LINFO2142 - Group G - LibreSpeedQuic - Second Draft

> Giot Adrien & Gaudin Félix

Preambule

Librespeed provides an open-source speedtest like server that runs above TCP. The objective of this project is to deploy this test suite on a QUIC server (e.g. nginx using cloudflare's quiche) and explore the performance of QUIC when doing such speedtests.

In this report, we will make reference to our GitHub repository.

Things modified

Front-end

- Change display to look more like cloudflare
- A warning in case http2 is used for the test
- Allow the user to choose the number of connections (upload and download)
- Add field to the request for telemetry (add http version, number of connections and the browser)
- A dashboard

Back-end

For the backend we inspired ourselves of the node js backend of librespeed (https://github.com/librespeed/speedtest/tree/node) but we added :

- Telemetry
- A getter for the database (used in dashboard.html)

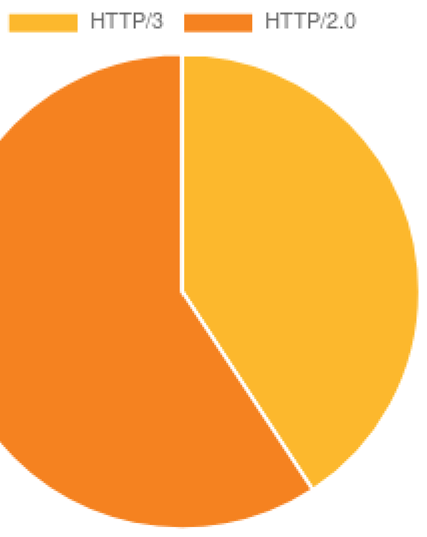
And we dockerized the JS back-end

We also made a route direcly in nginx to get the http version of the user.

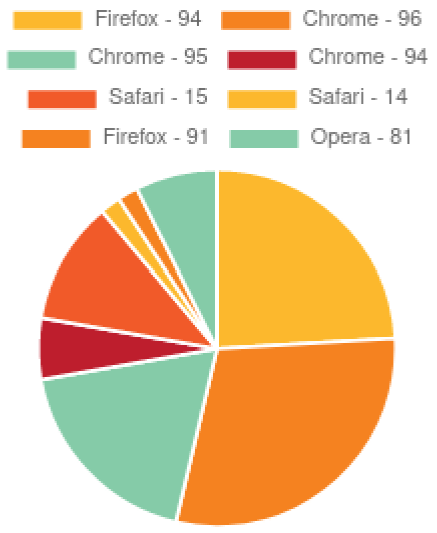
Current results

We allowed people to use the site to perform speedtest (linfo2142-grpg.info.ucl.ac.be/).

Depending on where the site is hosted, the speedtest is performed using HTTP/3 or HTTP/2. We summarize this with the following chart :

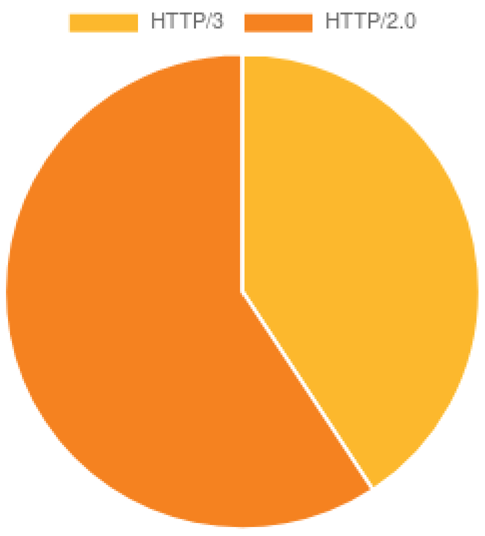


We see that the protocol with which the speedtest is performed highly depends ont the navigator used :



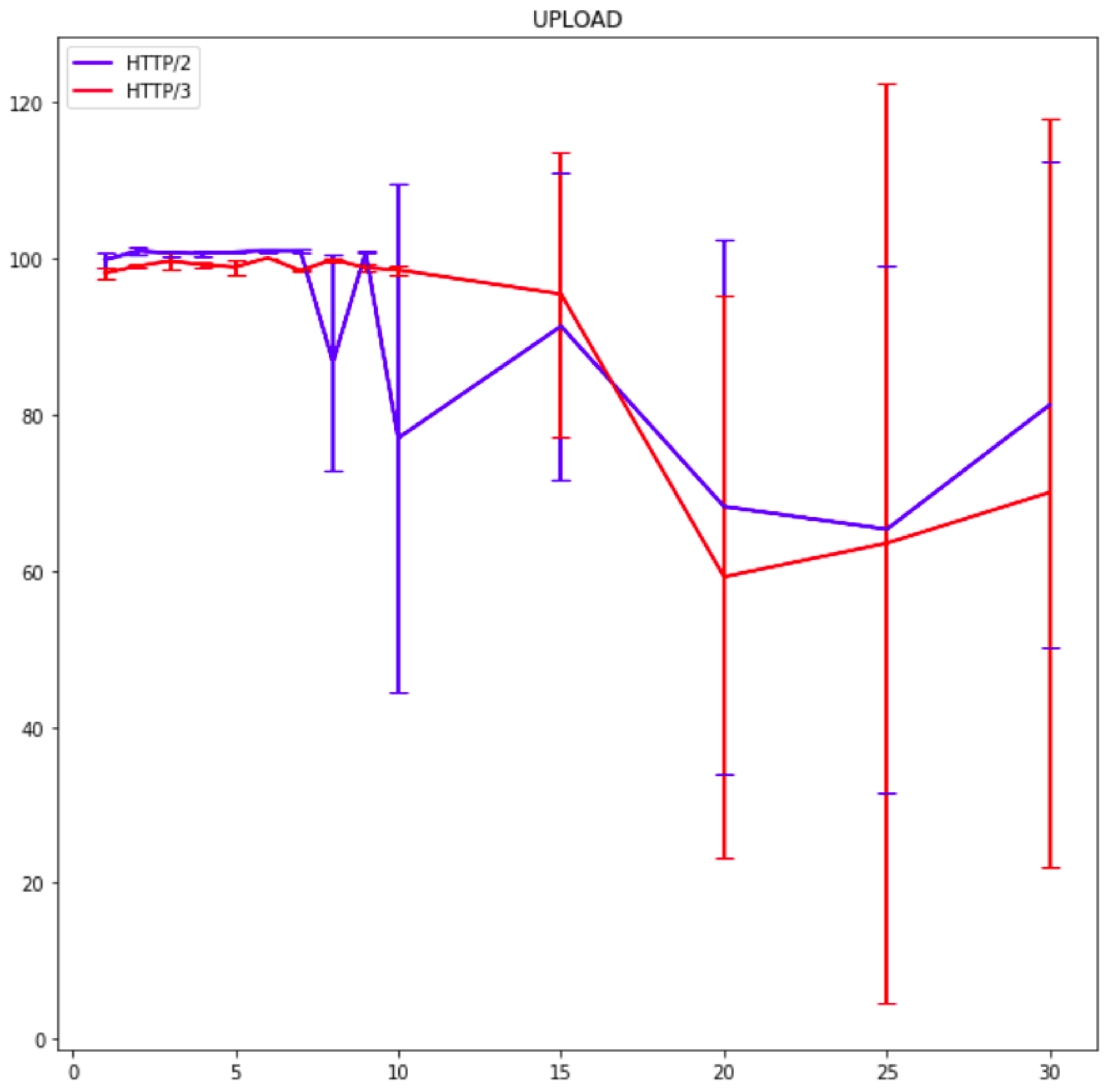
browser	http2 number	http3 number
Safari - 15	3	3
Safari - 14	1	0
Opera - 81	3	1
Firefox - 94	3	10
Firefox - 91	1	0
Chrome - 96	13	4
Chrome - 95	9	4
Chrome - 94	0	3

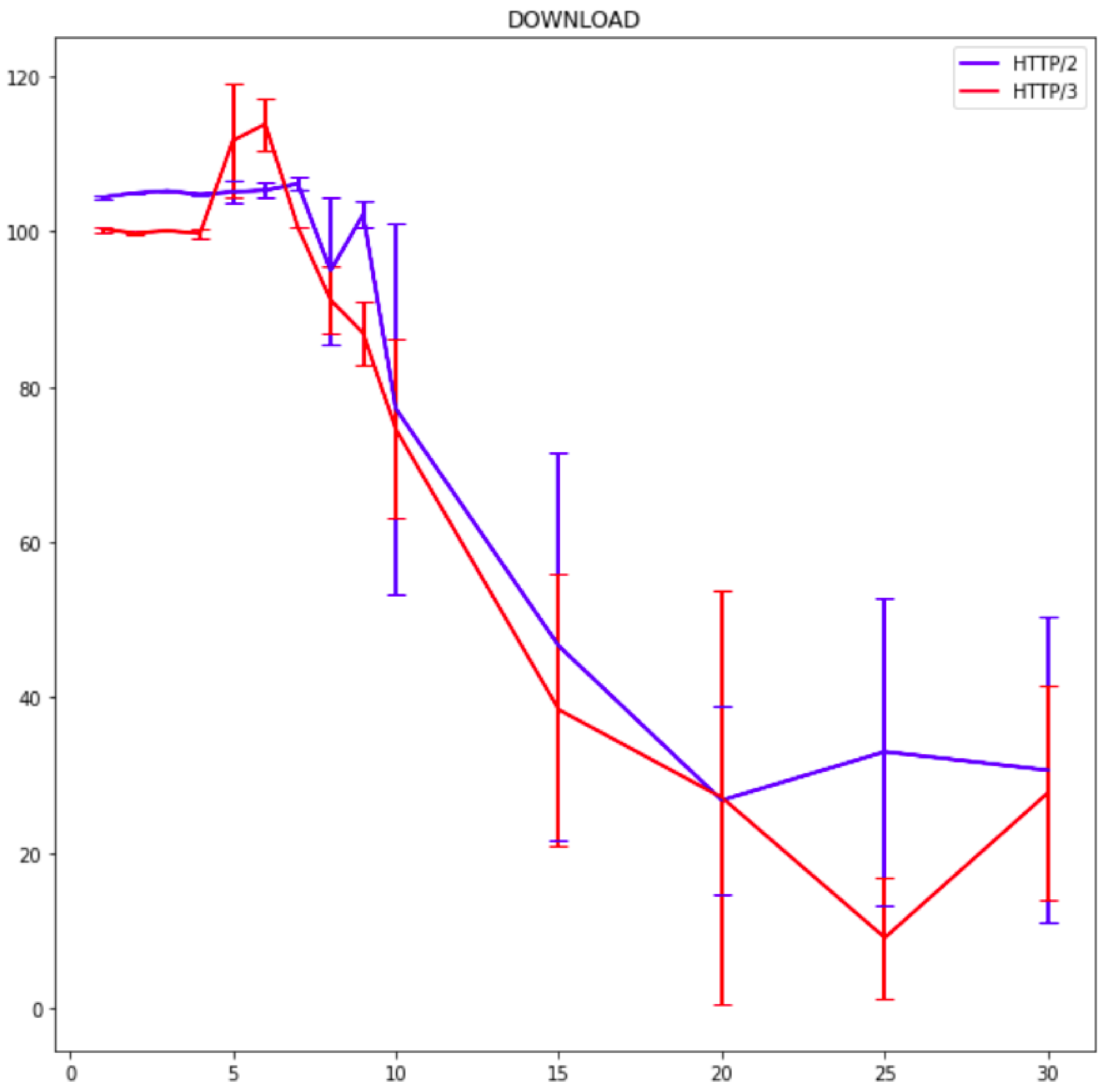
We get the final repartition between HTTP2 and HTTP3 :



We can see that despite that the config is done correctly (the alt-svc header is added to the HTTP responses), HTTP/3 is often not used.

In order to compare the performances of HTTP2 and HTTP3, we performed multiple tests using a machine of the intel room, while varying the number of connections. We get the following results :


UPLOAD
HTTP/2
HTTP/3
0
20
40
60
80
100
120
0
5
10
15
20
25
30



Note : To disable HTTP/3 (in firefox) we set `network.http.http3.enabled` to false in `about:config`

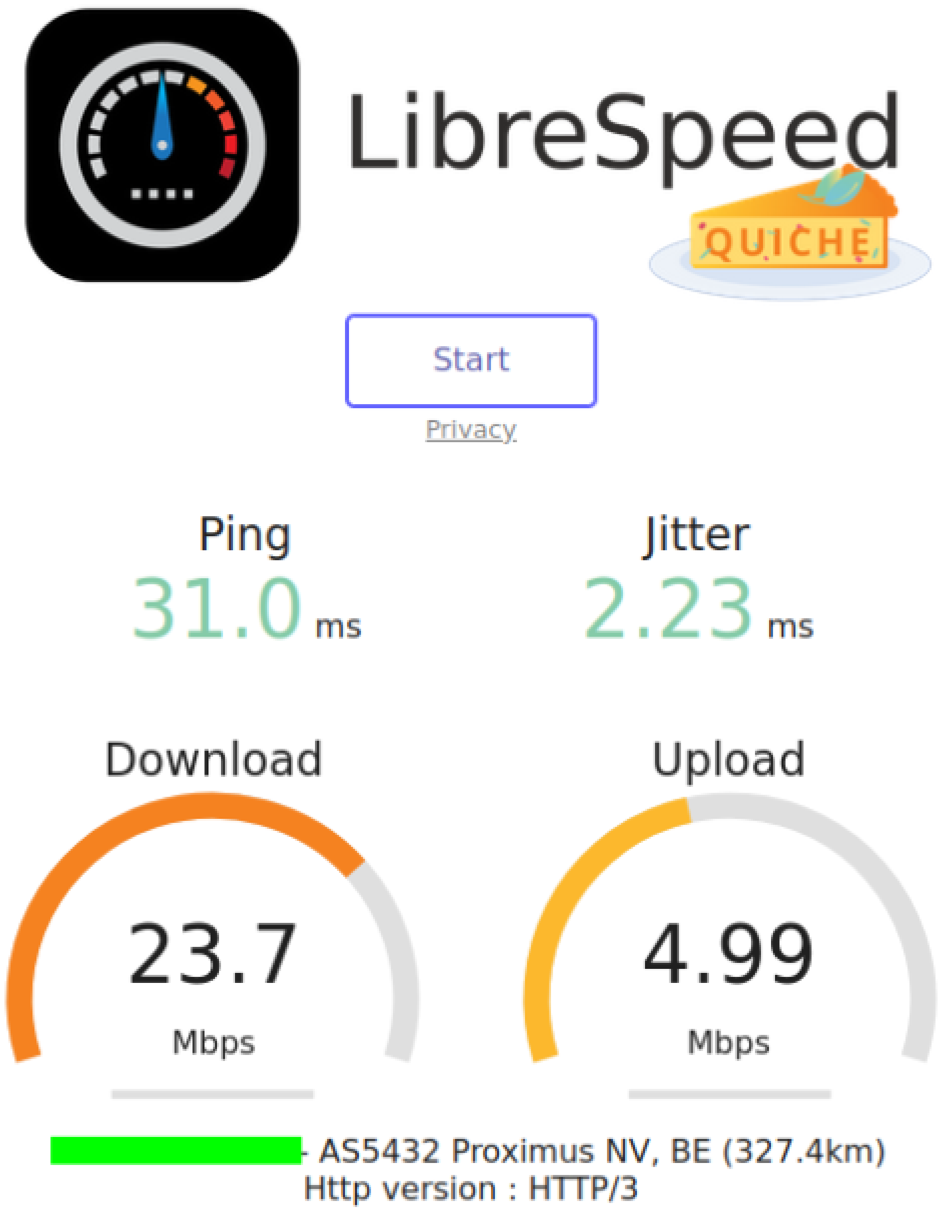
Issues encountered when deploying the site on the ingi VM

We first hosted the site on a personal computer. When we enabled HTTP/3 on it, we tried to move to the VM lent to us. After a bunch of mails to the system administrator, we were able to perform the speedtest using HTTP/3, even with a client supporting it while putting an alt-svc header in the server responses.

While looking for a way of forcing the switch to HTTP/3 on the VM, we discovered that the possibility of using DNS records to inform of the use of HTTP/3 was being studied by Google and Akamai (see cloudflare blog and IETF draft). We could not use that since it is not deployed yet.

Using a SSL certificate that was not autogenerated did not help either.

We also made tests using the OVH server of a friend of ours. They were quite successfull, since the speedtest was done using HTTP/3, and the results were consistent.



Ping
31.0 ms

Jitter
2.23 ms

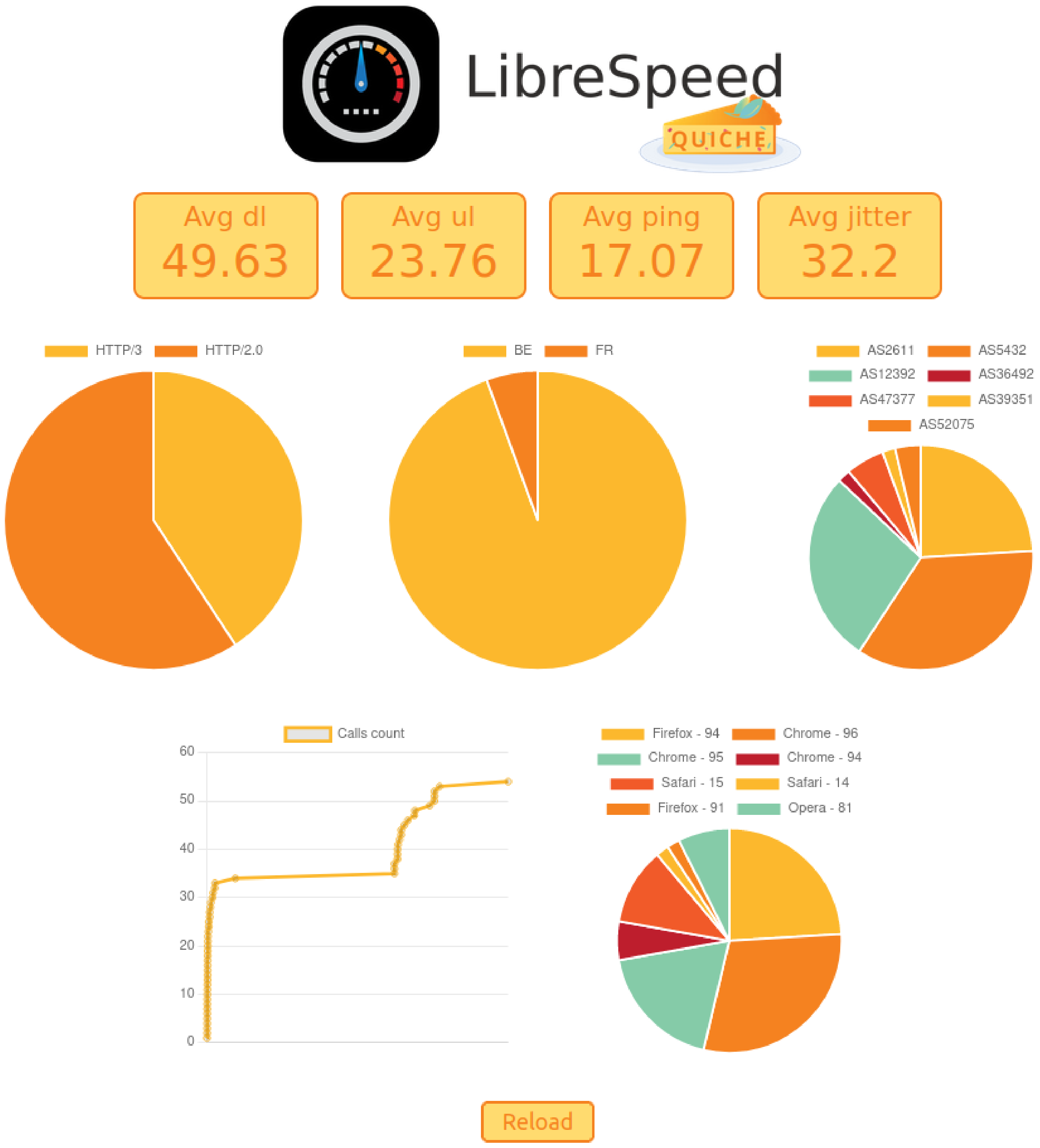
Download
23.7
Mbps

Upload
4.99
Mbps

AS5432 Proximus NV, BE (327.4km)
Http version : HTTP/3

We finally discovered that the firewall forbid the use of the port 443 in UDP, which explained why the speedtest was not performed using HTTP3. After the system administrator fixed it, the speedtest could finally be performed using HTTP3. We then linked the site to students so that they could test it.

Dashboard



Avg dl 49.63

Avg ul 23.76

Avg ping 17.07

Avg jitter 32.2

Thanks

- Nicolas Beuel for helping us to make a SSL certificate on the personal computer (allowing us to perform tests with several people).
- Guillaume Vandenneucker for lending his OVH server to test from a *real* server.
- Thomas Wirtgen for helping us in general and particulary to make a SSH bridge to test the webside (when the ports weren't opened)
- Olivier Bonaventure who gave us some food for thought on the problem (from the VM the webside is in HTTP/2)
- Chipie for the support

Appendix

Installation

quiche can be integrated into nginx using an unofficial patch to provide support for HTTP/3 (see their Github).

Nginx with quiche

You can use the Makefile to install the modified version of nginx (`make nginx`)

The patch only works with the 1.16.x release branch (the latest stable release being 1.16.1).

1. Install Rust

We must have a version of rust >=1.53 (use `rustc -V` to know the version).

If you don't have it juste use :

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

Than you have to add the `cargo` binaries to your binary files :

```
sudo ln -s ~/.cargo/bin/cargo /bin/
```

2. Install PCRE

This will be used to configure nginx :

```
sudo apt-get install libpcre3 libpcre3-dev
```

3. Download nginx with the right version (1.16.1)

You may have to remove nginx if you have another version installed

```
curl -O https://nginx.org/download/nginx-1.16.1.tar.gz

tar xzvf nginx-1.16.1.tar.gz
```

4. Clone the quiche repository

```
git clone --recursive https://github.com/cloudflare/quiche
```

5. Apply the quiche patch to nginx

```
cd nginx-1.16.1

patch -p01 < ../quiche/extras/nginx/nginx-1.16.patch
```

6. Configure nginx

You can add more options. All options are available here : https://nginx.org/en/docs/configure.html

```
./configure \
--prefix=$PWD \

--build="quiche-$(git --git-dir=../quiche/.git rev-parse --short HEAD)" \

--with-http_ssl_module \

--with-http_v2_module \

--with-http_v3_module \

--with-openssl=../quiche/deps/boringssl \

--with-quiche=../quiche \

--with-pcre
```

7. Install the modified version

You can try :

```
make && make install
```

If you get an error with `cp conf/koi-win ...` while running the `make install` you have to comment the line and the three below (`cp conf/koi-win ...`, `cp conf/koi-utf ...` and `cp conf/win-utf ...`) and retry the commands.

8. Test it !

You can use `./sbin/nginx` to launch the server (and `killall nginx` to killit). By default the page url is at `localhost` and the default port is 80. (http://localhost:80)

LibreSpeed

We will use the librespeed project to perform the speedtest. If you want to know more about it, please look at the project GitHub https://github.com/librespeed/speedtest .

To perform more tests we modified a bit the original files. They are in the folder `frontend`. To use them, you can just redirect the nginx root folder to it.

Build the backend

Files

- `main.js` receives all the possible requests (GET and POST). Will listen on port 8888 by default. You can change it at the beginning of the file.

- `helper.js` computes the distance between the server and the client.
- `database.js` includes the SQLite3 database for telemetry.
- `secret.js` The file where you store your login and your password in order to acces to the dashboard. The file should look like this :

```
const  LOGIN = "Your Login";

const  PASSWORD = "Your password";


module.exports = {

LOGIN :  LOGIN,

PASSWORD :  PASSWORD

}
```

How to launch

Native

Just run `node main.js`. (You may have to install all the dependencies)

Docker

We dockerized the backend. You may run it using these commands (while being in the backend directory):

```
docker build . -t <img name>

docker run -p 8888:8888 -d <img name>
```

Modify the Nginx config

In order to enable HTTP/3 in nginx you have to modify the config file. It's located in `nginx-1.16.1/conf/nginx.conf`.

We made an example file nginx_example.conf.

We will now describe its key elements.

This line enables quic :

```
listen 443 quic reuseport;
```

Also don't forget to enable TLSv1.3 otherwise quic won't work (`ssl_protocols TLSv1.3;`)

This line adds a header which informs the client of the possibility to switch to QUIC/HTTP3.

```
add_header alt-svc 'h3-27=":443"; ma=86400, h3-28=":443"; ma=86400, h3-29=":443"; ma=86400, h3=":443"; ma=86400';
```

Case of 403 : Forbiden error

This may mean that your html files are in a folder with to strict permissions. First try this https://linuxhint.com/fix-nginx-403-forbidden/. It may not work, so just put the html files in a *normal* file (eg : /home/) and then use this https://stackoverflow.com/questions/10631933/nginx-static-file-serving-confusion-with-root-alias