Computational Science and Engineering
(International Master's Program)

Technische Universität München

Master's Thesis

Automated IETF QUIC Interoperability Matrix

Angelin Rashmi Antony Rajan

Computational Science and Engineering
(International Master's Program)

Technische Universität München

Master's Thesis

Automated IETF QUIC Interoperability Matrix

Author:	Angelin Rashmi Antony Rajan
1st examiner:	Univ.-Prof. Dr. Qui-Gon Jinn
2nd examiner:	Univ.-Prof. Dr. Obi-Wan Kenobi
Assistant advisor:	Dr. rer. nat. Anakin Skywalker
Submission Date:	April 1st, 2222

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

April 1st, 2222   Angelin Rashmi Antony Rajan

Acknowledgments

If someone helped you or supported you through your studies, this page is a good place to tell them how thankful you are.

"People sometimes ask me if it is a sin in the Church of Emacs to use vi. Using a free version of vi is not a sin; it is a penance. So happy hacking"

-Richard Stallman

Abstract

This document will serve as an example to you, of how to use LaTeXto write your CSE Master's Thesis. It will have examples and recomendations, and hopefully a few laughs. Because this is the abstract, it will have to convince you that this template is something you want to use. It has been proven, that without using this template, writing your thesis will be much more dificult. The template is based on previous work, and has been improved apon and updated. The result of this template is a modern latex template that everyone can contribute to and use for their studies of CSE @ TUM.

Some more great abstract tips can be found here: Great Abstract tips

Contents

Part I.

Introduction and Background Theory

1. Introduction

To check if the any two implementations inter operate with each other, we can use the below ways to validate the test cases

- Using Server and Client Logs

  Each implementation logs the states, packet sent/received, error message in their own way. There is no standard strucrure followed in these implementations. Thus, they require a lot of time in understading how each implementation follow their logging mechanism.

- Using Wireshark

  The current stable version of wireshark is 2.4.5. This still supports only Google QUIC. The latest development version 2.5.1 supports IETF QUIC. 0-RTT decryption is still not supported, so we will not be able to use Wireshark for testing that. For testing version negotiation, handhsake, and stream data wire shark packet capture can be used.

1.1. 0-RTT

ngtcp2 provides a way to resume a session and send 0-RTT packets in their example/client implementation. This is done by first establishing a connection with a server using the below command

./examples/client 127.0.0.1 4444 –session-file /Users/Rashmi/Documents/workspace/sample/ngtcp2 –tp-file /Users/Rashmi/Documents/workspace/sample/ngtcp2/tp.txt

The above command stores the transport parameter and session ticket locally. This can later be used for resuming the session and sending 0-RTT packet.

1.2. Version Negotiation

Each implementation development happens at a different pace ie one implemetation can be much complete than other implementation. This makes the testing process difficult. Most(???) of the QUIC implementations are not backward compatible. So if implementation1 currently supports only draft-09 and implementation2 supports draft-10, then they cannot be tested against each other as the Version Negotiation always fails. This is bound to happen when both the draft specification and implementation is continously evolving.

Part II.

Body: What Was Done for the Thesis

Part III.

Results and Conclusion

Appendix

A. Detailed Descriptions