Bufferbloat and Other Internet Challenges

Vinton G. Cerf • *Google*

Several years ago, James Gettys drew attention to a problem that he encountered at the edge of the Internet: an interface with a high-speed and a low-speed side led to the filling of a large buffer at the point where the high-speed side built a queue waiting to empty into the low-speed side. The latency end-to-end skyrocketed and persisted for long time periods. He called this problem "bufferbloat" (see https://gettys.wordpress.com/category/bufferbloat/, www.bufferbloat.net, and http://en.wikipedia.org/wiki/Bufferbloat). Detailed analysis by Kathleen Nichols and Van Jacobson determined that part of the solution is to control the permitted length of the buffer so as to avoid overfilling the queue in the low-speed direction with its subsequent long and potentially persistent latency for traversal.[1]

John Nagle recognized this kind of problem as early as 1985.[2] Nichols and Jacobson pursued a solution, initially proposing what they called CoDel (http://en.wikipedia.org/wiki/CoDel) to control queue length on the basis of measuring and managing local latency. Dave Täht and Eric Dumazet have implemented this algorithm for Linux, and Dumazet has gone further with his FQ_CoDel algorithm, which adds fair-queueing mechanisms to the active queue management (AQM) scheme. From the Linux manual pages (http://man7.org/linux/man-pages/man8/tc-fq_codel.8.html):

> FQ_Codel (Fair Queuing Controlled Delay) is queuing discipline that combines Fair Queuing with the CoDel AQM scheme. FQ_Codel uses a stochastic model to classify incoming packets into different flows and is used to provide a fair share of the bandwidth to all the flows using the queue. Each such flow is managed by the CoDel queuing discipline. Reordering within a flow is avoided since Codel internally uses a FIFO queue.

The bufferbloat problem is only one of many performance challenges faced at the edges of the Internet. Implementation of IPv6 in addition to IPv4 as well as stronger security measures remain significant issues in bringing a safer and more flexible Internet to the general public. Residential routers are inexpensive but often very out of date relative to their software. An initiative that's attempting to remedy this situation is CeroWrt (www.bufferbloat.net/projects/cerowrt):

> CeroWrt is a project built on the OpenWrt firmware to resolve the endemic problems of bufferbloat in home networking today, and to push forward the state of the art of edge networks and routers. Projects include proper IPv6 support, tighter integration with DNSSEC, and most importantly, reducing bufferbloat in both the wired and wireless components of the stack.

The use of open source software to promote broad adoption and use of new technology is now well demonstrated — for example, through the widespread adoption of the Linux and Android operating systems. The CeroWrt/OpenWrt effort could have a similar effect, especially if the resulting software can be ported to a variety of hardware platforms. Perhaps most important, however, is the possibility of enabling easy and safe automatic updating of residential router software to share new developments. That such updates carry with them the risk of ingesting malware is obvious and must be dealt with before we can realize any such benefit. This is no different than other systems for providing new software updates in laptops, desktops, tablets, and mobiles. We must also consider that such a system is dealing with the

cont. on p. 79

Published by the IEEE Computer Society 1089-7801/14/$31.00 © 2014 IEEE

cont. from p. 80

general public, not with software experts, and recourse must be found when things go wrong. We want to avoid "In case access to Internet is broken, consult the Internet."

This same line of reasoning applies to the Internet of Things, with similar concerns and consequences. Given that we don't know how to write perfect code, and even if we did, we might wish to improve or add functionality, we need some convenient means for updating software for all kinds of devices that is safe for users up to some degree. Digital signatures and certificates authenticating software's origin have proven only partly successful owing to the potential for fabricating false but apparently valid certificates by compromising certificate authorities one way or another.

We have come a long way from queue management to Internet security in this brief column, but I hope it's apparent that these disparate topics are linked by the need to find a path toward adapting Internet-based devices to change, and improved safety. Internet users will benefit from the discovery or invention of such a path, and it's thus worthy of further serious research.

References

1. K. Nichols and V. Jacobson, "Controlling Queue Delay," *Comm. ACM*, vol. 55, no. 7, 2012, pp. 42–50.
2. J. Nagle, *On Packet Switches with Infinite Storage*, IETF RFC 970, Dec. 1985; http://tools.ietf.org/html/rfc970.

Vinton G. Cerf is vice president and chief Internet evangelist at Google, and president of ACM. He's widely known as one of the "fathers of the Internet." He's a fellow of IEEE and ACM. Contact him at vint@google.com.