Universiteit Leiden
The Netherlands

Opleiding Informatica

The state of Bufferbloat in the Netherlands

Bernardus A. Jansen

Supervisors:
prof. dr. H.A.G. Wijshoff & dr. K.F.D. Rietveld

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

01/08/2017

Abstract

In recent years, internet connection bandwidth has steadily increased, with a large majority of dutch households now having access to broadband internet connections. Even so, complaints about slow or sluggish internet are still commonly heard. It appears that many internet connections are still plagued by the effects of Bufferbloat, a phenomenon introducing high latency by excess buffering of packets. The existence of this phenomenon has been known for many years and several solutions have been proposed and demonstrated to alleviate its effects. As measures demonstrated to rid networks of Bufferbloat have now been around for several years, we have performed a number of tests on modems supplied to dutch consumers to check for the presence, severity and location of Bufferbloat in the network. The modems tested represent a significant share of dutch home internet connections. We have found that although some measures seem to have been taken, Bufferbloat is still found in many modems commonly found in the Netherlands. Further study could be directed to implementing our test procedure in a web application, allowing consumers to easily test their own internet connection for Bufferbloat and to collect a greater number of results.

Contents

Chapter 1

Introduction

Internet Service Providers eagerly market their offered services as being very fast. In recent years, bandwidth of home internet connections has increased steadily, with 96% of homes in the Netherlands having access to a wired connection with at least 30mbit/s of download bandwidth at the start of 2015 [Cen16]. Even though practically everyone in the Netherlands now has access to 'fast' broadband internet connections, complaints about the internet being slow or sluggish are still commonly heard. This is because the figures reported by ISPs actually do not concern the *speed* of an internet connection, but rather its *bandwidth*. The actual speed of an internet connection is better expressed in terms of the time it takes to receive a reply from a certain server, the Round Trip Time. As these RTTs are very similar between different ISPs and the actual signals involved in internet communication always travel at fixed speeds, these numbers are much less interesting to use in marketing. Even though internet signals always travel at constant speeds, the other component of what makes up a round-trip-time, the time it takes to process a packet, can dramatically increase under certain circumstances. As in such a situation it takes longer for a packet to reach its destination, the 'speed' of the connection is decreased, and many applications will feel sluggish to use.

The term Bufferbloat was coined by Jim Gettys in 2011 to describe the problem of high latency in packet-switched networks introduced by excess buffering of packets [Get11]. This high latency severely impacts the usability and perceived quality of an internet connection as all traffic passing through the buffer is significantly delayed. The problems underlying bufferbloat have been described as far back as 1985 [Nag85].

Buffers are an integral part of the internet and as such, practically all networking hardware uses buffers. The main use of these buffers is to prevent packet loss for short bursts of traffic, when not all packets can be immediately processed or forwarded to another host. In this capacity buffers are very important mainly for 'edge' devices, devices located between two networks with different bandwidth capacities. A modem used for home internet connections is a prime example of such a device. A problem with buffers is however that they do not play very well with TCP's built-in congestion avoidance. The way in which TCP determines how fast data can be sent across the network is by incrementing its speed step by step until notification of dropped packets is received. It then decreases the rate of sending data so no further packets are lost. After waiting a certain time it again ramps up its speed in case the link speed has changed to be able to achieve continuous maximum throughput. This behaviour combined with buffers however can lead to problems. As buffers are specifically designed to prevent packet loss, no packets are lost until the buffer is completely full. As packets are only dropped after the buffer has been completely filled, TCP's constant ramping up will ensure the buffers stay full at all times.

A long established rule of thumb for buffer sizes is that buffers should be able to buffer the full data rate of the corresponding interface for 250 ms [AKM04]. This means that a device with a gigabit network interface should have a buffer capable of accomodating 250 mbit of packets. This rule of thumb holds up quite well for switches and devices in a core network where all devices operate at the same speed, but it can form problems when applied to edge devices. A modem for a home internet connection is a prime example of a situation where large buffers can lead to problems. In home internet connections, bandwidth is often asymmetric on the WAN side while it is symmetric on the LAN side. Bandwidth available for downloading is often much greater than for uploading. Additionally, the available bandwidth is often dictated not by the hardware on either side of the connection (modem on one, DSLAM[1] or CMTS[2] ('edge' devices on the ISP side) on the other), but by external factors such as the length of the link, cable quality or arbitrary bandwidth constraints depending on the service the user subscribes to. The data rate that is negotiated is dependant on these external factors and can be any number between near zero and the maximum data rate supported by the hardware on either end. This in contrast with commonly found ethernet hardware where negotiated speeds are either 10, 100 or 1000 mbit/s, and always symmetrical. As buffers exist in hardware, it is difficult to implement a buffer that is optimal for all negotiable link speeds.

In recent years gigabit interfaces on consumer modems have become commonplace while connections to the internet rarely exceed 100 mbit/s and with upload rates rarely reaching even 20 mbit/s. Even so, internet service providers are more and more moving to supporting only a very small range of devices or even only a single device that allows their customers to connect to the internet. For example, the Dutch provider KPN supplies all of its customers, both DSL and high-speed fiber subscribers with the same residential gateway hardware. This means that this single device could have to handle a symmetric 500 mbit/s fiber connection in one situation or an asymmetric 100/20 mbit/s DSL connection[3] in another situation. If the device's buffers are designed to accomodate 250 ms of 500 mbit traffic as per the rule of thumb mentioned earlier, this means that when this device is used for a DSL connection with the characteristics mentioned just above, this device will buffer up to 5 whole seconds of traffic while uploading data. For transfers of large files, this is not a problem, but should a user want to browse the internet while uploading data, the additional packets sent to request data from a webserver now also have to traverse the entire buffer. This means it will now take at least 5 seconds before a page can even begin to be downloaded.

A lot of problems regarding buffering induced delays can be lessened by simply using smaller buffers, appropriate to the actual data rate. In 2011 the DOCSIS[4] specification was amended to enable ISPs to adjust

[1] Digital Subscriber Line Access Multiplexer
[2] Cable Modem Termination System
[3] In this notation the download bandwidth is 100mbit/s and the upload bandwidth is 20mbit/s
[4] Data Over Cable Service Interface Specification

cable modem buffer sizes, allowing them to size buffers appropriate to the actual data rate of the internet connection [Cab11]. While using more appropriately sized buffers reduces the effects of Bufferbloat, filling them up still adds latency. For relatively small buffers this added latency can now be acceptable for applications like VoIP or web browsing, added delays can still exceed 50ms, which can hinder online gaming. Additionally, connections with high data rates would still employ large buffers. Therefore, solutions have been developed in the form of Advanced Queue Management, scheduling algorithms to manage existing (large) buffers in a smart manner. Most notable of these solutions are RED (Random Early Detection) [FJ93] and CoDel (Controlled Delay) [NJ12]. The latter of which has been specifically developed to be easy to implement in order to stimulate adoption by manufacturers and developers as this was found to be a problem for RED. These solutions work by dropping packets and/or when supported by the network, sending Explicit Congestion Notifications to prevent buffers from completely filling up, allowing other traffic such as DNS requests or VoIP traffic to pass through as normal. Even though these queue management solutions have now been around for several years, it appears a lot of currently used consumer modems still lack advanced queue management implementations and are still plagued by the problems described above.

High latency has a huge effect on the perceived quality of an internet connection. Using a modem that is not prone to causing bufferbloat is therefore almost essential, especially for users of real-time applications such as internet telephony and online gaming. As in the Netherlands ISPs customarily provide their customers with modems and prevent users from connecting self-bought modems, internet subscribers have to rely on their provider to provide them with adequate hardware and are often unable to simply buy a better device themselves. While complaints about the quality of internet service are often heard nowadays, end users are often unaware of what is actually causing the problems they are experiencing. When these complaints reach customer service desks, users are often directed to perform a 'speedtest' to test their connection. Like most commercial communications from ISPs, these tests report the connection's bandwidth as its 'speed'. Even when a connection suffers from severe bufferbloat, these speedtests will suggest everything is fine as bufferbloat does not impact bandwidth. Even when ping round-trip-time is included as a metric in the reported results, this ping is often performed before the connection is saturated to test the available bandwidth and will also not show any sign of bufferbloat, suggesting the connection is performing fine.

1.1 Related Work

Over the years, several projects have been launched to increase awareness of Bufferbloat and to help users mitigate the effects of Bufferbloat. One of these is the Bufferbloat.net website. Bufferbloat.net has itself spawned several projects such as the CeroWrt project which was built upon the popular OpenWrt Linux distribution for routers, modems and access points, and served as a research project to test the implementation of CoDel [NJ12] and fq_codel [HMT+16] which eventually concluded in these algorithms being implemented in the mainline Linux kernel, greatly increasing the availability of advanced queue management.

The effects of Bufferbloat are also being recognized by manufacturers of networking equipment, with the research and development consortium of cable operators CableLabs mandating the implementation of AQM for DOCSIS 3.1 devices and recommending it for the current generation of DOCSIS 3.0 devices [Whi14].

Dave Täht of Bufferbloat.net has also developed the Realtime Response Under Load (RRUL) test specification [T10], designed to comprehensively test network responsiveness while under load. The tool Flent which we will use in our tests implements part of the RRUL specification.

Additionally, ISP reviewer DSLReports provides a speedtest[5] that tests for Bufferbloat in addition to bandwidth and allows to compare results with other users of the same or other ISPs. Unfortunately, no tests can yet widely be found that allow to test for Bufferbloat in a specific device on a network path.

1.2 Thesis Overview

In this thesis, we will test the performance characteristics under load of a number of modems as supplied by dutch ISPs and currently found in homes in the Netherlands to determine whether these devices exhibit signs of bufferbloat and thus in what way Bufferbloat still is a problem for consumers.

This chapter contains the introduction; Chapter 2 describes the setup of the tests performed; Chapter 3 presents the results of the tests and Chapter 4 concludes. This thesis was written as a bachelor project at the Leiden Institute of Advanced Computer Science, Leiden University, and was supervised by prof. dr. H.A.G. Wijshoff and dr. K.F.D. Rietveld.

[5] http://www.dslreports.com/speedtest

Chapter 2

Test setup

To determine how everyday users of the internet in the Netherlands are still affected by Bufferbloat, tests have been performed on a number of modems as supplied to subscribers of basic consumer internet connections offered throughout the Netherlands. As the home internet market in the Netherlands is dominated by just two major parties and both of these parties supply their customers with only a limited range of modems, only a small number of test sites is required in order to achieve results representative of a large number of internet connections. In order to be able to detect whether Bufferbloat manifests itself in a network, a number of programs are used to execute tests.

2.1 Netperf

Netperf is a tool originally developed by Hewlett-Packard that allows for testing the available bandwidth between two hosts[1]. It works in a client-server setup with one of the hosts acting as a netperf server and the other as a client. The client can then start a netperf stream, transferring data to or from the netperf server as fast as the link between server and client allows. In this way netperf can be used to saturate a link which allows to test network responsiveness while the network is congested.

2.2 Flent

Flent is an acronym for FLExible Network Tester [Høi]. It is a tool developed to characterize network performance in a number of scenarios. One of the tests supplied by Flent is the Realtime Response Under Load (RRUL) test. This test was specifically developed to expose the effects of Bufferbloat. The RRUL test is implemented using concurrent netperf streams connecting to a netperf server, reliably saturating the connection wherever the bottleneck may be. Before, while, and after the network is saturated, Flent continually pings a

[1]https://hewlettpackard.github.io/netperf/

specified host. If Bufferbloat is present in the network, the ping round-trip-times are expected to (dramatically) increase during the period of network saturation.

2.3 Buffchar

Buffchar (Buffer characteristics) is a tool developed by students at the university of Amsterdam [GK11] in order to log buffer characteristics in a network path over time. Buffchar works by executing a number of traceroutes to a specified server and calculating the latency added in every hop. Using this tool, the link where Bufferbloat occurs in the network path can be discovered allowing us to determine which device is responsible for the added latency. As bufferbloat only occurs at network bottlenecks, the link that displays the largest increase in added queue delay is the link responsible for Bufferbloat.

2.4 Setup

We first use Flent to determine the presence of Bufferbloat in the network. When Bufferbloat is present, this will be represented by Flent in the form of a characteristic trapezoid graph as under the effects of Bufferbloat, ping round-trip-times will increase while the network is under load. The netperf streams Flent uses to saturate the network connect to the server *stor.bajansen.nl* while it will ping the server 37.97.254.1. Both servers are located in datacenters in the Netherlands and have high-speed connections to the internet, ensuring that the bottleneck in the connection between the test location and the test server will always be link between the test location and the ISP's network and is not limited elsewhere in the route. Additionally, the netperf server is located in a datacenter different to the server that is used to determine the ping round-trip-times. In this way, the netperf streams only risk filling up buffers in the part of the route that is shared between both servers. If Bufferbloat is to occur anywhere in this part of the route it will therefore always influence the responsiveness of the internet connection as all traffic to and from this location has to pass through the same hops.

In order to be able to determine what device in the route to our ping endpoint is responsible for causing Bufferbloat, we utilize a version of Buffchar that has been modified to support SQLite to store results as opposed to MySQL in order to simplify handling of test results. To detect the device causing Bufferbloat Buffchar is executed twice, first when the network is idle and then when the network is congested. To saturate the network we run netperf using the same configuration as used by Flent. The server Buffchar performs its traceroutes to is the same Flent pings to and has been specially selected such that there are no hidden hops on the route from test locations to the server, in order to be able to precisely determine what delay is added by which hop.

To be able to execute the tests in a straightforward and structured fashion for each test location a wrapper script is used to execute the tests and in the case of the second Buffchar test to concurrently run netperf to saturate the network.

2.5 Test Protocol

Besides the metrics collected by running the wrapper script to execute the tests, it is important to note additional details and specifications about the location the test is executed. To accurately capture this information and correctly execute the tests the necessary steps have been laid out in the following test specification:

1. Note address (postal code + house number) of test location
   - Note this and all following data in a text file named *TESTID.txt* where *TESTID* is a unique name for the specific test location. The argument passed to the wrapper script in the last step of this protocol should be equal to this unique name.
2. Note length of the cable between the modem and where it plugs into the provider's cabling in the wall when this is more than two meters.
3. Note type of modem (Brand + type + version + whether it is in bridge mode or router mode if applicable)
4. Connect laptop used for testing directly to the modem. The laptop's ethernet adapter has to support gigabit and the cable used should be at least CAT5E
5. Note the maximally available bandwidth
   - Find out the bandwidth as per the user's contract
   - Use speedtest.net to check the actual available bandwidth
   - In case of KPN the bandwidth available at a specific address can be found using https://netco-fpi-info.fourstack.nl
6. Note the IP address of the connection as reported by ipv4.icanhazip.com
7. Execute the wrapper script to perform the actual tests using the command *sudo./wrapper.shTESTID* while in the appropriate directory.

Chapter 3

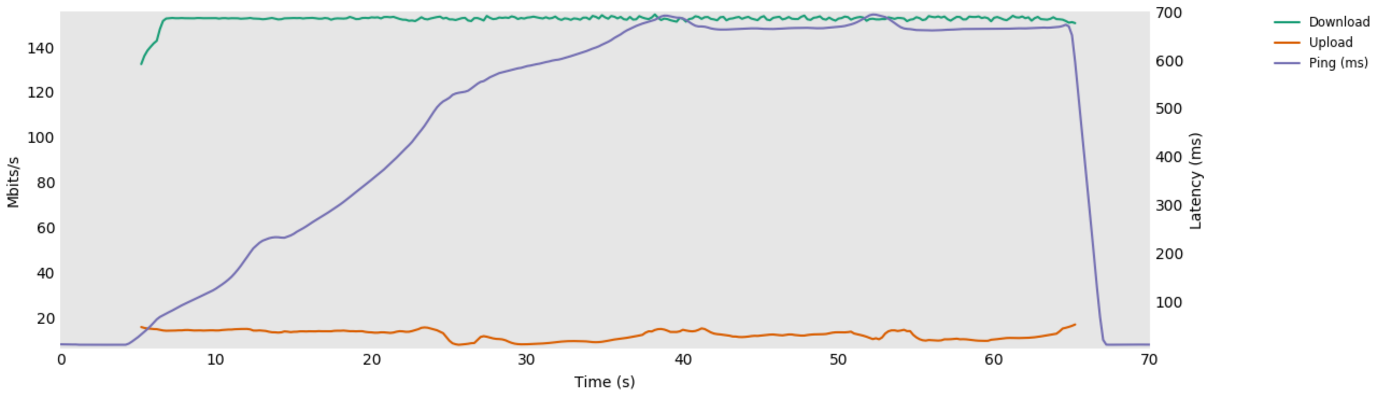
Results

The results from our tests are represented by two graphs for each test site. The first is a graph generated by Flent. The first five seconds of the graph the network is idle. In the middle 60 seconds of the graph the network is congested and the last five seconds the network again returns to idle. As Bufferbloat causes high latency under load, presence of Bufferbloat is visible as a significant increase in ping RTTs during the period of congestion. The purple line represents the ping RTTs and the green and orange line represent the combined bandwidth of the download and upload streams respectively and as such shows the maximally supported bandwidth in either direction.

The second graph is generated by buffchar and consists of two bars. The first bar represents the state of the network while idle, the second the state of the network while under load. The first bar can therefore be compared with the first five seconds of the graph produced by Flent and the second bar can be compared with the middle 60 seconds of the Flent graph. The horizontal subdivisions of the bars represent hops in the network path and the queue delay added by each hop. The bigger a horizontal subdivision, the higher the delay added in that hop. As buffchar averages all reported RTTs during each execution, the numbers it reports can differ from the figures reported by Flent. The two bar charts generated by buffchar should mainly be compared with each other to determine which hop adds more delay while under load when compared to the idle situation.

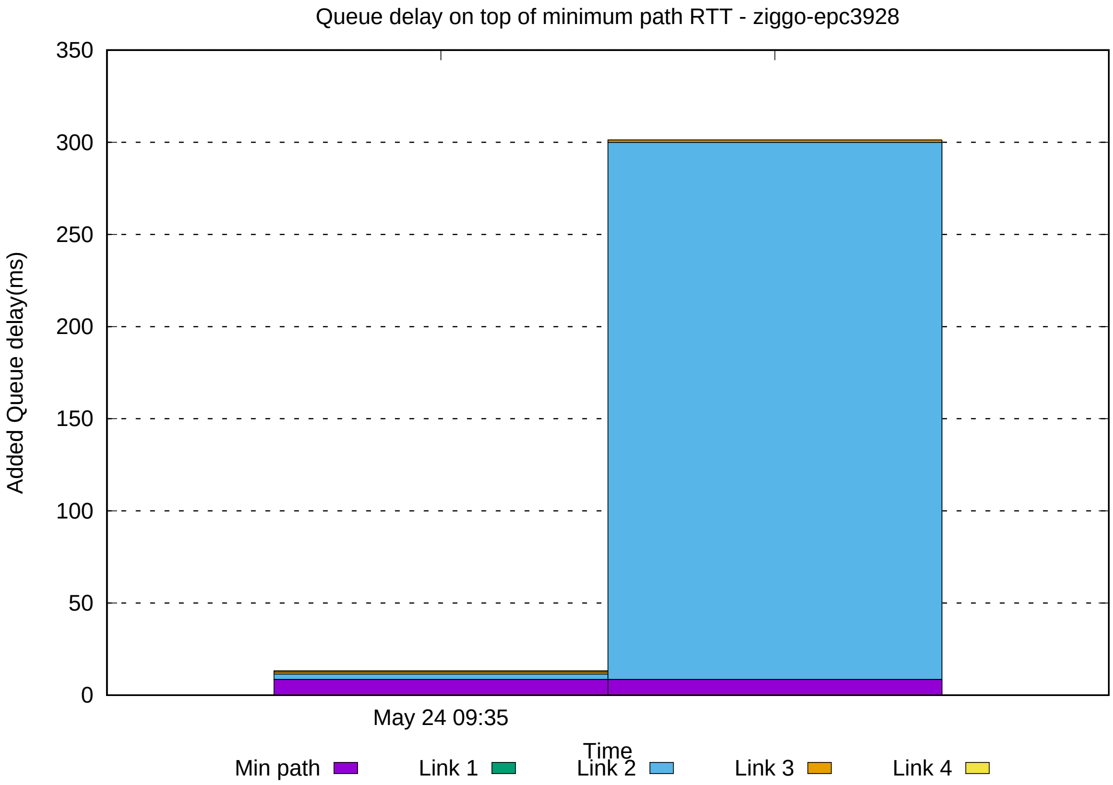
3.1 Site 1

Ziggo 150/15 Cisco EPC3928 Leiden



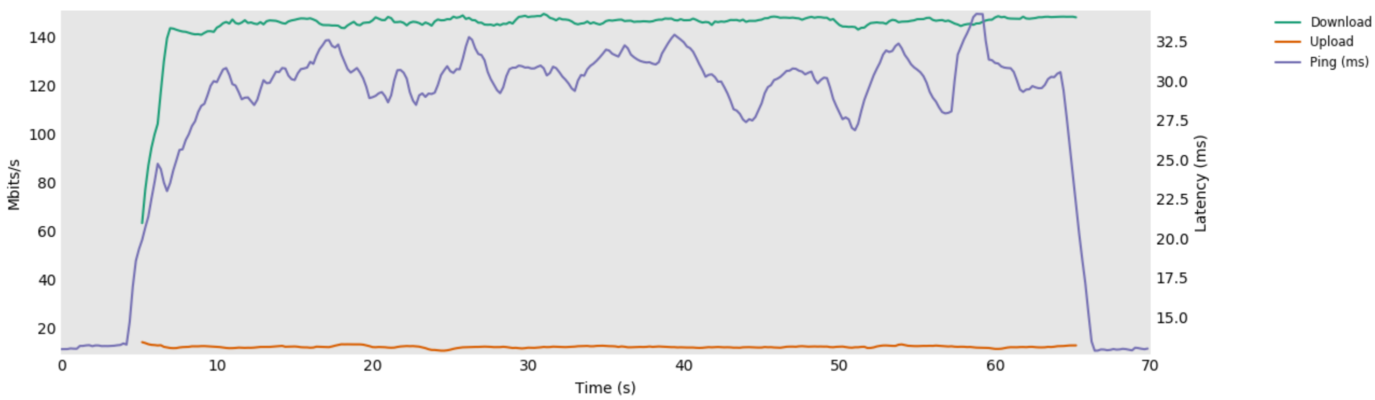
The first device tested is a Cisco EPC3928 as supplied by dutch ISP Ziggo, starting around 2012. This device was configured in bridge mode and as such its routing features were disabled. As it still functioned as a modem, traffic still passes through its buffers. The first five seconds the network is idle with RTTs to the test server around 12ms. After five seconds, load is generated congesting the network. This has a drastic effect on the RTTs, rising to nearly 700ms. After 60 seconds the netperf streams are stopped and the network is no longer congested, allowing the buffers to empty with RTTs immediately dropping to again around 12ms. This device can thus be said to exhibit very severe effects of Bufferbloat. The buffers in this device are in fact so large, the filling of the device's buffer can actually be seen in the gradual increase of the ping RTTs while the device is under load.

It is expected that the high latency seen in the graph above originates from the modem's buffers. As the internet connection is transparantly bridged to the laptop used for testing, this should be indicated by buffchar as a large amount of added queue delay in Link 1 in the second bar. Buffchar however shows a huge amount of added queue delay in Link 2, which would mean there is a bottlenecked connection between two routers in the provider's network. Investigating buffchar's raw collected data however shows that the first hop in our network path did not respond to the traceroute command and as such the delay that probably originates from Link 1 is added to the delay added in Link 2.



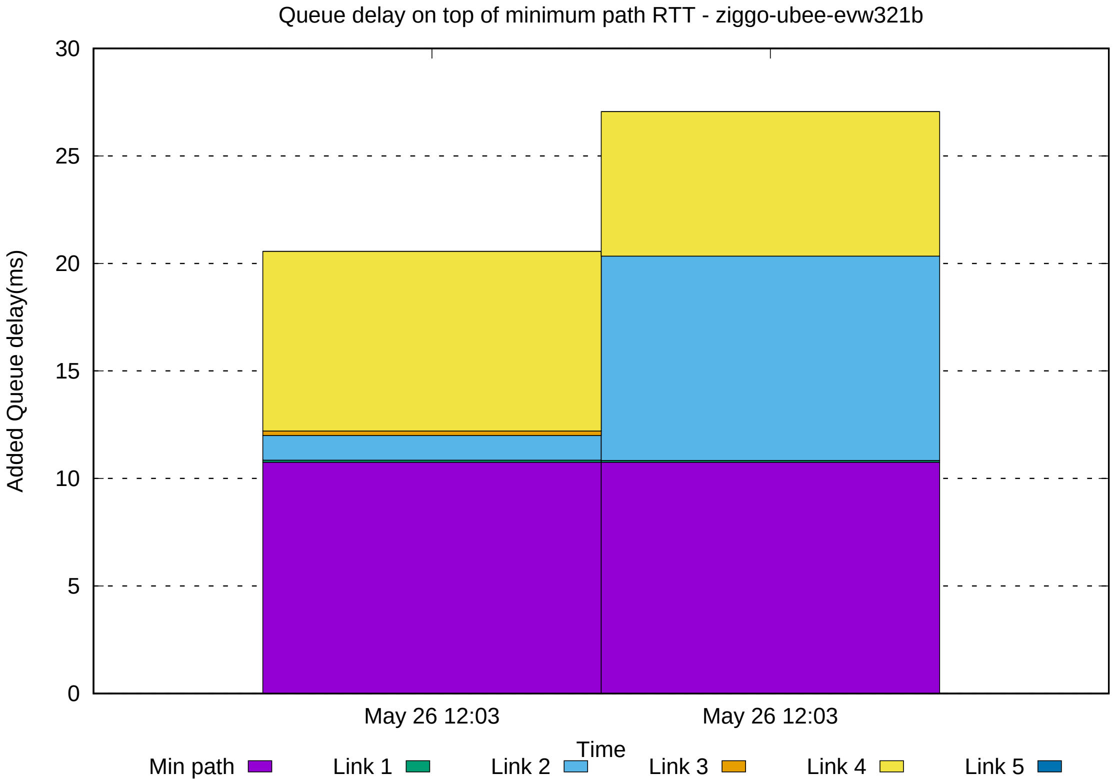
3.2 Site 2

Ziggo 150/15 Ubee EVW321b Oudehaske



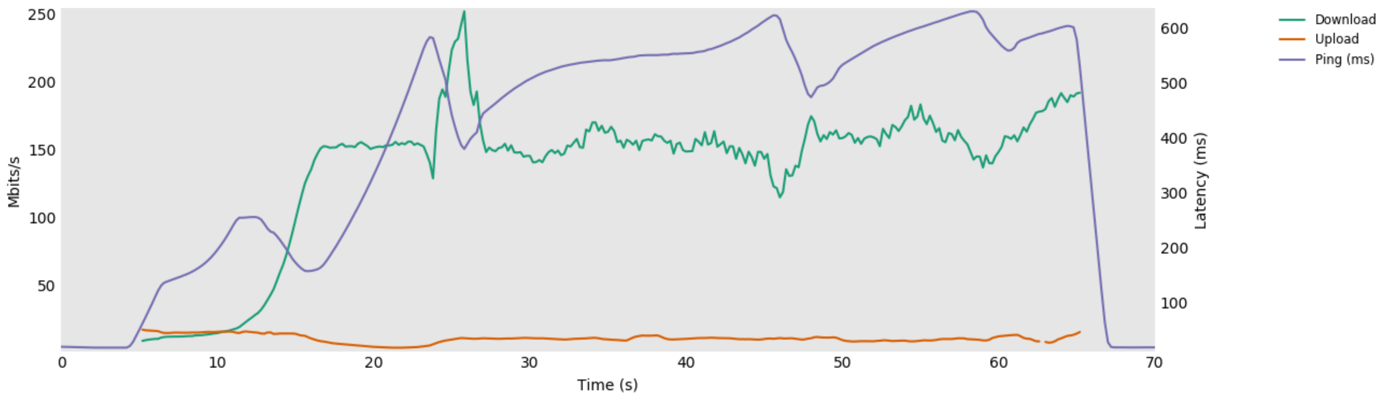
A device that performs much better is the Ubee EVW321B. This device was supplied to Ziggo subscribers around the same time as the Cisco EPC3928. While this device does add latency while under load, it adds much less than the Cisco modem, with RTTs rising from an initial 13ms to a maximum of under 35ms. As such, this device can not be said to be suffering from bufferbloat. A possible explanation is that as of 2017 the device is still supplied to new subscribers to Ziggo's 40/4 mbit/s service. As these relatively low bandwidth connections would be suffering from severe Bufferbloat while under load, this could have motivated Ziggo to improve the device's handling of its effects.

As this device is configured in the standard routing mode, Link 1 is now the link between the device and the test laptop. Any added queue delay is therefore expected in Link 2 as this is now the link between modem and the ISPs gateway, which is exactly is reported by buffchar. It is also immediately clear that the increase in added delay in the link between the modem and the ISP is much lower than in the previous test. The relatively large added delay in Link 4 should not be taken to suggest Bufferbloat, as it remains relatively equal in size both when the network is idle and when it is congested.



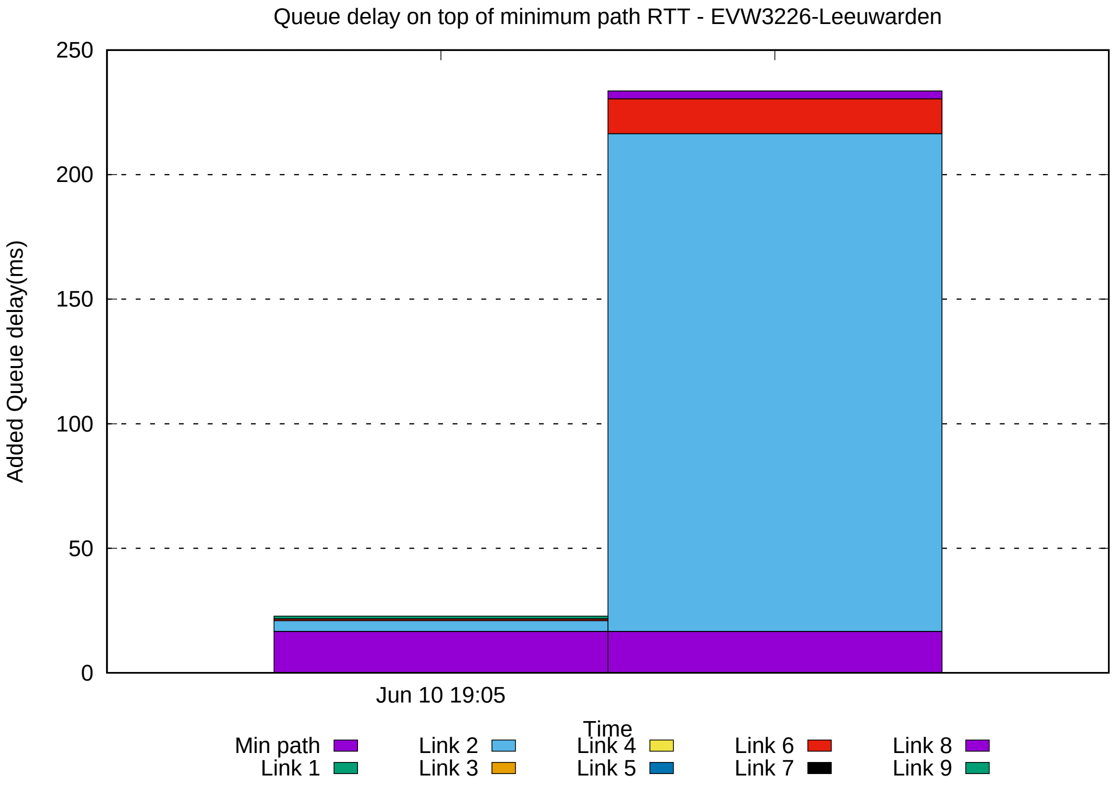
3.3 Site 3

Ziggo 150/15 Ubee EVW3226 Leeuwarden



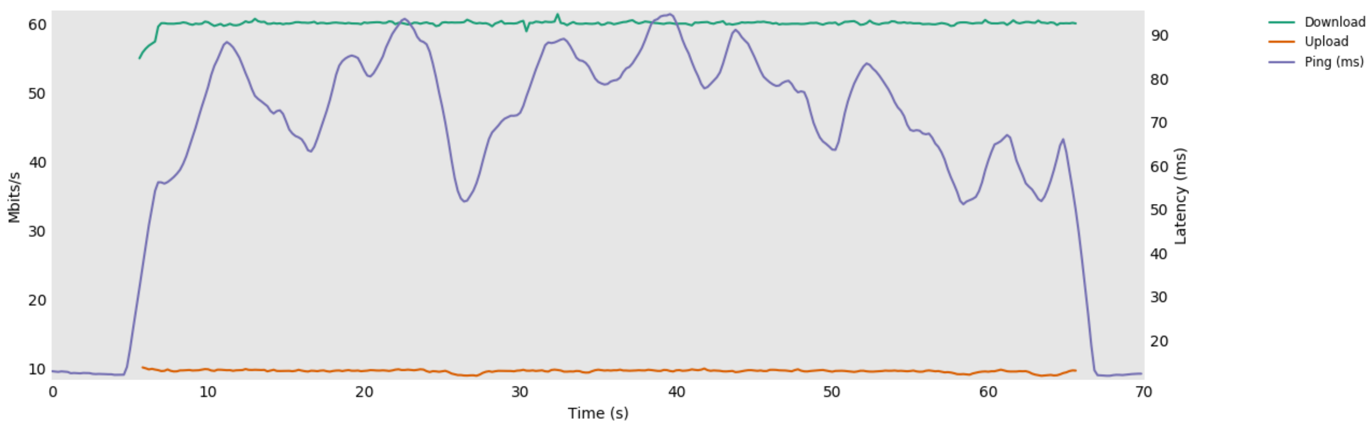
While the above graph shows some anomalies with regards to the reported download bandwidth and latency over the course of the test, the general trend of a steadily increasing very high added latency also seen in the first test can be recognized. This observation is interesting, as it shows even devices produced by the same manufacturer can show enormous differences when it comes to Bufferbloat. As the Ubee EVW321B that was shown to perform very well and the Ubee EVW3226 appear to be very similar devices, tout a lot of the same hardware features [Ube10b, Ube10a] and were supplied to internet subscribers around the same time, one would expect for the devices to perform similarly as well. As seen in the graph above however, this is clearly not the case. Ping RTTs rise from an initial 16ms to well over 600ms, indicating severe bufferbloat.

The EVW3226 was configured in routing mode and we thus expect significant added queue delay in Link 2, which is again exactly what buffchar reports. Notable is that the path to our test server consists of many more hops than our previous two Ziggo examples. In 2014 the dutch ISP Ziggo was acquired by Liberty Global resulting in the merger of UPC Nederland and Ziggo with the name UPC being phased out. While all of former UPC is now named Ziggo as well, differences in network setup can still be found, as is the case for this test site, a former UPC connection.



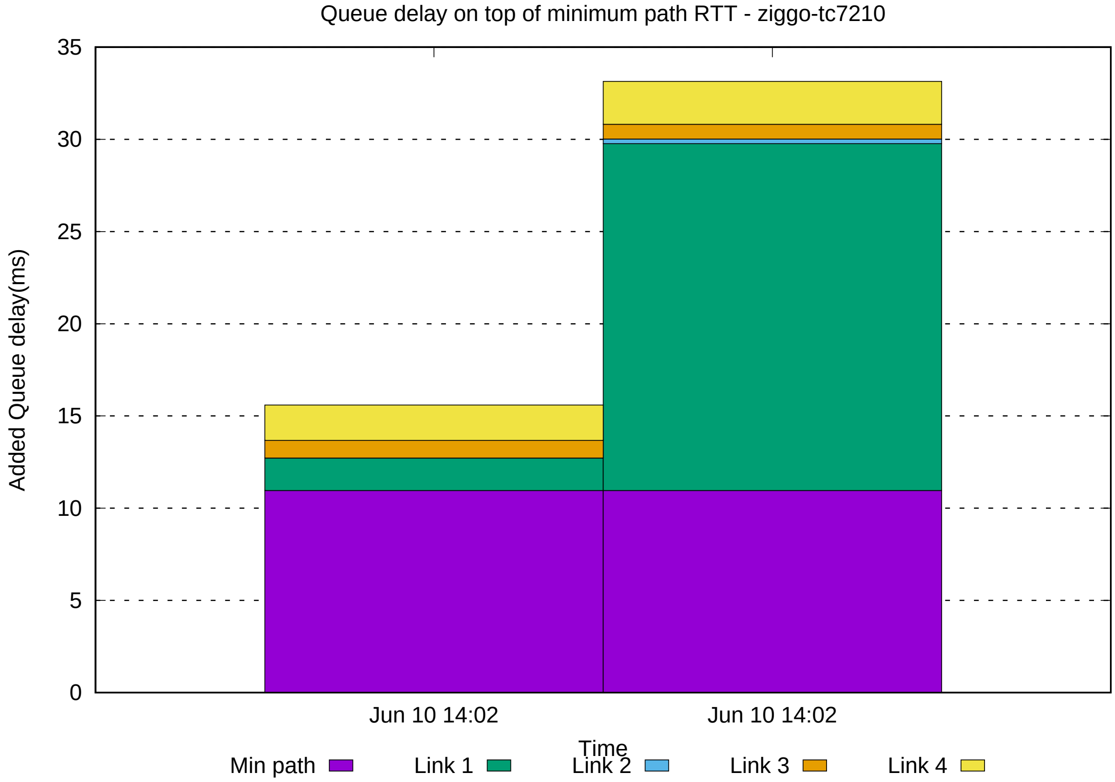
3.4 Site 4

Ziggo 60/10 Technicolor TC7210 Oldeberkoop



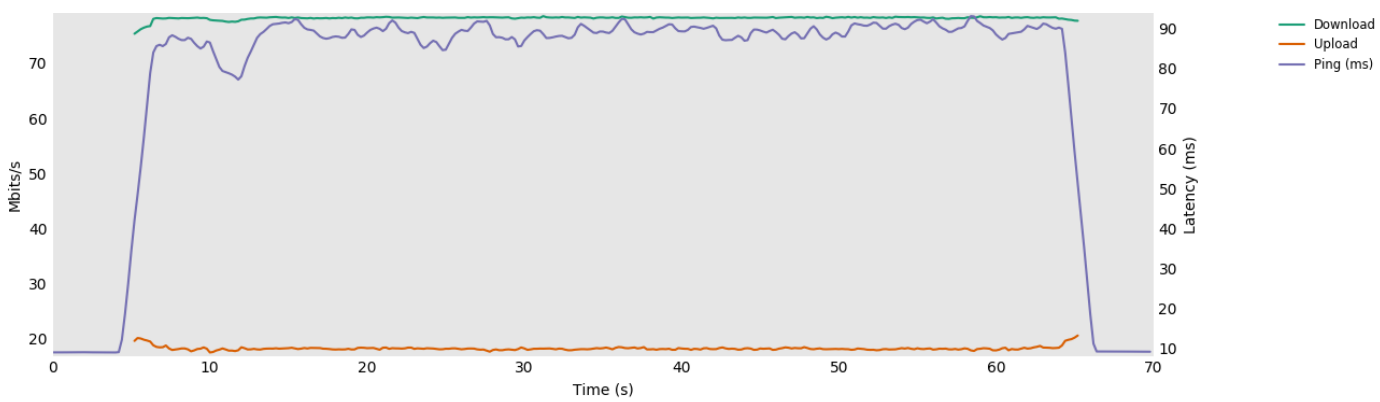
The Technicolor TC7210 is one of the newer devices supplied by Ziggo. This device does not perform too bad, but RTTs still rise to over 90ms while under load. Interesting is that the ping plot exhibits a quite pronounced jagged effect when compared to the Ubee EVW321B. This could indicate that this device employs some kind of Active Queue Management, with the amount of packets in the buffer increasing and decreasing while throughput remains constant.

As this device was configured in bridge mode, Link 1 is in this case the connection between the laptop executing the tests and the ISPs gateway, with the modem transparantly bridging in between. As expected, comparing the two bars shows that the added queue delay while under load originates from this first link.



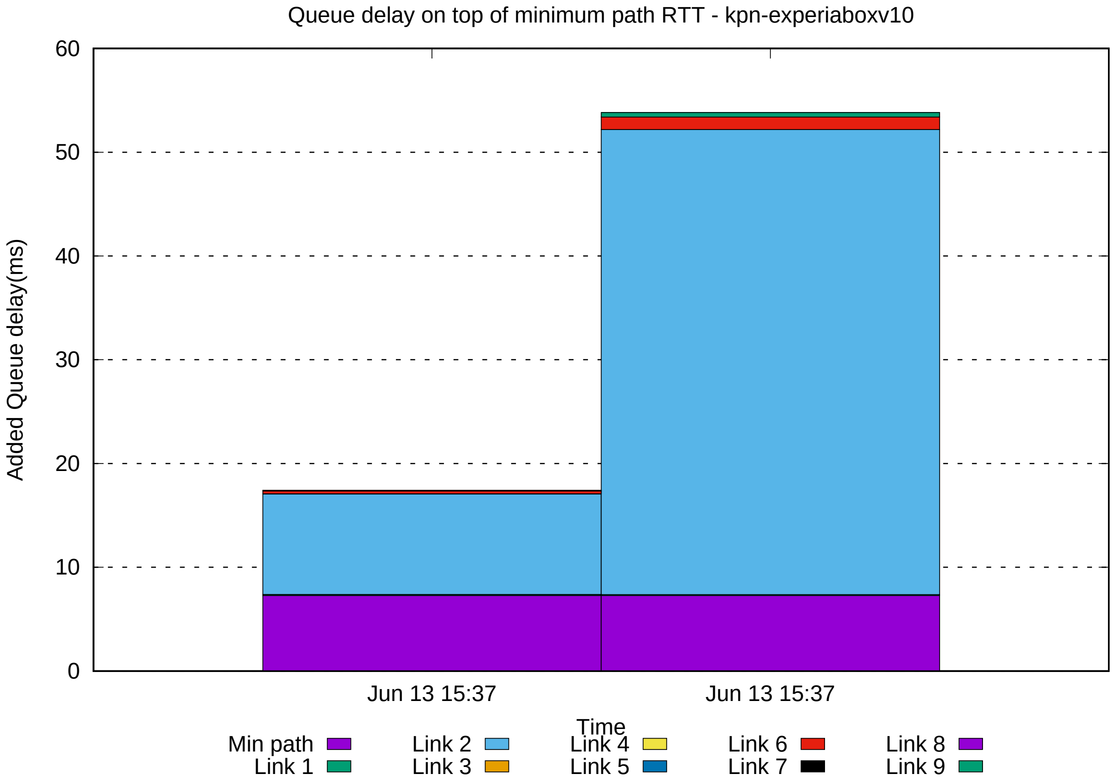
3.5 Site 5

KPN 100/10 Experiabox V10 (ZTE H369A) Leiden



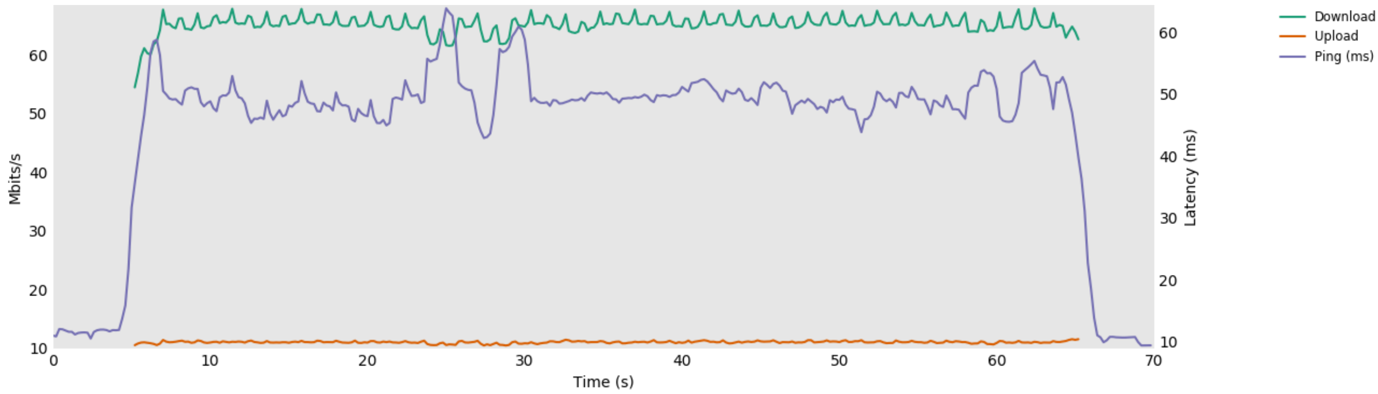
The Experiabox V10 is the latest generation of residential gateway as supplied by the other significant internet service provider in the Netherlands, KPN. While this device exhibits much less Bufferbloat than the devices in the first and third tests, RTTs still rise to over 90ms. When comparing this Flent graph with the previous test site, the RTTs are in this case much more stable. This could indicate the device's only attempt to lessen the effects of Bufferbloat is use of reduced buffer sizes.

As the device is configured in its standard routing mode, the effects of Bufferbloat are expected to originate from Link 2. Comparing the buffchar idle and under-load bars, an increased added queue delay is seen in this link, confirming the hypothesis.



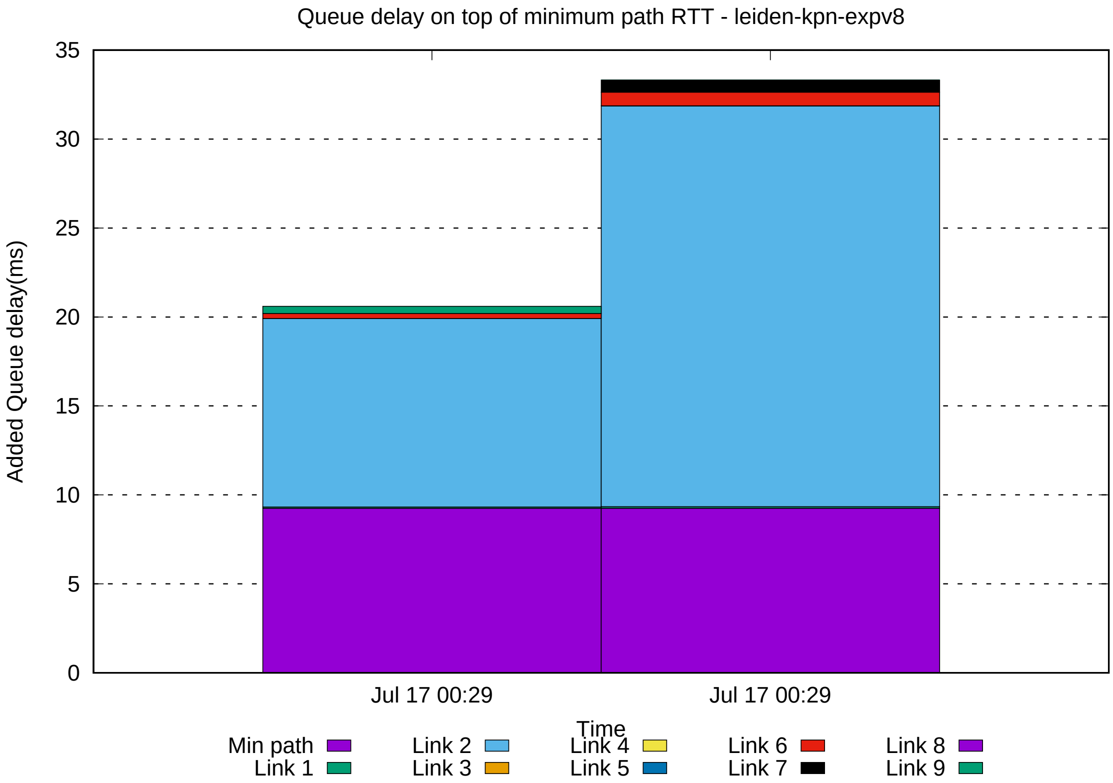
3.6 Site 6

KPN 100/10 Experiabox V8 (Arcadyan VGV7519) Leiden



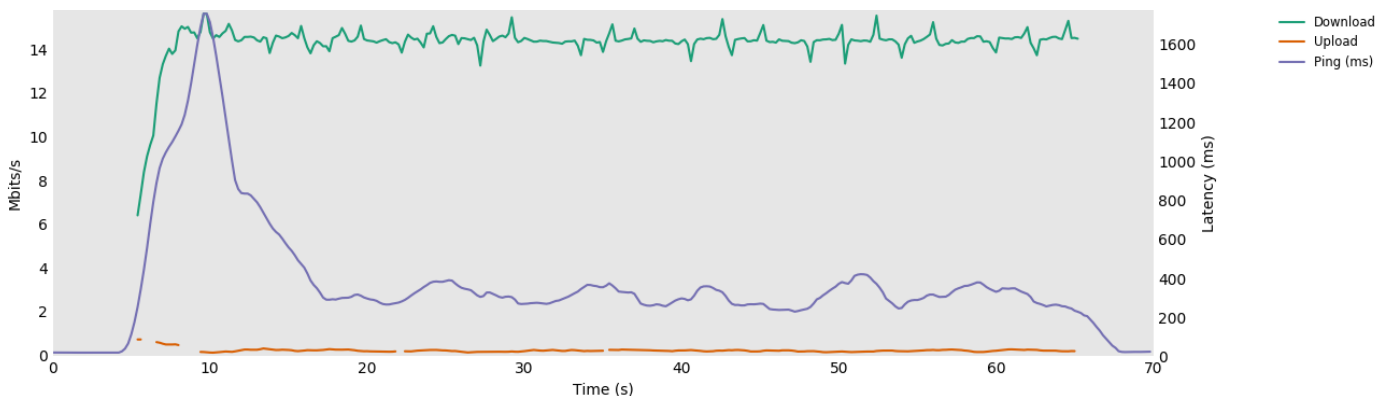
Interestingly, the KPN Experiabox V8 which is several years older than the V10, but as of 2017 is still commonly supplied to consumers, suffers less from Bufferbloat than its successor. While appearing slightly more jagged than in the test directly above, the ping plot does not show significant variation while under load which could again indicate that it does not employ AQM but merely uses relatively small buffers to reduce the effects of Bufferbloat. The main difference between the V8 and the V10 with regards to bufferbloat is therefore probably just its buffer size, with no evidence of AQM being applied. The 'smoother' graph of download bandwidth in the case of the Experiabox V10 when compared to this device might be due to newer, improved VDSL hardware in the V10 when compared to the older V8.

The figures reported by buffchar are also very similar to the Experiabox V10 in the previous test, but with the subdivision for Link 2 being slightly smaller under load because of the lower added delay.



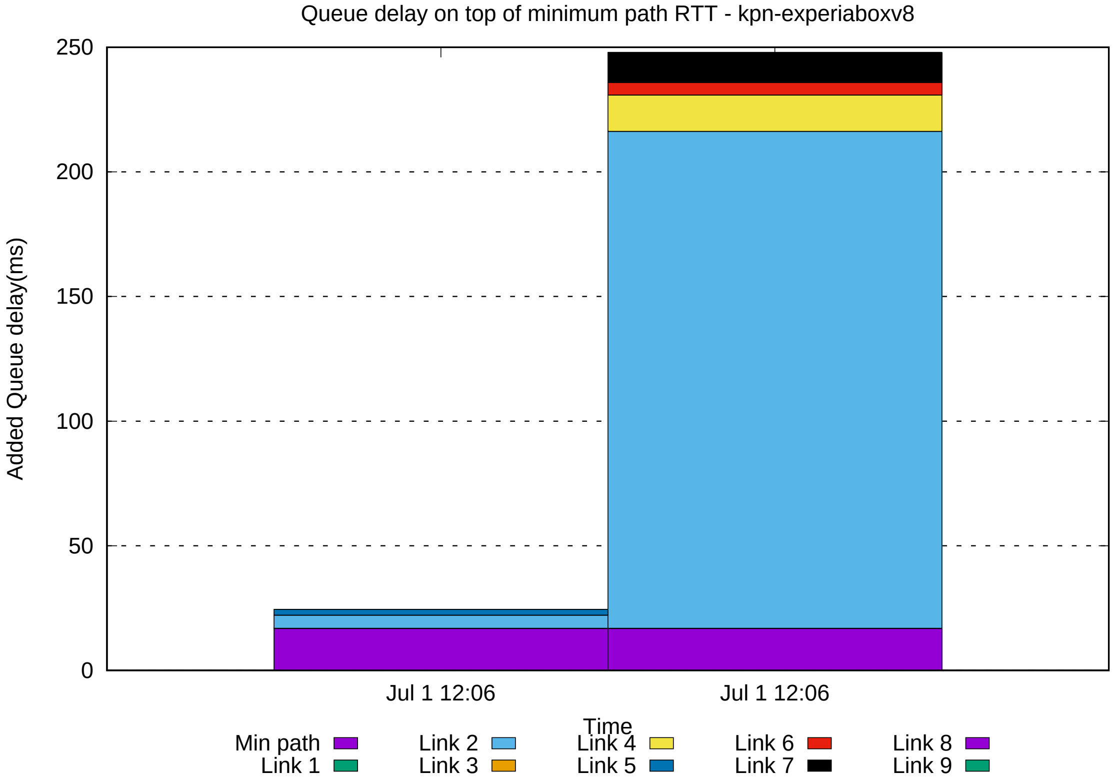
3.7 Site 7

KPN 20/2 Experiabox V8 (Arcadyan VGV7519) Wolvega



The final test again concerns the eighth generation of the experiabox, but now connected to an internet connection with significantly lower data rates than the previous test. Notable is that apart from the initial spike, RTTs remain around ten times higher than in the previous test, with the upload bandwidth around ten times lower. This indicates the device's buffer size is not configured by the provider according to the actual data rate, but is fixed in size. While this buffer size worked out fine at our previous test site, it severely impacts internet responsiveness in this lower bandwidth situation.

Also notable is that this connection concerns an older ADSL2 connection while the previous two KPN tests concerned VDSL connections. The initial ping spike in this test does not appear to be an anomaly and was present in multiple tests performed for this site. The spike may be a result of buffering of the download streams in the ADSL DSLAM hardware used by the ISP. If this is the case, the DSLAM appears to utilize AQM to reduce latency after the initial spike, with the latency added by the experiabox itself remaining, as the modem itself does not appear to use AQM, as also seen in test 6. This assumption still holds when looking at the Buffchar graph. Significant latency is added in Link 2, which is the link between modem and DSLAM, but delay can be added by both sides of the link.



Chapter 4

Conclusions

In this thesis we have tried to determine whether Bufferbloat is still a problem for consumers in the Netherlands. We have tested the performance characteristics under load of a number of devices, representative of a large number of dutch internet connections. The results show that handling of Bufferbloat varies greatly between the devices tested. While some devices perform quite well, others still add a significant amount of latency while under load. It is also observed that the best performing devices are those that are relatively new and/or are still supported by their supplier.

While it can be seen the existence of Bufferbloat has been recognized by device manufacturers and measures are apparently taken to reduce its effects, severe Bufferbloat was still found in three out of our seven test sites and the best performing device among our tests still increases RTTs threefold while under load. While this situation is likely to further improve in the future due to for example the mandatory implementation of active queue management in DOCSIS 3.1 devices, it is unlikely all older Bufferbloat-plagued devices will be replaced overnight. In addition, no governing body exists for the implementation of the more widespread DSL standards, meaning DSL device manufacturers and ISPs themselves have to take responsibility to banish Bufferbloat from their networks.

For internet subscribers suffering from Bufferbloat, the testing procedure developed for this thesis can prove to be valuable, enabling them to determine where in the network Bufferbloat occurs, and allowing them to conclusively point a finger at their ISP if their supplied hardware is responsible.

In order to increase awareness of Bufferbloat and to motivate ISPs to take measures to eradicate it, implementing this thesis's test procedure in a web application may prove to be a useful. This can allow consumers to easily test and diagnose the performance characteristics of their internet connection, while also collecting a large number of test results, which could in turn be used to increase awareness of Bufferbloat.

While this thesis has focused on Bufferbloat occuring in cable or DSL modems, Bufferbloat is also found in WiFi. While multiple solutions exist to mitigate the effects of Bufferbloat in wired connections, these solutions can not be readily applied to WiFi, and Bufferbloat in wireless connections is still an area of ongoing research [HKT+17]. As practically all home networks now include wireless connectivity and so do increasingly many devices, continued study and understanding of network buffering could prove to be very valuable.

Bibliography

[AKM04] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. Sizing router buffers. In Raj Yavatkar, Ellen W. Zegura, and Jennifer Rexford, editors, *Proceedings of the ACM SIGCOMM 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 30 - September 3, 2004, Portland, Oregon, USA*, pages 281–292. ACM, 2004.

[Cab11] Cable Television Laboratories. Cable modem buffer control, 2011.

[Cen16] Centraal Bureau voor de Statistiek. Ict, kennis en economie 2016, June 2016.

[FJ93] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4):397–413, 1993.

[Get11] Jim Gettys. Bufferbloat: Dark buffers in the internet. *IEEE Internet Computing*, 15(3):96, 2011.

[GK11] Danny Groenewegen and Harald Kleppe. Detecting and quantifying bufferbloat in network paths. Master's thesis, Universiteit van Amsterdam, 2011.

[HKT+17] Toke Høiland-Jørgensen, Michal Kazior, Dave Täht, Per Hurtig, and Anna Brunstrom. Ending the anomaly: Achieving low latency and airtime fairness in wifi. *CoRR*, abs/1703.00064, 2017.

[HMT+16] Toke Høiland-Jørgensen, Paul McKenny, Dave Täht, Jim Gettys, and Eric Dumazet. The FlowQueue-CoDel Packet Scheduler and Active Queue Management Algorithm, March 2016.

[Høi] Toke Høiland-Jørgensen. Flent: The FLExible Network Tester. `https://flent.org/flent-the-flexible-network-tester`.

[Nag85] John B. Nagle. On packet switches with infinite storage. RFC 970, December 1985.

[NJ12] Kathleen M. Nichols and Van Jacobson. Controlling queue delay. *ACM Queue*, 10(5):20, 2012.

[Tïo] Dave Täht. Realtime response under load specification, 2012 (Accessed: 2017-07-10). `https://www.bufferbloat.net/projects/bloat/wiki/RRUL_Spec/`.

[Ube10a] Ubee Interactive. Evw3210, Accessed: 2017-07-10. `http://www.ubeeinteractive.com/products/cable/evw3210`.

[Ube10b] Ubee Interactive. Evw3226, Accessed: 2017-07-10. http://www.ubeeinteractive.com/products/cable/evw3226.

[Whi14] Greg White. Active queue management in DOCSIS 3.x cable modems, May 2014.