QUIC and HTTP/3 Debugging Survey - March 2020

Thank you for participating in this survey. It is mainly aimed at IETF QUIC/H3 implementers, their colleagues and researchers who are testing and optimizing early IETF QUIC stacks. Our goal is to get insight into how people approach debugging and evaluating these new implementations and how they make use of existing tools. We assume some familiarity with QUIC/H3 and existing debugging options, though we will provide additional information where necessary.

The survey focuses specifically on endpoint/decrypted logging formats and tooling. Other debugging options (such as quictracker, QUIC network simulator, spindump, etc.) are left for follow-up research. The survey consists of three parts: The first considers logging formats, the second tooling and visualizations and the last (future) use cases for these debugging approaches. We expect it will take you 15-20 minutes to fill out the survey, all of the questions are optional. We would prefer to hear from you before April 1st.

We encourage you to be honest and to provide constructive feedback to help us improve the current offerings in logging and tooling options.

We go to great lengths to ensure the privacy of the participants (and their employers). While the results of this survey are intended for scientific publication, all data will be fully anonymized and no direct names of participants or their companies will be used. We will mainly use aggregated data from all participants. Direct quotes or text snippets of anecdotal answers are edited manually where necessary to remove direct hints to their specific contexts. Further information on privacy and GDPR compliance can be found at: https://quic.edm.uhasselt.be/files/survey2/QUICDebuggingSurveyMarch2020_GDPRInformedConsent.pdf

1/3: Endpoint logging

This section will consider logging of QUIC/H3 implementation behavior. In contrast to TCP, QUIC Is end-to-end encrypted at the transport layer. This means that merely capturing encrypted QUIC packets on the wire provides much less usable information than comparable TCP+TLS packet traces (where much of the transport metadata is sent in plaintext).

As such, QUIC will have to shift to logging at (or supported by) the endpoints themselves. This can be done by the endpoints either providing decryption keys for external packet traces (for example by exporting TLS secrets) or by them outputting internal implementation state directly. This latter approach provides several benefits over using direct packet captures: it can include information not sent on the wire (e.g., internal congestion controller state) and it can be smaller in size (QUIC packet captures need to be stored -in full- first and can only be trimmed after decryption).

For the direct endpoint logging, there are two main options:

(A) Unstructured logging: the typical command line stdout/stderr output when running the implementation directly. We call this unstructured, because it is typically logged in an ad-hoc format, different for each implementation, and not easily consumed by scripts for further processing. Most implementations provide this type of output. An example would be https://quic.edm.uhasselt.be/files/survey2/unstructured_log_example.html

(B) Structured logging: information is logged according to a well-defined schema and is the same across implementations. The output is in a machine-readable format (e.g., JSON, protobuf, ...). Some available options for this are qlog (https://github.com/quiclog/internet-drafts) and quictrace (https://github.com/google/quic-trace). An example of a qlog file would be https://quic.edm.uhasselt.be/files/survey2/structured_log_example.qlog.txt

1. What types of logging are available in your implementation of choice?

Check all that apply.

- [ ] decrypted packet captures (exporting of TLS secrets via e.g., SSLKEYLOGFILE)
- [ ] qlog (structured)
- [ ] quictrace (structured)
- [ ] custom format (structured) (e.g., in-house event tracing framework)
- [ ] custom format (unstructured) (e.g., command line output)
- [ ] Other: _____

2. If you have implemented a structured logging format: why did you decide to do so? What are the key benefits?

   If you implement a custom, in-house format, please explain why you prefer that above other options.

3. If you have not implemented a structured logging format: why not? Do you plan to do so in the future?

   We are looking for concrete problems, both with currently available formats and with the general principle of structured logging.

4. Would you consider completely replacing your unstructured logging with purely structured logging? Why (not)?

   For example, qlog specifies events that correspond to typical logging categories like error, warning, info, debug and verbose (see https://tools.ietf.org/html/draft-marx-qlog-event-definitions-quic-h3-01#section-7)

5. How important are the following aspects of a structured logging format?

There has been a large discussion around using a textual format (such as JSON or csv) vs a binary format (such as protobuffers or flatbuffers). We want to figure out the correct tradeoffs.

Mark only one oval per row.

	Not important	Of little importance	Of average importance	Very important	Absolutely essential
Small file size	◯	◯	◯	◯	◯
Easy integration in QUIC implementation (e.g., availability of libraries)	◯	◯	◯	◯	◯
Streamable (files do not need to be read/written in full)	◯	◯	◯	◯	◯
'Grep'-able output (direct string search on a file)	◯	◯	◯	◯	◯
(De)serialization performance	◯	◯	◯	◯	◯
Ability to log raw packet (payload) data	◯	◯	◯	◯	◯
Easy to load in (web-based) tooling	◯	◯	◯	◯	◯
Ability to easily log new custom event types/categories (flexibility)	◯	◯	◯	◯	◯

6. Do you use (public) logs from other implementations?

When testing other implementations, you typically point your client at a public server endpoint. Some implementers have made (both structured and unstructured) server-side logs publicly available for easy sharing. For example: https://quic.aiortc.org/logs. Similarly, projects like QUIC network simulator ("interop runner") store logs for both actors in their simulations. It is however unclear how much people actually look at logs from other implementations.

Check all that apply.

- [ ] No or only rarely (I typically ask the other implementers to identify possible errors)
- [ ] Yes, but mainly unstructured logs
- [ ] Yes, but mainly structured logs (e.g., in combination with qvis)
- [ ] Yes, both unstructured and structured logs

Other: [ ] _____

7. If you indeed use logs from other implementations, please provide some additional insight into when especially this is useful to you

_____

_____

_____

_____

_____

2/3: Tooling and visualizations

This section considers tools and visualizations that can help explore the logs discussed in the previous section in a deeper way than just the command line interface or a text editor.

Sadly, there are not too many dedicated tools available. The options range from relatively straightforward viewers (like wireshark) to more complex, special purpose visualizations. In this latter category, we are mainly aware of two public/open source options:

- The online qvis toolsuite (https://qvis.edm.uhasselt.be) with 4 visualizations
- The quictrace tool (https://github.com/google/quic-trace) for debugging congestion controller behaviour

8. How often do you use the following tools/visualizations when working with QUIC/H3 implementations?

Please mainly consider your usage over the last few months

Mark only one oval per row.

	Never	Rarely	Occasionally	Frequently	Very frequently
Texteditor / command line interface	⬭	⬭	⬭	⬭	⬭
Wireshark	⬭	⬭	⬭	⬭	⬭
qvis sequence diagram	⬭	⬭	⬭	⬭	⬭
qvis congestion graph	⬭	⬭	⬭	⬭	⬭
qvis multiplexing graph	⬭	⬭	⬭	⬭	⬭
qvis statistics overview	⬭	⬭	⬭	⬭	⬭
quictrace congestion graph	⬭	⬭	⬭	⬭	⬭
Custom, in-house tooling	⬭	⬭	⬭	⬭	⬭

9. If you have used custom tools, or public tools not in the above list, please provide additional information here.

Please describe the tools' features in as much detail as possible. Please also indicate which logging formats the tools accept as their input.

10. What tools (or features of tools) that currently do not exist yet would be most useful to you?

This will help us prioritize to choose which tools to work on next.

11. Do you have some examples of bugs or issues that additional tools helped you to identify more easily than if you had not used tools? Or issues that you would probably not have found without extra tooling?

We are looking for the types of issues that really require additional tooling to be detected or solved.

12. How important is the availability of public/open-source tools to your choice of logging format?

Mark only one oval.

- ◯ Not important (we (plan to) employ mainly custom tools)
- ◯ Neutral (it does not influence our choice of logging format)
- ◯ Important (we use outside tooling and/or adapt open-source projects to our needs)
- ◯ Very important (we have implemented a type of logging only to be able to use outside tooling)
- ◯ Other: _____

3/3: (future) use cases

In this section, we want to get a feel for the further potential of the logging and tooling approaches discussed before. At this point, most are mainly in the initial development phase for the QUIC/H3 protocols, while only some are already in the active deployment and optimization phase.

13. For which use cases do you envision using structured logging and extra tooling?

Mark only one oval per row.

	Never	Seldom	Sometimes	Often	Almost always
Teaching new people (students, interns, junior team members)					
Debugging newly implemented features (e.g., multipath, new application layer protocols)					
Debugging live deployment issues/bugs (e.g., problems with consumer traffic)					
Evaluation, testing and optimization of (live) deployments (e.g., A/B testing different configurations)					
(Academic) research					

14. How important of a feature is structured logging (and subsequent tooling support) in your overall offering/strategy?

For example, do you plan to provide and promote this as a commercial feature of your QUIC/H3 product? Or do you conversely plan to disable it for actual deployments and only use it for (local) debugging? Please share your reasoning.

15. If you have implemented a structured logging format (such as qlog), would you consider to (or already plan to) remove it at some point? Why (not)?

16. Do you feel structured logging would be useful for non-QUIC/H3 setups? Why (not)?

For example, we have been working on qlog schemas for TCP + TLS + HTTP/2 as well and envision support for DNS-over-HTTPS and DNS-over-QUIC.

17. Is there anything else you would like to share?

As this is the end of the survey, feel free to mention points you feel are important that we forgot to ask about during the survey

Wrap-up

This final short section allows you to provide a bit more personal information and context, primarily for future follow-up. All this information is optional.

None of this section's information will be made public and it will be removed from storage after processing of the survey results. You or your company will not be named directly in any publication or public data repositories.

For further information on our data processing pipeline in the context of the GDPR, please refer to https://quic.edm.uhasselt.be/files/survey2/QUICDebuggingSurveyMarch2020_GDPRInformedConsent.pdf

18. Please share some details on your setup (optional)

If possible, let us know which specific QUIC implementation you are working with, for which company, at which scale, etc. This helps us understand in which situations our proposed logging and tooling is most useful.

19. Your e-mail address (optional)

We might have some follow-up questions for you. If you would be willing to participate further, please leave your email address here.

20. Unique participant token (optional)

For privacy reasons, we will remove any and all personally identifiable information from the survey results after processing. This in turn makes it difficult to allow participants to request to change or remove their data after the fact. If you would like to retain the option to alter your data, please provide us with a unique token (preferably a random string) that you can later use to identify your survey entry with.