**Chromium:**

1) First, you'll need to get your hands on Google's repo cloning tool:

a.       Run this command:*git clone* [*https://chromium.googlesource.com/chromium/tools/depot\_tools.git*](https://chromium.googlesource.com/chromium/tools/depot_tools.git)

b.      Add depot\_tools to your PATH

c.       Run this command:*gitconfig --global core.deltaBaseCacheLimit 2G*

2) Download my version of Chromium

a.       Run this command to switch to home dir: *cd ~*

b.      Run this command: *gclientconfig --spec 'solutions = [{u'"'"'managed'"'"': True, u'"'"'name'"'"': u'"'"'src'"'"', u'"'"'url'"'"': u'"'"'*[*https://github.com/devasia1000/chromium.git*](https://github.com/devasia1000/chromium.git)*'"'"', u'"'"'custom\_deps'"'"': {}, u'"'"'deps\_file'"'"': u'"'"'.DEPS.git'"'"', u'"'"'safesync\_url'"'"': u'"'"''"'"'}]*

c.       Run this command: *gclient sync* – will take 1 hour or more

3) Detecting and logging bit-rate switches

a.       Switch to *bitrate\_detector* branch

b.      Run *build\_chromium.sh* – this will take a long time

c.       Start Chromium: *perl*[*start\_test.pl*](http://start_test.pl/)*{youtubeVideoLink} > {outputFile}*

4) Detecting and logging loading times and stall durations/locations

a.       Switch to *stall\_detection* branch

b.      Run *build\_chromium.sh* – this will take a long time

c.       Start Chromium: *perl*[*start\_test.pl*](http://start_test.pl/)*{youtubeVideoLink} > {outputFile}*

***youtube\_playback***

**Cloning GET requests:**

**Pre-reqs:**

- 2 physically separate machines (I'm going to refer to one machine as the 'client' and the other as the 'server')

- youtube\_recorder from <https://github.com/devasia1000/youtube_recorder>

- youtube\_playback from <https://github.com/devasia1000/youtube_playback>

**Steps**:

1) Setup a 'client' machine and 'server' machine on the same network

2) On server, run: cd ~ && git clone <https://github.com/devasia1000/youtube_recorder> && cd youtube\_recorder&& make &&sudo service apache2 stop &&sudo ./pretender

3) On client, redirect all hostnames to server's IP address by adding address=/#/{serverIPAddress} to your DNSMASQ config file

4) On client, restart DNSMASQ

5) On client, open a YouTube video of your choice (make sure it is HTML5) with Google Chrome. You should see the GET requests being printed out on the server machine, if you don't see anything, then you probably don't have DNSMASQ setup correctly.

6) On client, manually switch through all available video streams while the video is playing

7) On server, kill youtube\_recorder

8) On server, run: python req\_res\_mapping.py

**Cloning the video streams**

**Steps:**

NOTE: All steps to be done on server

1) Open the same YouTube video with Google Chrome- make sure the video is playing in HTML5

2) Press 'F12' to bring up Chromium's webpage debugger, click on 'Network' - you should see all the GET requests

3) While the video is playing, manually change the quality level

4) Wait a few seconds for the video to change quality

5) Once you've noticed a change in quality, go to the network debugger and click on the most recent 'videoplayback' in the left hand panel. The right hand panel of the debugger should change.

6)  Go to the right hand panel of the debugger and click on 'Headers' and take note of the MIME type and CLEN headers. CLEN will be an integer and MIME could be 'audio/mp4' for the audio stream and 'video/mp4' for a video stream. To download the stream, copy the request URL.

7) Paste the link in the URL bar of a new tab, change the initial range parameter to 0 and the final range parameter to an arbitrarily large number. Press enter

9) Right click on the video player that pops up and click on 'Save As'

10) Save your video to ~/media/{clen}-{mime}. Eg: if the video has MIME='video/mp4' and CLEN='7923837', then you must save it to ~/media/7923837-video

11) Repeat above steps for all video and audio streams

**Replaying a YouTube video locally**

**Pre-reqs:**

- 2 physically separate machines (I'm going to refer to one machine as the 'client' and the other as the 'server')

- youtube\_recorder from <https://github.com/devasia1000/youtube_recorder>

- youtube\_playback from <https://github.com/devasia1000/youtube_playback>

**Steps**:

1) Setup a 'client' machine and 'server' machine on the same network

2) On server, run: cd ~ && git clone <https://github.com/devasia1000/youtube_playback> && cd youtube\_playback&&javac \*.java &&sudo service apache2 stop &&sudo java Main

3) On client, redirect all hostnames to server's IP address by adding address=/#/{serverIPAddress} to your DNSMASQ config file

4) On client, restart DNSMASQ

5) On client, open the same YouTube video with Google Chrome.

6) The YouTube video should play locally! :)

**Cellsim**

**Prereqs:**

- Finish debugging the 1080p and 720p versions of video, get it working perfectly before moving on

- Change your username to 'chromium'

**Steps**:

1. cd ~ && git clone <https://github.com/devasia1000> && cd cellsim
2. Add ‘address=/#/{mininetServerIP}’ to /etc/dnsmansq.conf

3) Extend the uplink trace files: cat {networkName}-uplink\_processed.txt | perl [trace\_extender.pl](http://trace_extender.pl/) > {networkName}-uplink\_extended.txt

4) Extend the downlink trace files: cat {networkName}-downlink\_processed.txt | perl [trace\_extender.pl](http://trace_extender.pl/) > {networkName}-downlink\_extended.txt

3) Run Chromium within Cellsim: sudo python cellsim\_with\_youtube\_playback.py {username} {uplinkTraceFile} {downlinkTraceFile} {lossRate} {youtubeLink}