EDB: Debugger for Ethereum's Programming Languages

Report #4: System Design
Advised by Dr. Jackowitz
University of Scranton

Andrew Plaza

October 26, 2018

Abstract

1. Hello

Contents

0.1 Introduction 2
0.2 Levels 2
0.3 User Interface 2
0.4 Help System 2
0.5 2

0.1 Introduction

The EDB client application may be launched via the shell with the 'edb' command, which launches the main program routine for the binary. Depending upon the options the user has passed EDB, the main program will first connect to the locally-run ethereum test node the user has setup, which EDB references throughout the rest of it's execution. The solidity source code file that will be debugged must also be provided by the user. From these inputs EDB creates two of it's highest-level abstractions, the Compiler Model and the Emulator model, which make up the functionality of the debugger. The compiler model handles objects created by the compilation of the target source-language, such as the Abstract Syntax Tree, Source Mappings, and Bytecode. The Emulator model handles execution control and stores the Ethereum Virtual Machine(EVM) state at different steps in execution. State that is stored includes the current EVM Stack, Memory, and Non-Volatile Storage. Internally, the Emulator model uses the 'sputnikvm' library.

The design of EDB started first with a general structure/graph development plan where models that were needed were identified. Based upon this plan, generic interfaces were created based on the needs of the structure. Once this was complete, development was constrained to one part or sub-part of one model/interface, which worked towards fitting into the generic interface that was created during the first step. Unit tests were created alongside the original development of the part or sub-part, and testing took place to ensure the part worked individually before moving on to connect the part with the rest of the program. In order to keep track of work that is under development, finished, or needs to be started the Kanban board on 'Github Projects' was used.

0.2 Levels

0.3 User Interface

0.4 Help System

0.5