NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

School of Computer Science and Engineering
College of Engineering

Smart Contract Analysis and Verification

solv: a property-checking tool for the Solidity language

Student: Qiu Haoze

Supervisor: Asst Prof Lin Shang-wei

Introduction

As smart contracts gain more and more applications across many industries, security and correctness of them has become more important than ever. A property-checking tool for smart contracts written in the Solidity language, *solv*, is developed in this project. Unlike other analysis tools, *solv* incorporates user's input on what properties the smart contract is expected to satisfy in its analysis.

Flow of *solv*

1. **Properties declared in source code as special comments** (Fig 1)
2. ***solv* invoked in command-line** (Fig 2)
3. **Source code parsed and analyzed to obtain AST**
4. **Property checking tasks created** (Fig 3)
5. **AST checked against properties declared**
6. **Property checking results reported in the console** (Fig 2)

```
pragma solidity >=0.4.22 <0.6.0;
contract Wallet {
    address public owner; //@verifier fixed-after-init
    uint amount;

    ...
};
```

Property declaration with special comments

Fig 1. Property declaration as special comments

```
└─▪ cmake-build-debug/solv/solv Bank.sol

======= Bank.sol =======
Function "changeOwner", which contains an assignment to the
 fixed-after-init variable "owner", can be invoked through
a public function "somePublicFunc".
```

Fig 2. Example usage and output of *solv*

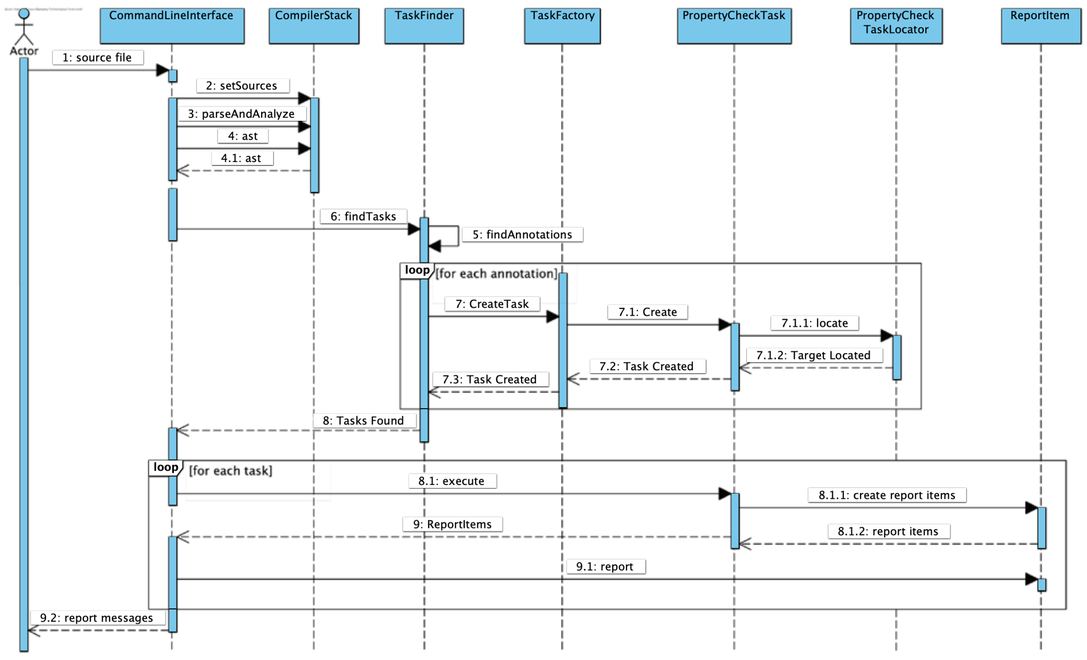

Fig 3. Sequence diagram of *solv*

Results

The framework for property checking, as well as the command-line interface is fully developed in *solv*. There is currently one property, *fixed-after-init*, implemented in *solv*. Another property, *reentrant-safe*, is still under development. More techniques in property checking are to be incorporated.