ackee blockchain

Woke Fuzzer

EthPrague 2022

Dominik Teiml, Michal Převrátil

Contents

Intro

- Ackee Blockchain is a blockchain security firm based in Prague, Czech Republic.
- Dominik Teiml: Ethereum Tech Lead @ Ackee Blockchain.
    - Fml. Gnosis, CertiK, Trail of Bits
- Michal Převrátil: Woke Developer @ Ackee Blockchain.

- Github repo: https://bit.ly/ethprague
    - > `/ctf`
- Telegram group: https://bit.ly/woke-fuzzer

Beginning

I think we all feel the unmaturity of existing Ethereum tooling.

This is usually NOT the fault of the developers of those tools.

Making great tools requires:

- a lot of experience with Ethereum,
- experience with the implementation language,
- a lot of expended effort,

and the gains (at least for the developer) are not immediately super high.

Sometime in October 2021, Dom saw enough possible improvements for Slither to warrant building a new tool.

Ideation

So originally, the idea was just to build a better Slither.

Later, we realized that if we have that, we can easily implement a lot of other features:

- a state-of-the-art language server,
- research new methods of contract visualization, both on the static level (source code) and dynamic (transaction execution)
- a line-by-line debugger for Solidity contracts,

and many others. We got to work.

Woke Fuzzer

First, we implemented a config parser, a Solidity version manager, a regex parser for versions and imports, and a compilation manager.

Sometime in March, Dom implemented a simple fuzzer during a smart contract audit.

We then used it during several audits and it found several high-severity bugs.

The Fork

In May 2022 (a few weeks ago) Dom decided to fork the project.

- to explore new methods of symbolic execution and contract visualization

The original Woke team at [Ackee Blockchain]() will focus on the Language server. 😍

The Merge?

Existing Fuzzers

Echidna

Echidna is powerful, but we think the following are very real issues:

- Prior to Solidity 0.8, it was not able to find assertion failures outside of main contract (issue#601)
- It cannot find assertion failures in non-top-level message calls (issue#668)
    - Not sure about the status right now

But most importantly, we don't like to use Solidity for testing. Compared to Python:

1. you have a slower iteration speed due to Solidity's absurd type system
    - in most PLs, subclasses are consistent with superclasses when passed in as parameters
        - in Solidity, `mapping balances (address => uint)`

    - `balances[someContractInstance]` will *not* be a valid expression

2. there is no REPL (iteractive console) for Solidity
3. it's harder to do differential fuzzing because you have to re-implement the logic in the same language

Brownie (Hypothesis)

- For a while, we liked `brownie’s State machines
    - It uses the Python library `hypothesis` under the hood
- However, at some point we realized it is not optimized functionally for smart contracts
- It was difficult to create complex fuzzing models

Woke Fuzzer

- Woke Fuzzer is Woke’s first released feature
- It currently uses Ganache for Node implementation and Brownie for Python bindings
- In the future, we plan to remove the dependency on Brownie, and optionally experiment with Anvil or Hardhat Network.

How to use

1. Create a `tests` directory
2. Create a directory for each fuzzing model, e.g. each smart contract
   - `token/`
   - `amm/`
3. Create a file `X_test` that serves as the entrypoint to the fuzzing model
   - `token_test.py`
   - `amm_test.py`
4. There are various possible structures for each of the directories

```
project/
├── contracts/
│   ├── Token.sol
│   └── Amm.sol
└── tests/
    ├── token/
    │   ├── setup.py
    │   └── flows.py
    ├── amm/
    │   ├── setup.py
    │   └── flows.py
    ├── __init__.py
    ├── token_test.py
    └── amm_test.py
```

```
setup.py
flows.py
```

```
helpers.py
setup.py
models.py
validators.py
transactions.py
flows.py
invariants.py
issues.py
```

Setup

Resources

```
git clone git@github.com:Ackee-Blockchain/presentation-ethprague-2022.git
```

Installation

You can install Woke with pip (`pip install woke`).

Pythonistas probably know how to do this, if you don't have much experience, you can just:

```
cd ctf;
python3 -m venv venv
source ./venv/bin/activate
pip install woke
```

Contracts

- You can find the contracts in the `/ctf/contracts` directory
    - https://bit.ly/ethprague
- We have two targets:
    a. A token with ICO mechanism
    b. An Amm (automated market maker)
- In `/ctf/tests` you can find the harness for your tests including the setup
- Your goal is to add flows that identify the bugs in the contracts 🙂

FAQ

> How can I get help?

Join the Telegram group! https://bit.ly/woke-fuzzer

> I’m done!

Try fuzzing some of your own contracts.