How Effective is cFosSpeed Traffic Shaping?

Christoph Lüders & Christian Carazo,
cFos Software GmbH

November 16, 2018

The effect of cFosSpeed Traffic Shaping was tested on a VDSL-100 connection by measuring ping times while an upload or a download, respectively, were running. The average times with and without cFosSpeed were compared and lead to the conclusion that cFosSpeed Traffic Shaping can keep the delay from data transfers 3 to 10 times lower than without.

1 Introduction

Traffic Shaping is a technique to reorder Internet data packets in such a way that urgent traffic is transferred first and the rest of the data later. For example, data packets of a VoIP application should be transferred quicker (i.e., with less latency) than packets of an up- or download. Usually, it's not important if the download is ready 1 second earlier or later, whereas a VoIP connection with a delay of 0.5 seconds is almost unusable.

To that end, cFosSpeed was created. It is a software for the Windows operating system and is in widespread use since 2004. There exist some other programs that have a built-in capability to reduce their data bandwidth so as not to interfere with other applications, but cFosSpeed is the only fully automatic software solution in the market today that can be used for all programs on Windows.

This report assesses how well cFosSpeed works in today's Internet with a modern, fast Internet connection. One could think that modern Internet connections with their high bandwidth have made cFosSpeed superfluous. We will show that this is not the case.

Even though the authors work for cFos Software GmbH that develops and sells cFosSpeed, we aim at maximum clarity in this report. You should be able to reproduce all the effects at home, so we provide a lot of detail.

2 Test setup

The tests were done on a VDSL line in Germany, namely a VDSL-100 line by Deutsche Telekom. The router used was a Speedport W724V (Typ C), firmware 05011603.06.001. At the time of the test (25.10.2018) it showed a "DSL Downstream" speed of 109341 kbit/s and a "DSL Upstream" speed of 40143 kbit/s. This router model has no internal prioritization support. The test machine was connected to the router by Gigabit Ethernet LAN.

The test was done on a Windows 10 RS4 machine (Microsoft Windows 10 Pro, build 10.0.17134) with 64 bit. cFosSpeed version was 10.27.2330, downloadable from https://www.cfos.de/en/index.htm. cFosSpeed was in "Favor Ping Time" mode and set to "cooperative", signifying there are no other non-cFosSpeed clients in the LAN. The "medium" was set to "VDSL".

For transfers we used Firefox, version 63.0 (64-bit). For the download tests we downloaded the 1 GB file (incompressible data) from https://www.thinkbroadband.com/download. For upload, we uploaded the 512 MB file from the same source to http://www.cfos.de/en/upload/index.htm. This page allows cFos users to upload files for diagnostic purposes. The transfers were canceled after about 60 seconds.

To measure responsiveness, we used the hrPing utility, version 5.07.1148, from https://www.cfos.de/en/ping/ping.htm. Its main advantage is that it measures with microsecond accuracy. Furthermore, it allows UDP pings and measures the RTTs of ICMP error messages as well.

The exact command used was:

```
hrping heise.de -t -y -g -s200 -u -l0 -i3 -F
values.txt.
```

This will use heise.de as target, ping continuously (`-t`), print a summary to the screen (`-y`), show the results graphically (-g), leave 200 ms delay between successive pings (-s200), send UDP packets (-u) with 0 bytes payload (-l0) and TTL 3 (-i3) and additionally write all output to a file (-F values.txt).

We used the UDP ping option, since some routers prioritize ping packets, thus preventing the use of them to measure line congestion. Since we set TTL to 3, the packets are bounced at hop 3; this is already on the ISP side of our DSL line, thus it measures the bottleneck and *only the bottleneck* of the whole path to almost any destination. Setting a higher TTL value will increase the inaccuracy of the measurement. Furthermore, replies to the UDP ping by the host are filtered by firewalls, thus we need to use -i with -u to make the measurement work. Since we use TTL 3, ping packets will not reach the host heise.de, so we could have used almost any host that is not in the 3 hop range.

In the following, we compare the ping time *increase* that a transfer (up- or download) causes, compared to the base ping, i.e., the ping time without any load on the line. This is more useful than comparing raw ping times, since the raw ping time almost entirely depends on the ping time to the ISP (usually, the first hop after the router) and is thus quite arbitrary.

The raw ping time can not be changed anyway, so the second best thing anyone can do is keep the increase from transfers small.

2.1 Rolling your own

If you want to replicate the download tests, be sure to choose a sending host that is sending fast enough. It must be more than able to fully utilize your connection, since you want to simulate congestion, i.e., sending of *too much* data.

The same holds for the upload tests. Make sure your machine sends fast enough and the receiver is able to handle data at that speed.

In both cases, hosts near to you are usually better than hosts on other continents. A significantly elevated ping time while transferring (without the use of cFosSpeed) indicates that the test setup is good.

As ping target, make sure the host is on “the Internet”, i.e., no longer in your LAN, but on the ISP’s side of your connection. Even so, one can select a TTL value that is too small and doesn’t show any delay when the line is under full load. A larger TTL solves this issue.

Be sure to use the latest version of cFosSpeed, since updates are released often.

Disable all other traffic through your router and don’t forget to switch off your WLAN.

3 Measurements

We first measured the base ping time (ping time without any traffic) for 60 seconds, then we downloaded data for 60 seconds, waited some, then we uploaded data for the same time.

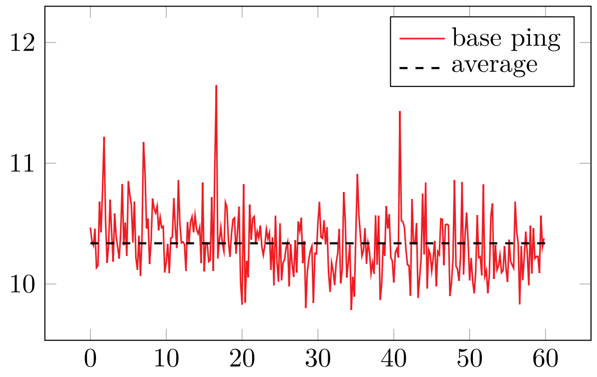
This test was done once without cFosSpeed installed and once with.

3.1 Without cFosSpeed

For the first series of tests, cFosSpeed was not installed.

3.1.1 Base ping measurement

The base ping time was measured. The resulting ping time plot looks like this:

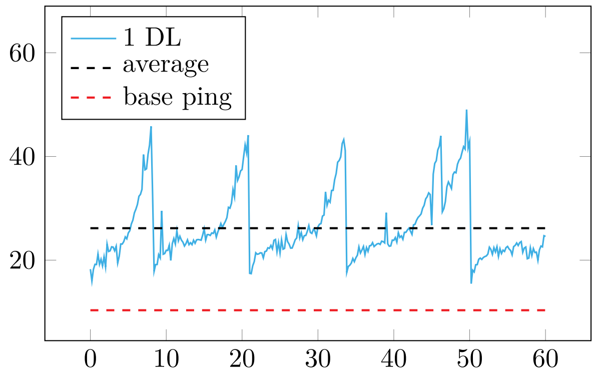


The ping time statistics are: min=8.71, average=9.233, max=10.148, stddev=0.24624, median=9.216.

There was some fluctuation, but almost all measurements lie within 1 ms. That is accurate enough for us.

3.1.2 Download test

Next, we started a download while we still measured ping times. The ping time plot looks like this:



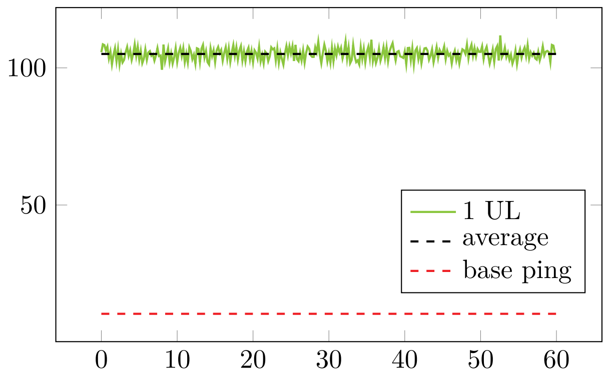
The ping time statistics are: min=15.486, average=26.162, max=49.045, stddev=6.6275, median=23.841.

The average increase from the base ping was about 16 ms.

One can see nicely the increase in congestion by the sender until (presumably) some packet loss happens and the congestion window is lowered again. One can even guess that the sender uses TCP CUBIC, since parts of the plot have a fairly cubic ($f(x) = x^3$) appearance.

3.1.3 Upload test

Following, we started an upload and measured the ping time while it ran. The plot looks like this:



The ping time statistics are: min=99.338, average=105.07, max=111.83, stddev=2.1675, median=105.22.

The average increase from the base ping was about 95 ms.

The test machine kept a high level of “pressure” on the line all the time, resulting in high congestion.

3.2 With cFosSpeed activated

For the second series of tests, cFosSpeed was installed with Traffic Shaping activated and “Favor Ping Time” mode enabled.

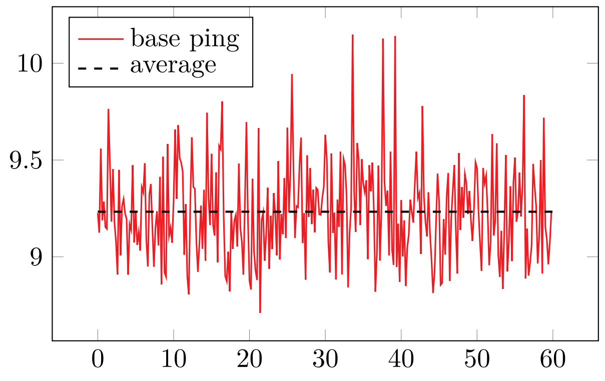
3.2.1 Base ping

The base ping time was measured.

The ping time statistics are: min=8.71, average=9.233, max=10.148, stddev=0.24624, median=9.216.

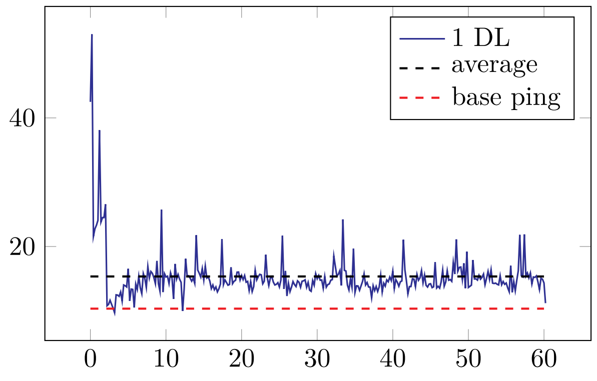
The average of 9.233 ms is not totally the same as without cFosSpeed (10.337 ms), but this is within the normal fluctuations that we experienced. Even in these one minute snippets one can see certain fluctuations in the moving average.

This is how the plot looks like:



3.2.2 Download test

Then we did the download test and this is how the resulting plot looks like:



The ping time statistics are: min=9.713, average=15.343, max=53.004, stddev=3.849, median=14.615.

The average increase from the base ping was only about 5 ms.

It took less than 2 seconds until cFosSpeed had tamed the TCP stream. After that, the ping time stayed nice and low.

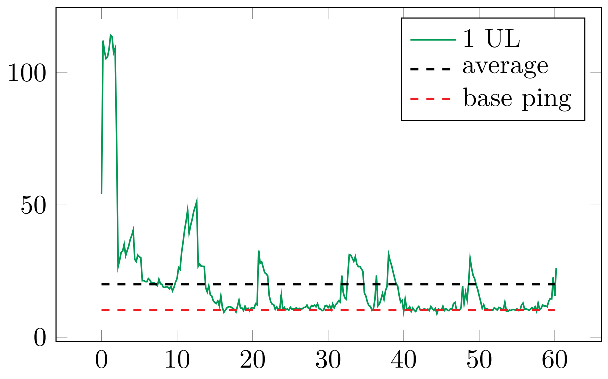
3.2.3 Upload test

Next came the upload test.

The ping time statistics are: min=8.888, average=19.999, max=114.14, stddev=18.29, median=12.791.

The average increase from the base ping was only about 10 ms.

The plot of ping times looks like this:



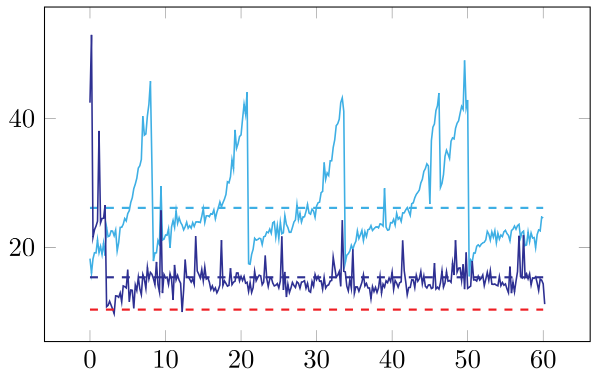
Within the first 2 seconds, cFosSpeed adapted to the high upload bandwidth and could afterwards keep the congestion at bay, leading to an average ping time of even 16.929 ms (starting from second 2).

4 Comparison

To provide a visual comparison, we overlay the plots. The light colored plots are without cFosSpeed, the darker colored ones are with cFosSpeed. The dashed lines are the averages.

4.1 Download

These are the plots of the downloads:



The ping times were much lower with cFosSpeed installed, specifically, there were no more high outliers; the regular sawtooth pattern had disappeared and a reasonably even distribution of low ping times was achieved.

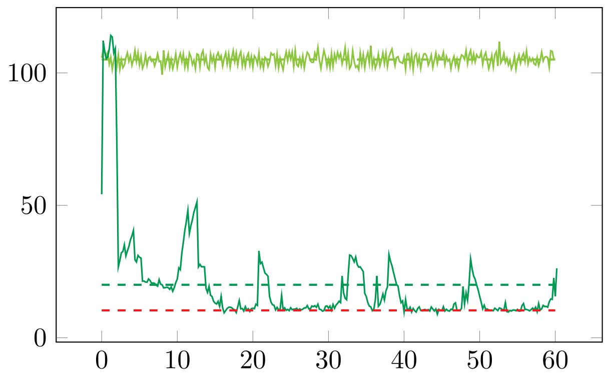
The following table shows the numbers side by side.

Scenario	Ping in ms	Increase in ms
Download, no TS	26.162	15.825
Download with TS	15.343	5.006

Dividing the increase without cFosSpeed by the one with cFosSpeed leads to a factor of 3.2. That is, with cFosSpeed installed, a download slows down the connection 3.2 times less!

4.2 Upload

This is how the upload plots look like:



It's much more obvious in the upload case: here the ping time was always high if cFosSpeed was not used. On the other hand, if cFosSpeed was active, the ping times were very low, oftentimes almost as low as the base ping.

Let's compare the numbers:

Scenario	Ping in ms	Increase in ms
Upload, no TS	105.070	94.733
Upload with TS	19.999	9.662

Dividing the average increased ping time without cFosSpeed by the one where cFosSpeed was active yields an impressive factor of 9.8, meaning the upload caused almost 10 times less delay when cFosSpeed was installed!

5 Conclusion

We took ping times while a download respectively an upload was in progress. We compared times taken with and without cFosSpeed Traffic Shaping on a VDSL-100 connection (109341 kbit/s down, 40143 kbit/s up).

cFosSpeed Traffic Shaping could lower the ping times considerably. When downloading, cFosSpeed kept the additional delay 3.2 times lower. When uploading, the delay was even 9.8 times less!

This means that even today with high-speed Internet connections, cFosSpeed can greatly help to keep the delay that transfers impose low. This can help in cases where low ping times are essential, like when using VoIP or terminal sessions, playing online games, etc.