Survey On The Current State Of Tor Over QUIC

Mohamed Mehdi Gharam, Lion Steger*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: mehdi.gharam@tum.de , stegerl@net.in.tum.de

***Abstract*—Despite its popularity, Tor is currently bugged with high latency and other performance issues. In response, a transition in Tor's transport layer protocol to QUIC has been proposed, aiming to solve many of the problems caused by TCP's inherent limitations.**
In this paper, we present a literature survey on the current state of Tor over QUIC. We examine proposed designs and evaluate their impact on performance and security. We conclude that transitioning Tor to QUIC holds significant potential, yet further work on fully examining the security and performance impacts of such a transition still remains to be done.

***Index Terms*—Tor, QUIC**

1. Introduction

Anonymity networks have become a very important tool to face the increasing trend of tracking users and infringing on their privacy rights. Tor is the most popular anonymity network, currently used by over 3 Million clients [1]. Yet, it has long suffered from performance problems that stood in the face of its adoption. Many of the underlying causes, such as Head-of-Line blocking, unfairness in bandwidth allocation, and inefficient congestion control are caused by multiplexing Tor circuits over TCP connections [2]. While a lot of research has been done on improving Tor, most attempts have failed because they either compromise security or introduce additional performance overhead [3]. This motivates the choice behind a change in Tor's transport layer protocol to the UDP-based QUIC.
In this paper, we present two proposed designs for a Tor over QUIC implementation: an end-to-end design proposed by the Tor community that aims to improve end-to-end congestion control, and a hop-by-hop design adopted by the research community that aims to use QUIC's features to improve fairness and decrease latency. We then proceed to evaluate these designs from a security and performance perspective while giving an overview of the current state of research in the process.

2. Background

In this section, we introduce the most important concepts discussed in this paper. We give a brief overview of how Tor works, introduce the QUIC communication protocol, and go over the reasons behind changing Tor's transport layer to QUIC.

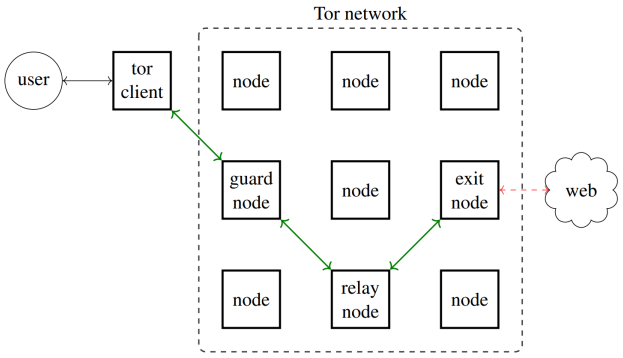

Figure 1: A visualization of a Tor circuit. Taken from [5].

2.1. TOR

Tor is a low-latency anonymity network that allows users to communicate online without exposing their identity [4]. It does this by relaying all traffic over multiple nodes in a process known as *Onion Routing*. Onion Routing allows the anonymisation of TCP traffic by rerouting it over multiple volunteer relays, known as *Onion Routers* (OR). By default, Tor uses three of these routers to build what is called a *circuit*. A circuit is first established when a client, also called an *Onion Proxy* (OP), attempts to connect to a server and proceeds to pick 3 Onion Routers: *a guard node, a middle (or relay) node,* and *an exit node*, as shown in Figure 1.
A TLS-protected TCP connection is then formed between each part of the circuit and the node directly following it. Each node of the circuit only knows its immediate predecessor and successor, making it so that only the guard node is aware of the client's identity and only the exit node knows the server. The middle node then ensures that the guard and exit nodes are unaware of each other. To further guarantee anonymity, messages are encrypted multiple times using pre-exchanged symmetric keys with the onion routers, with each OR adding or only removing its own layer of encryption so that messages are only revealed on the last node, by which time the sender's identity is already anonymous.
It's important to mention that there's only one TCP connection between Tor nodes. This means that circuits belonging to different clients are multiplexed over the same OR-OR connection. This has the obvious performance benefit of saving the time needed to establish such a connection every time it is needed, and it also increases the security of the network, as it makes it significantly harder to identify traffic of different circuits if the connection

 doi: 10.2313/NET-2023-11-1_14

is compromised. On the other hand, Tor also multiplexes several end-to-end TCP streams belonging to the same OP into the same circuit since establishing a circuit for each stream would increase latency.

2.2. QUIC

QUIC is a transport layer network protocol based on UDP, developed by Google as a new alternative to TCP [6]. It aims to solve many of the problems caused by TCP and comes with several features, such as integrated TLS1.3 encryption, user space implementation, and middlebox resistance through encryption of packet headers, making it easier to roll out updates. It also attempts to improve latency through decreasing handshake delay by reducing the required round-trip time (RTT) to establish a connection. Since security is an integral part of Tor, it is important to mention that the offered 0-RTT handshake raises some security concerns, as extra protection against replay attacks is needed [7].
Most importantly, QUIC offers native support for multiplexing multiple data streams over a single connection. These streams represent bidirectional byte-streams, and each comes with its own priority, flow control, and congestion control. Re-transmission also occurs on a per-stream basis so that dropped packets in a stream do not inhibit the performance of other streams sharing the same connection, effectively solving the Head-of-Line blocking problem outlined in Section 2.3.1.

2.3. The Current State of Tor

Tor has long suffered from serious performance problems. Some of the notable causes include ineffective congestion control, unfair path selection, and low network capacity [2]. And while a lot of research efforts have gone into addressing these issues, many of them proved either ineffective or raised security concerns, proving the need for further research on the matter [3]. In this subsection, we will go over two particular problems affecting Tor's performance that can be potentially solved using QUIC.

2.3.1. Head-Of-Line Blocking. Head-Of-Line (HoL) blocking is a problem that occurs because of TCP's in-order property, which ensures that data is received in the same order it was sent in. When a TCP segment is lost, all succeeding packets must wait for successful re-transmission. Since multiple Tor circuits are multiplexed over a single TCP connection, this puts a halt to data transmission over all circuits, even if they are unrelated to the lost segment. As shown in Figure 2, if a high-bandwidth, high-latency stream *(e.g. bulk transfer)* shares a TCP connection with an unrelated low-bandwidth, low-latency stream *(e.g. web browsing)*, the latter could experience throttling, affecting the fairness of the network [2]. Basyoni *et al.* mention in [8] that as the popularity of Tor increases, it is expected that this will occur more frequently.
Nowlan *et al.* [9] attempted to address this with uTor: a Tor implementation that used un-ordered TCP to ensure that data is transmitted despite lost packets. However, this approach showed "modest performance gains", likely because of the additional processing of packets it required [2].

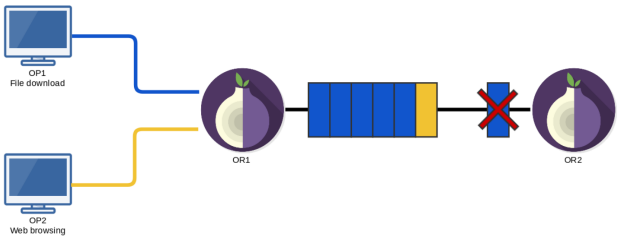

Figure 2: The Head-of-Line blocking problem in Tor. The blocks represent packets, their color corresponding to the originating client. Since one of *OP1*'s packets is lost, all other packets in transit are blocked until successful re-transmission, including *OP2*'s unrelated packet. Taken from [5].

2.3.2. Congestion Control. Ineffective congestion control is a major contributor to Tor's performance issues. Hop-by-hop congestion is managed by the TCP connections between nodes. However, due to Tor multiplexing multiple circuits over a single TCP connection, any slowdown caused by congestion will affect all the circuits over the connection, causing the same effects discussed in the previous section (2.3.1).
More importantly, end-to-end congestion control is simply ineffective. Unlike circuit level congestion control, there is no end-to-end TCP connection between the client and the exit relay. Because Tor connections are composed of multiple independent TCP connections, Tor does not currently have a "low latency method of informing the client of congestion in later links of the circuit" [10]. When a Tor relay receives a cell, it is required to forward it reliably. This makes it unable to drop the cell and causes its buffer to fill up when it receives more than it can send. Combined with other scheduling issues [11], [12], this results in relays being overburdened, with no effective way of informing the client to decrease its sending rate [10]. Currently, Tor uses a sliding window mechanism for end-to-end congestion control, but it has not been effective at reducing latency [13].

3. Proposed Designs for Tor over QUIC

In this section, we will go over the current proposed designs for Tor over QUIC that were briefly mentioned in the previous section. Most of the research community has adopted the hop-by-hop design, making use of QUIC's powerful streams to replace the existing TCP connections with QUIC ones [5], [8], [10], [14]–[16]. On the other hand, members of the Tor community have proposed an alternative end-to-end design that aims to solve Tor's congestion control problems [17]. In this section, we will examine these designs, their purposes, and their drawbacks.

3.1. End-To-End

Members of the Tor community have taken a very different approach to the research community, strongly deviating from Tor's current design that was presented in Section 2.1. In this design [17], Tor's circuit is replaced by a single end-to-end QUIC connection between the client

 doi: 10.2313/NET-2023-11-1_14

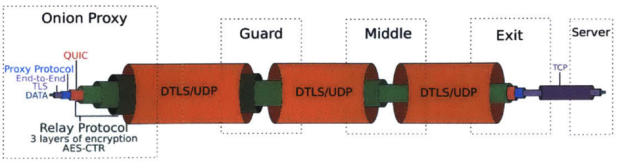

Figure 3: End-To-End Design. Taken from [10].

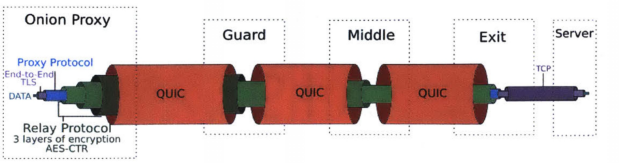

Figure 4: Hop-By-Hop Design. Taken from [10].

and the exit node and the TCP connections between Onion Routers are replaced with DTLS connections. DTLS is a protocol based on TLS that aims to secure datagram-based communication, such as UDP traffic [18]. This enables an unreliable and faster encrypted connection between relays, leaving congestion control to the inner traffic layers. The client and the exit node are then able to establish a QUIC connection by sending QUIC packets packaged into onion-encrypted Tor cells through the circuit (Figure 3). At the end of the circuit, the exit node then converts the connection back to TCP to communicate with the server. This design aims to solve Tor's inefficient congestion control by adding end-to-end congestion feedback between the client and the exit node, making use of QUIC's powerful and flexible congestion control.
Research has proven that congestion and other queuing-related problems are significant contributors to Tor's latency, even more than HoL blocking [11], [12], [19]. But while this design may have the greatest potential to improve Tor's performance, it comes with serious drawbacks. Kyle H. [10] points out several major flaws within this design. First, it completely changes Tor's original design, making it difficult to implement and deploy. This means that it could compromise Tor's security by causing unintended side effects. In fact, Sy *et al.* [20] show that end-to-end connections over QUIC facilitate web tracking, potentially compromising client anonymity. On the other hand, Tscorsch *et al.* [19] showed that end-to-end paths with high round trip times can impact fairness and increase latency due to longer reaction times.

3.2. Hop-By-Hop

Most existing proposals from the research community follow the same hop-by-hop design, staying faithful to Tor's current protocol [5], [8], [10], [14]–[16]. In this design, Tor's existing TCP connections are simply replaced by QUIC connections, and different circuits are multiplexed over different QUIC-streams within the same connection. Note that the connection between the exit node and the server has been kept as it is, as shown in Figure 4. The reason for this is that many servers do not support QUIC yet. In fact, QUIC traffic only represented 7% of all internet traffic in 2017 [21].
This approach has two major advantages: It immediately solves the HoL blocking problem because QUIC provides several logical streams over a single connection, making it so that loss of data in a stream does not affect the others. Furthermore, QUIC comes with a pluggable congestion control module that can be configured separately for each stream. This can be used to improve fairness between multiple Tor circuits, as congestion on one connection no longer affects all streams that use that connection.
However, it is important to note that this does not solve most of Tor's congestion control problems that were discussed in Section 2.3.2. In fact, most implementations of this design completely ignore end-to-end congestion and do nothing to address it. While the end-to-end design has its fair share of problems, it can still serve as inspiration on how to fix Tor's lack of end-to-end congestion control. Kyle H. [10] was the first to consider the potential for backpressure-based congestion control using QUIC. Essentially, this was achieved using QUIC's per-stream flow control, by allowing Onion Routers that experience congestion caused by a specific circuit to reduce the maximum data it accepts from the stream associated with that circuit. This, in turn, forces the previous relay to reduce its sending rate, potentially propagating this information back until the client's sending rate is reduced.

4. Results

Ideally, a Tor over QUIC implementation would solve the HoL problem, improve fairness in the network, improve congestion control, and ultimately not compromise Tor's security. In this section, we will proceed to examine whether these design goals have been successfully achieved or not, from both a performance and security standpoint.
Note that we will only examine the hop-by-hop design, as no implementations currently exist for the end-to-end one.

4.1. Performance

In this subsection, we will focus on the performance evaluation of the Tor over QUIC implementations proposed by Basyoni *et al.* and J. Heijligers [8], [16], as they are the most recent and come with detailed performance evaluations.
There are 2 important metrics that are used to evaluate the performance of a Tor implementation. *Time to First Byte* (TTFB) represents the time it takes for the client to establish a circuit and receive its first byte. Similarly, *Time to Last Byte* (TTLB) represents the time it takes for the client to establish a circuit and receive the last byte from the server.
All performance measurements outlined in the section follow the model laid out by Jansen *et al.* in [22] for accurate performance evaluations. Specifically, there were two types of simulated clients: low-bandwidth clients sending regular HTTP requests to represent web browsing and high-bandwidth clients performing bulk downloads (*e.g.* over BitTorrent).
QuicTor [8] reported very minimal improvement regarding TTFB for low-bandwidth clients. However, it managed to

 doi: 10.2313/NET-2023-11-1_14

reduce the TTLB for high-bandwidth clients by almost 80%, likely because these clients are more prone to being affected by HoL blocking. It also reported significant improvements for video streaming applications. Overall, it managed to outperform vanilla Tor in all scenarios and even mostly outperformed two other Tor implementations designed to address the problem of circuit multiplexing [23], [24].
Similarly, J. Heijligers [16] reported a 50% performance improvement over vanilla Tor. It also showed more fairness in distributing bandwidth among clients, reaching a near-perfect score on Jain's fairness index [25]. Although these measurements may not be perfect and do not exactly simulate real-time conditions, they show very promising results. It's also worth mentioning that these improvements are mostly attributed to improved circuit-level congestion control, since the hop-by-hop design does not address end-to-end congestion control.

4.2. Security and Privacy

Security and Privacy are fundamental aspects of Tor, as client anonymity is the major goal of the network. It is therefore important that Tor over QUIC implementations do not compromise security, either through exposing new attack vectors, aggravating existing ones, or undermining Tor's current defense mechanisms. In this subsection, we will focus on examining [8] and [10], as they include a comprehensive security analysis.
QuicTor [8] investigated the impact of switching to QUIC on *traffic correlation* attacks, as they are more likely to be impacted by a change in the transport layer protocol. In such an attack, an adversary would try to correlate traffic observed at one of Tor's nodes and one of its endpoints to deanonymize the client. The authors implemented two known attacks [26], [27] and observed that QuicTor did not behave much differently than vanilla Tor, even showing better resistance in some cases. This is due to timing-based attacks becoming less effective because of QUIC since they assume that one stream can influence all other streams passing through the same node [26].
Kyle H. [10] provided a more in-depth security analysis of his proposal by investigating whether QUIC leaks any extra information that may be used for an attack vector. The author concluded that it was important not to use more than a QUIC-stream per Tor-circuit, as per-object streams could leak detailed information about client traffic. 0-RTT connections also come with some security concerns, as they can allow client-tracking across different sessions, but countermeasures against this exist [20].
Furthermore, the author investigated whether his proposal for backpressure-based congestion control comes with any security risks, and concluded that these could be mitigated by only backpropagating limited information at set intervals.
However, some concerns remain about *website fingerprinting* (WF) attacks. These aim to analyze traffic to conclude which website the client visited. Nie *et al.* [28] developed QUIC-CNN, a novel model for WF attacks on QUIC traffic on Tor, and even found that it performs better than the current state-of-the-art model.

5. Conclusion and Future Work

In this work, we presented a literature review over the current state of Tor over QUIC. Tor currently has significant performance problems, and it is the main motive behind this change in its transport layer protocol to solve them without compromising security.
We examined two proposed designs: an end-to-end design that tunnels a QUIC connection between the client and the exit node, aiming to solve Tor's congestion related issues, and a hop-By-hop design that replaces Tor's current TCP connections with QUIC ones, aiming to solve the Head-of-Line blocking problem and improve fairness. Although the former is riddled with performance and security concerns, an ideal Tor over QUIC implementation should still strive for its goal of improving end-to-end flow control. We presented one design proposal [10] that suggested doing this through backpropagation of flow control information. While performance tests have been promising so far, more work still needs to be done on the matter. The aforementioned design proposal has still not been implemented, and its claims about improving performance still need to be corroborated. Furthermore, the security behind these designs still needs to be studied, as current research only provided a purely theoretical analysis or only considered one specific type of attack vectors.
Although these designs seem promising, we still have a long way to go before Tor finally switches over to QUIC.

References

[1] "Welcome to Tor Metrics," https://metrics.torproject.org, [Accessed 30-May-2023].

[2] S. J. M. Roger Dingledine, "Performance Improvements on Tor or, Why Tor is slow and what we're going to do about it," *Tor Tech Report 2009-11-001*, [Online; accessed 15/05/2023; https://research.torproject.org/techreports/performance-2009-11-09.pdf].

[3] I. G. Mashael AlSabah, "Performance and Security Improvements for Tor: A Survey," *ACM Comput. Surv.*, vol. 49, no. 2, 2016.

[4] R. Dingledine and N. Mathewson, "Tor Protocol Specification," (visited on 28-05-2023). [Online]. Available: https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt

[5] W. Sabée, "Adding QUIC support to the Tor network," 2019.

[6] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. B. Krasic, C. Shi, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. C. Dorfman, J. Roskind, J. Kulik, P. G. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, and W.-T. Chang, "The QUIC Transport Protocol: Design and Internet-Scale Deployment," 2017.

[7] M. Fischlin and F. Günther, "Replay Attacks on Zero Round-Trip Time: The Case of the TLS 1.3 Handshake Candidates," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 60–75.

[8] L. Basyoni, A. Erbad, M. Alsabah, N. Fetais, A. Mohamed, and M. Guizani, "QuicTor: Enhancing Tor for Real-Time Communication Using QUIC Transport Protocol," *IEEE Access*, vol. 9, pp. 28 769–28 784, 2021.

[9] M. F. Nowlan, D. I. Wolinsky, and B. Ford, "Reducing Latency in Tor Circuits with Unordered Delivery," in *3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)*. Washington, D.C.: USENIX Association, Aug. 2013. [Online]. Available: https://www.usenix.org/conference/foci13/workshop-program/presentation/nowlan

[10] K. Hogan, "Security analysis of Tor over QUIC," 2020.

[11] Rob Jansen and John Geddes and Chris Wacek and Micah Sherr and Paul Syverson, "Never been KIST: Tor's congestion management blossoms with Kernel-Informed socket transport," in *23rd USENIX Security Symposium (USENIX Security 14)*. San Diego, CA: USENIX Association, Aug. 2014, pp. 127–142. [Online]. Available: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/jansen

[12] R. Jansen and M. Traudt, "Tor's Been KIST: A Case Study of Transitioning Tor Research to Practice," 2017.

[13] F. Tschorsch and B. Scheuermann, "Mind the gap: towards a backpressure-based transport protocol for the Tor network," *Networked Systems Design and Implementation*, Mar 2016.

[14] R. Aissaoui, O. Erdene-Ochir, M. Al-Sabah, and A. Erbad, "QUTor: QUIC-based Transport Architecture for Anonymous Communication Overlay Networks," in *Qatar Foundation Annual Research Conference Proceedings Volume 2016 Issue 1*, vol. 2016, no. 1. Hamad bin Khalifa University Press (HBKU Press), 2016, p. ICTPP2961.

[15] A. Clark, "Quux: a quic un-multiplexing of the tor relay transport," 2016.

[16] J. Heijligers, "Tor over QUIC," 2021.

[17] M. Perry, "[tor-dev] The case for Tor-over-QUIC," [Online; accessed 22/05/2023; https://lists.torproject.org/pipermail/tor-dev/2018-March/013026.html].

[18] E. Rescorla, H. Tschofenig, and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3," RFC 9147, Apr. 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9147

[19] F. Tschorsch and B. Scheuermann, "How (not) to build a transport layer for anonymity overlays," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 101–106, 2013.

[20] Sy, Erik and Burkert, Christian and Federrath, Hannes and Fischer, Mathias, "A quic look at web tracking." *Proc. Priv. Enhancing Technol.*, vol. 2019, no. 3, pp. 255–266, 2019.

[21] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W.-T. Chang, and Z. Shi, "The QUIC Transport Protocol: Design and Internet-Scale Deployment," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 183–196. [Online]. Available: https://doi.org/10.1145/3098822.3098842

[22] R. Jansen, K. Bauer, N. Hopper, and R. Dingledine, "Methodically Modeling the Tor Network," in *5th Workshop on Cyber Security Experimentation and Test (CSET 12)*. Bellevue, WA: USENIX Association, Aug. 2012. [Online]. Available: https://www.usenix.org/conference/cset12/workshop-program/presentation/Jansen

[23] M. AlSabah and I. Goldberg, "PCTCP: Per-circuit TCP-over-IPsec transport for anonymous communication overlay networks," 11 2013, pp. 349–360.

[24] J. Geddes, R. Jansen, and N. Hopper, "IMUX: Managing Tor Connections from Two to Infinity, and Beyond," 11 2014, pp. 181–190.

[25] R. Jain, D. M. Chiu, and H. WR, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems," *CoRR*, vol. cs.NI/9809099, 01 1998.

[26] S. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *2005 IEEE Symposium on Security and Privacy (S&P'05)*, 2005, pp. 183–195.

[27] P. Mittal, A. Khurshid, J. Juen, M. Caesar, and N. Borisov, "Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting," in *Proceedings of the 18th ACM conference on Computer and Communications Security*, 2011, pp. 215–226.

[28] M. Nie, F. Zou, Y. Qin, T. Zheng, and Y. Wu, "QUIC-CNN: Website Fingerprinting for QUIC Traffic in Tor Network," in *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, 2022, pp. 663–671.