Parsing the QUIC Packet Description Language

Ryan Murphy - 2385677M

December 16, 2021

1 Status report

1.1 Proposal

1.1.1 Motivation

Parsing RFC documents opens up countless possibilities for automation; specifically automatic code generation from packet descriptions. However, many RFC documents do not follow the same standards, so it is difficult to write a generic parser that covers all cases. Automatic code generation will reduce inconsistencies and errors when implementing protocols in software, so adding support for different RFC document structures will increase the coverage of protocols that can be represented.

1.1.2 Aims

This project will extend an automatic code generation codebase to parse QUIC-specific packet description structures from RFC documents. It is likely that protocols developed in the future will adopt qualities from QUIC, so it is important that the program can parse QUIC structures to support automatic code generation for future protocols.

1.2 Progress

- Learned existing codebase and Parlsey basics
- Built the Parsley grammar file to parse QUIC-specific structures
- Implemented custom parser utilising the Parlsey grammar file
- Converted parsed data into internal representation
- Wrappers built to generate internal representation objects straight from the grammar file
- Built initial version of the parsed representation, adding a layer of abstraction to the internal representation with less constraints and speeding up the codebase significantly
- Conversion of parsed representation to the internal representation
- Basic working code generation using parsed representation conversions
- Extended QUIC packet description language to support Enum definitions
- Added parsable variable-length integer definitions to the QUIC RFC using enums and structures

1.3 Problems and risks

1.3.1 Problems

Problems so far:

- Code was hard to read and largely undocumented, a lot of debugging was necessary to understand each file and its role in parsing data
- Internal representation constraints were too restrictive and complex to parse from the grammar file
- QUIC RFC did not have a consistent way to represent enums
- Variable-length integers are not represented in a parsable format

1.3.2 Risks

Potential risks in the future:

- Variable-length integers, arbitrary sizes, repeating fields, and optional fields are difficult to represent due to unknown factors. **Mitigation**: Need to parse data in a way that it knows the size of the unknown fields.
- QUIC RFC doesn't state which structures are PDUs explicitly, making it impossible to define PDUs. **Mitigation**: Extend the packet description syntax to clearly define PDUs.
- When building fields with references to other types, need to ensure that the type has been built before it's referenced. **Mitigation**: Build structures recursively when referenced from a field if not already built.

1.4 Plan

1.5 Semester 2

- Week 1: infer variable-length integer sizes from the enum type automatically.
- Week 2-3: represent arbitrary, repeating, and unknown fields with the correct sizes
- Week 4: ensure implementation is generalised for documents that are structured similar to QUIC and generalise any QUIC-specific parsing.
- Week 5-6: write tests for parsed representation and automatically generate code.
- Week 7: implement custom error messages and roughly document the parsed representation.
- Week 8+: focus on dissertation write-up

1.6 Ethics and data

This project does not involve human subjects or data. No approval required.