STRUCTURE-AWARE FUZZING WITH LIBFUZZER

Introduction to LIBFUZZER

- LibFuzzer is part of the LLVM compiler infrastructure project
- It is an in-process, coverage-guided, evolutionary fuzzing engine
- LibFuzzer's primary purpose is to help developers find bugs in software by automatically generating a vast amount of test inputs.

Key Concepts

- In-process fuzzing
- Coverage guided fuzzing
- Integration with sanitizers
- Seed corpus fuzzing
- Custom mutation

Benefits

- LibFuzzer runs the target program within the same process
- It can handle structured input formats more effectively with the use of custom mutators, making it suitable for structure-aware fuzzing
- It offers built-in functionality to minimize the corpus size, which can be helpful for managing storage and improving fuzzing efficiency
- It is often considered easier to use and integrate into existing projects due to lack of external dependencies

How to modify

- Define a .ksy file that describes the structure of your seed inputs using Kaitai Struct's syntax
- Write a C++ program that includes the Kaitai Struct library.
- Within the program,use kaitai struct’s parsing functionality to read the seed input data based on the .ksy definition
- During parsing, Kaitai Struct will automatically extract and interpret the data according to the defined structure, providing access to individual fields and objects
- Once we have access to the extracted data elements via Kaitai Struct, implement custom mutation logic within C++ program.