the java parameters, the second defines the main client parameters and the third defines the server parameters. Please refer to the previous section for a definition of the parameters.

3. After selecting the proper configuration, start the benchmark run by clicking the "ok" button.

Quality of Service:

Measuring quality of service is challenging for the media streaming applications. Relying on the time required to download the video is not a good indicator of quality because the download time doesn't indicate whether the video was not disturbed (potentially several times) during the play time. Among several choices to measure quality, we found that monitoring the server displayed statistics is the most accurate method. To guarantee that the videos are served with quality, make sure to run the server with a granularity of 1 second statistics display:

```
/usr/local/sbin/DarwinStreamingServer -dDS 1
```

And make sure that the AvgDelay field is always negative.