ChainSoft: Collaborative Software Development using Smart Contracts

Michał Król **<m.krol@ucl.ac.uk>**, Sergi Reñé **<s.ren@ucl.ac.uk>**,
Onur Ascigil **<o.ascigil@ucl.ac.uk>**, Ioannis Psaras **<i.psaras@ucl.ac.uk>**

University College London

<online version with animations>

Outsourcing Software Development

- Already huge and growing market
- 99% being small and medium-sized firms with under 500 employees
- New technologies emerge every single day
- Smaller companies cannot keep a team of developers with expertise in wide range of domains

Outsourcing software development can be difficult

- Finding the right specialists
- Specifying the requirements
- Sticking to the schedule
- Trust issues (Fair Trade problem)

Chainsoft Overview

- Automate outsourcing software development
- Well defined requirements and rewards
- Test Driven Model
- Assures fair exchange
- Encourages open source software development

Background

Blockchain Technologies

- Smart Contracts
  - Allow to logic on top of a blockchain
  - Turing complete language (Solidity)
  - Submitted data is publicly visible
- Oracle
  - Smart Contracts have now knowledge about external world
  - Oracles provide a trusted data feed
  - Return data and a proof (data attestation)
  - Oraclize queries HTTPS servers and uses TLSNotary to generate proofs

Github and Travis CI

- Github
  - The most popular web-based hosting service for version control using Git.
  - Commonly used to host open-source software projects
- Travis CI
  - Continous Integration
  - Tests software in a user defined environment
  - Docker Based

Overview

Environment

Requester

Software Developer

Payment System

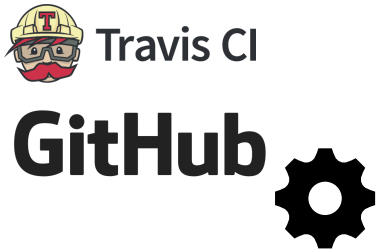
GitHub

Oracle

Assumptions

- Any developer can work on submitted tasks, but only the first valid solution will be rewarded
- The Requester and the Developer mutually distrust one another
- Both the Requester and the Developer trust the blockchain and the oracle
- Users can increase reward for a task even if they do not own it

Chainsoft



Code Verification

In order to verify a solution we need to:

- Compare if the included tests are the same as the ones submitted by the requestor.
- Recreate the specified environment and compile the whole project.
- Run the tests against the submitted solution.

...but running multiple tasks within a smart contract is expensive.

Code Verification

- We create a checksum file containing a list of all the test and their checksums.
- The checksum of the checksum file is stored in the environmental file
- When a solution is submitted, the Smart Contracts verifies only one single environmental file
- Travis CI uses the checksum file, to verify correctnes of all the tests

Code Verification

- Created the must prevent creating dummy software satisfying the tests only
- Different level of tests (unit/integration/...)
- Random or fetched input
- Tests should not rely on external data sources unless they are trusted by both the developer and the requester

Results

Results

Event	Ether Slow	Ether Standard	Ether Fast
deploy	0.057$	6.324$	16.673$
submitTask	0.013$	1.267$	3.675$
addReward	0.002$	0.241$	0.698$
submitSolution	0.007$	0.662$	1.919$
Total per task	0.022$	2.17$	6.292

Conclusion

- New platform for secure outsourcing software development
- Allows any user to submit tasks or solutions and enforces users' proper behaviour using Smart Contracts and trusted oracles
- Secure against rational adversary
- Future Work
  - Outsourcing without making the code public
  - Oracles using Trusted Execution Environments

Thank you