The Secure Transport Tussle

Christian Huitema, Microsoft

October 28, 2014

Submission to the IAB Workshop on Stack Evolution in a Middle-box Internet (SEMI)

Abstract

It is very hard to deploy transport protocol innovations such as encryption, multipath or forward error correction. This is largely due to the assumptions made by middle-boxes, which contribute to a practical freeze of the TCP specifications. Protocol developers tends to see that as a technical problem, but it really is a tussle in which middle-box vendors and operators compete for Internet value against end to end devices and services. We have an example of a similar tussle in the past, between real-time applications on one hand, NATs and firewalls on the other. The NAT traversal tussle was solved with the unilateral deployment of a variety of protocols, which repositioned the balance of power between end-to-end and middle-boxes. Protocol developers need to adopt the same approach for fielding transport innovation, and that probably means an encrypted transport on top of UDP, so they can win the "secure transport tussle."

Introduction

We see very little deployed innovation in transport protocols. This is not for lack of demand, or for lack of trying. The general explanation is that middle boxes are blocking future innovation. Protocol developers have to recognize this as a tussle between innovating end-to-end and adding value inside the network, and they need to adopt a strategy that lets innovation win this tussle.

If we could fix the transport, we could improve many Internet services, providing better latency and better reliability without sacrificing security and privacy. For example, we know that introducing forms of forward error correction would mitigate the latency variability caused by packet losses. We know that providing separate control for multiple embedded streams allows for better scheduling of application data, and also mitigate the "end of queue blocking" caused by packet losses. We know that collapsing transport and security setup in a single protocol drastically reduces the initial wait time of a connection. We know that embedding encryption and authentication inside the transport protocol provides a powerful API to applications, and also enables sophisticated handling of path redundancy and mobility.

We know all that because we have tried it before. Forward error correction for transport protocols has been advocated many times (e.g. [5], [6]) and is commonly used for sending real-time video streams. SCTP [7] demonstrated the benefits of independent stream control. SPDY [8] and QUIC [4] demonstrate how design changes can reduce the initial connection latency, something that was also attempted by Transactional TCP [9]. The authors of TCPCrypt make a strong case for ubiquitous encryption [10] and show that it could be done. We have experimented with TCP Multipath [11]. Yet, despite all that, very few of these innovations have been deployed. Some of the non-deployment may be caused by the merits of specific proposals, but the main reason is that they were not deployed because they would not successfully pass through various middle-boxes.

The effect of middle-boxes on transport protocols is obvious. Boxes are designed based on the current assumptions on "how the Internet is used," and tend to reinforce this design. For example, a variety of firewalls and NATs were deployed before the codification of Explicit Congestion Notification (ECN) [2] in 2001. These middle-boxes looked suspiciously at TCP packets in which a reserved flag was set, and dropped them. Requesting ECN support would have resulted in many connection failures, which prevented main-stream systems like Windows to support it "by default" [19]. Ten years later, in 2011, the most optimistic assessment according to a study by Steven Bauer, Robert Beverly, and Arthur Berger [3] was that 17% of servers and 4.8% of clients supported ECN. Yet ECN is probably one of the least controversial innovations that we could think of, with no impact on security or copyright. This is the very definition of "ossification."

If we want to break that ossification, we have to do something drastic.

The tussle between innovation and network value

Twelve years ago, Dave Clark, John Wroclawski, Karen Sollins and Bob Braden published "Tussle in Cyberspace: Defining Tomorrow's Internet"[1] in which they outlined that many protocol design decisions are not driven by architecture purity, but rather by "*tussles that arise among the various parties with divergent interest.*" The development of middle-boxes and their interference with transport protocol developments is very much the result of such tussles. Until now, the actors who do benefit from "end to end" transparency did not fight much, and that transparency found itself on the losing side of the tussle. The renewed emphasis on privacy and encryption gives us an opportunity to replay at least some of this tussle, and hopefully to arrive at a more balanced situation.

Each of the middle-boxes is deployed because it solves the narrow interest of some network actors, and of course the commercial interests of their manufacturers. The deployment is only limited by the global desire of the actors to not break the Internet, or at least not break it completely. As a result, we get various forms of NAT, firewalls, load balancers, intrusion detection systems, transport relays, content caches, to name a few types of middle boxes. In theory, the use of these middle-boxes and services could be negotiated between the actors of a communication: clients and their devices, servers, network providers. In practice, the deployment is unilateral. Intermediaries deploy a box "because they can," services and applications just have to "deal with it."

Let's call this the secure transport tussle, opposing middle network "value added" and end-to-end innovation towards a secure universal transport. If we look at the current state, the box vendors have won. Innovators focus on creating new applications using the existing transports. They have learned that they can rely on HTTP, because it crosses most NATs and firewalls. Yet, some examples show that the tussle can still be fought, as shown by the use of NAT traversal in voice over IP applications and in video games.

The example of the NAT traversal tussle

Unlike many other applications, VOIP and video games require minimal delays for data stream that are typically "client to client." With NAT and firewalls, they had of course the option of routing voice packets

and game clicks through a central server, and implement some form of HTTP forwarding. But they could not be content with that, because it meant some pretty bad end-to-end latency and jitter. There was thus a big pressure to “find a solution for NAT and firewall traversal.”

The first attempts at a solution were cooperative: find a way to tell the local boxes to open a port for the game, or for the VOIP application. This resulted in protocols like UPNP IGD [12] or NAT-PMP [20], which were developed outside of the IETF by the UPNP Forum and by Apple, years before the IETF developed PCP [13]. But these cooperative efforts met with only limited success. Many NAT did not implement either protocol. Even those that implemented UPNP IGD often left it blocked by default, requiring users to navigate the configuration interfaces of their home routers to set the parameters. This left application developers wanting for a more reliable solution, something they could deploy unilaterally without depending on details of the NAT configuration.

The first unilateral solution to NAT traversal was proposed by Dan Kegel, for use in videogames [14]. Instead of trying to convince NAT vendors to change their ways, he observed the behavior of the most prevalent NAT and devised an early version of what became known as “UDP hole-punching.” The same approach was then standardized in STUN [15] and Teredo [16].

The hole-punching solutions had one big advantage: they could be deployed in most networks, without asking for any permission from network managers or equipment vendors. There was of course some resistance. Deploying end-to-end solutions diminishes the effectiveness and the value of middle boxes. The often heard argument was that opening such holes was reducing the effectiveness of the firewall and exposing end-stations to unknown risks. There were regular attempts to improve the firewalls to block hole-punching, and the firewall vendors of course made sure to publicize the risks inherent to such solutions.

The NAT traversal tussle has been playing for a dozen years since the initial drafts of STUN or Teredo. It has now stabilized to a large win for NAT traversal, reinforced by extensions for combining port control and hole-punching, or better predicting NAT behavior [21]. The main remaining issue are the networks that block UDP traffic altogether. In the other cases, the solution just works. STUN and complements like ICE or TURN are widely used by VOIP services. The Xbox One uses Teredo to traverse IPv4 NAT when IPv6 is not available [17]. The manufacturers of home routers are now facing these deployments, and they know that breaking Skype or Xbox would not be a smart business move.

In short, the NAT traversal tussle demonstrated that we can score at least a partial win for end-to-end connectivity. The success is due mostly to choosing solutions that can be deployed on “the network we have, not the network we wish we had.” The solutions were adopted by powerful applications, and thus were clearly beneficial to users of these applications. Once deployed, the applications became “too big to break,” and the corresponding solutions become part of the Internet fabric.

The VOIP and game developers could manage to at least partially win the NAT traversal tussle. Can the application developers win a similar tussle, this time with secure transport?

Deploying a secure transport

There are many reasons to try develop a secure transport, which we could define as combining the properties of TCP and TLS. Of course, the simplest solution would be to do just that, run TLS on top of TCP, but that would be just keeping the status quo. An integrated solution would improve security by protecting the transport headers and reduce latency by simplifying negotiations. A good solution would also protect against various header forgery and packet injection attacks against TCP. Deploying such solution would clearly implying winning the tussle against middle boxes that attempt to "add value" by interfering with transport headers or with application content.

The TCPINC working group is chartered with designing such a solution. Predictably, the initial discussions concentrated on the need or not to protect the TCP header. On one hand, the argument is that middle-box traversal is easier to achieve if only the content is encrypted. The counter argument is of course that if TCP headers are not protected, the solution has only marginal added value compare to just running TCP and TLS. History tells us that in such conditions IETF working groups take a long time developing a compromise that does not really satisfy most of the participants. The cynical observer will notice that TCPINC is likely to play the same role in the secure transport tussle that MIDCOM played in the NAT traversal tussle.

Could application developers produce a unilateral solution that would be the secure transport equivalent of UDP hole-punching? The QUIC proposal [4] actually has many of these characteristics. A key element of the QUIC design is to send encrypted packets over UDP. The UDP header is followed by a context identifier, and then by encrypted data, split between a cryptographic hash and an encrypted content, per AEAD. Once decrypted, the content reveals the transport control information, as well as the application data.

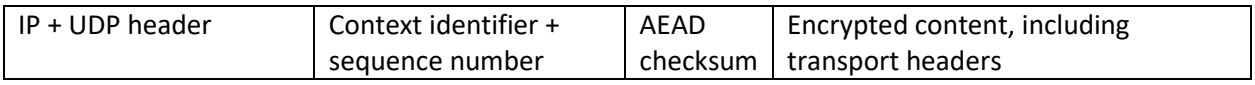

Figure 1 QUIC packet format

UDP encapsulation means that QUIC could be readily deployed in pretty much all networks that let UDP through – precisely those networks for which the NAT traversal tussle was won. The deployment requires the simultaneous deployment of clients and servers, but that can be solved by shipping the transport implementation as part of an application, something that the UDP encapsulation enables. The deployment also requires having a fallback solution to TLS and TCP for the cases where UDP cannot be used, which is very similar to what VOIP or game applications do. In practice, that fallback solution will have to be some form of HTTP or HTTPS tunneling.

UDP encapsulation also implies that QUIC will bypass most of the "TCP specific" middle boxes. The main remaining hurdle there will be to cross the load balancing devices that front the server banks. We can assume that if a large company like Google wants to deploy QUIC in its servers, it will ensure that its load balancers support UDP, but that may more difficult for small services relying on rented servers or colocation facilities.

Of course, just being deployable is not a guarantee of success. But if the QUIC protocol is well documented, and if it becomes adopted by other companies besides Google, then it could be the basis for a new generation transport protocol. The current QUIC spec delivers many of the expected transport innovation, including integrated security, multiplexing of streams, forward error correction and possibly

multipath, or at a minimum quick reconnections. The end-to-end nature of the protocol provides a path for future extensions. If these extensions provide a sizeable “user experience” advantage over the combination of TCP and TLS, the technology could well be adopted. At that point, end-to-end protocol developers would have won the “secure transport tussle.”

The secure transport tussle

Deploying a secure transport protocol like QUIC will place application developers on a collision course with vendors of managers of middle boxes. To put it simply, the encrypted packet structure reverses the balance of power between end-to-end and middle-boxes. In the current Internet, middle boxes can readily inspect and modify packets. If encryption seals these packets, the middle boxes can only recognize and block the offending traffic, a much more drastic option as it definitely affects the quality of service perceived by end users. Of course, there are different kinds of middle boxes, and we can expect different resolution of the tussle for each of them.

Most of the boxes deployed on customer premises perform only network address translation and a limited “stateful firewall” function. These boxes will not be affected by the deployment of an UDP based transport. In fact, the makers of these boxes could find the opportunity to develop new useful features, such as for example allowing port opening with PCP [13]. This would require fewer “maintenance” packets, thus preserving battery life.

There are a number of boxes deployed in the network to try manage the data flow. Some will rewrite TCP headers, adjusting for example the retransmission timers and the window sizes to try achieve better performance on satellite or wireless networks. Such boxes will not be able to operate if the packets are sealed, but they would not be needed if we can deploy optimized transports. If the transport is really efficient, the network operators may in fact welcome the opportunity to simplify their network.

The hardest fight against secure transport will probably come from the “content inspection” middle boxes. These devices are used for all kinds of purposes, from virus detection to various forms of censorship or surveillance. Of course, they will only be able to operate if they can decrypt the packets, which leaves them with the alternatives of breaking encryption, forcing users to choose non encrypted transports, or somehow cooperating with one of the parties in the connection. We can expect all these approaches to be deployed as part of the tussle, but we can also expect the secure transport to work in pretty much all the places were TLS is accepted.

The tension between encryption and content inspection is not specific to innovative transports. That tussle is already playing out with the generalized used of TLS for Internet applications. Various attackers try to break the encryption using MITM attacks, fake PKI certificates, or by somehow acquiring the public keys of services. Some authoritative regimes impose the use of national firewalls that block encrypted content. Some surveillance services are rumored to cooperate with big content providers and acquire user data. Changing the stack from using TCP+TLS to using an encrypted transport over UDP will not change that, it will just move the same old tussle to a new arena, while getting better performance and not blocking further innovation.

The role of the IETF

It is remarkable point that for the NAT traversal tussle the IETF was pretty much on both sides of the fence. Initial efforts at defining a standard NAT traversal protocol in groups like FOGLAMPS or MIDCOM did not produce a workable NAT traversal solution. MIDCOM’s efforts were slowed by continuous

debates between application developers and network managers, and the group only managed to develop an SNMP based specification that nobody used. It took 5 years and 18 draft revisions to publish Teredo, until all possible discussions and blocks could be resolved. The behavior would not have been different if the goal had been to tie the working groups in knots and ensure that it did not produce anything that went against the interest of network managers and middle box developers. However, MIDCOM also managed to publish STUN and the follow on working group BEHAVE later published TURN.

The tussle also played out in the IAB. The IAB put up a recommendation branding the unilateral solutions under the acronym UNSAF [18], which endorsed most of the arguments of middle-box manufacturers and stated the intention to "*deprecate any such workarounds when sound technical approaches are available*." Reading the statement several years later, it appears to contain two kinds of arguments. The first kind is that it is hard to build a reliable traversal service without cooperation with the middle box, but of course end-to-end application developers knew that, and considered it a "cost of doing business." The second kind of arguments is that unilateral solution expose risks by diminishing the value of middle boxes like firewalls, and clearly shows the concerns of middle box developers. Hopefully, the IAB recommendation only translated in another small administrative hurdle, one more pro-forma section to fill up in the draft describing the protocols. And, by the way, the supposed risks have not translated into real attacks so far, with phishing happening mostly through HTTP and SMTP.

If the past is a predictor, we can expect long protracted discussions if we try to develop a UDP based secure transport in the IETF. These discussions will of course be bypassed if the new protocols are developed by a small group outside of the IETF, as is already happening. The IETF may only have to intervene at a later stage, to derive a standard version from the existing running code. If there is a wide enough deployment of that running code, we can hope to contain the "standard block and tackle" that will undoubtedly be tried in the working groups.

References

[1] D. Clark, J. Wroclawski, K. Sollins and R. Braden, "*Tussle in Cyberspace: Defining Tomorrow's Internet.*" ACM SIGCOMM'02, August 19-23, 2002, Pittsburgh, Pennsylvania, USA. http://groups.csail.mit.edu/ana/Publications/PubPDFs/Tussle2002.pdf

[2] K. Ramakrishnan, S. Floyd, and D. Black, "*The Addition of Explicit Congestion Notification (ECN) to IP.*" RFC 3168, September 2001, http://tools.ietf.org/html/rfc3168.

[3] S. Bauer, R. Beverly, and A. Berger. "*Measuring the State of ECN Readiness in Servers, Clients, and Routers.*" Internet Measurement Conference 2011, http://conferences.sigcomm.org/imc/2011/docs/p171.pdf

[4] J. Roskind. "*Experimenting with QUIC*." Blog entry, Thursday, June 27, 2013. http://blog.chromium.org/2013/06/experimenting-with-quic.html.

[5] C. Huitema. "*The case for packet level FEC.*" IFIP TC6 WG6.1/6.4 Fifth International Workshop on Protocols for High-Speed Networks (PfHSN '96), 28-30 October 1996, Sophia Antipolis, France. http://huitema.net/papers/case4fec.ps

[6] D. Kliazovich, M. Bendazzoli, and F. Granelli. "*TCP-Aware Forward Error Correction for Wireless Networks."* MobiLight 2010, LNICST 45, pp. 68–77,2010. http://www.academia.edu/2699417/TCP-Aware_Forward_Error_Correction_for_Wireless_Networks

[7] R. Stewart. "*Stream Control Transmission Protocol.*" RFC 4960, September 2007. http://tools.ietf.org/html/rfc4960

[8] Various Google authors. “SPDY: An experimental protocol for a faster web.” http://dev.chromium.org/spdy/spdy-whitepaper
[9] R. Braden. “*T/TCP -- TCP Extensions for Transactions.*” RFC 1644, July 1994. http://tools.ietf.org/html/rfc1644
[10] A. Bittau, M. Hamburg, M. Handley, D. Mazieres and D. Boneh. “*The case for ubiquitous transport-level encryption.*” USENIX Security 2010. http://tcpcrypt.org/tcpcrypt.pdf
[11] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. “*TCP Extensions for Multipath Operation with Multiple Addresses”* RFC 6824, January 2013. http://tools.ietf.org/html/rfc6824
[12] UPNP Forum. “Internet Gateway Device (IGD) V 2.0.” http://upnp.org/specs/gw/igd2/
[13] D. Wing, S. Cheshire, M. Boucadair, R. Penno and P. Selkirk. “*Port Control Protocol (PCP).*” RFC 6887, April 2013. http://tools.ietf.org/html/rfc6887
[14] D. Kegel. “NAT and Peer-to-peer networking.” Blog entry, July 1999. http://alumnus.caltech.edu/~dank/peer-nat.html
[15] J. Rosenberg, J. Weinberger, C. Huitema and R. Mahy. “*STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs).*” RFC 3489, March 2003. http://tools.ietf.org/html/rfc3489
[16] C.Huitema. “*Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs).*” RFC 4380, February 2006. http://www.ietf.org/rfc/rfc4380.txt
[17] C.Palmer. “Xbox One: P2P, IPv6, Teredo, and IPsec.” Presentation to Nanog 59. http://www.nanog.org/sites/default/files/wed.general.palmer.xbox_.47.pdf
[18] L. Daigle. “*IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation.*” RFC 3424, November 2002. http://tools.ietf.org/html/rfc3424
[19] M. Sridharan, D. Bansal, and D. Thaler. “*Implementation Report on Experiences with Various TCP RFCs.*” Microsoft presentation to IETF 68, TSV area. http://www.ietf.org/proceedings/07mar/slides/tsvarea-3/tsvarea-3.ppt
[20] S. Cheshire, and M. Krochmal. “*NAT Port Mapping Protocol (NAT-PMP).*” Published by Apple as RFC 6886, April 2013. http://tools.ietf.org/html/rfc6886
[21] D. Thaler. “*Teredo Extensions.*” RFC 6081, January 2011. http://tools.ietf.org/html/rfc6081