Making Blockchains Smarter: Towards Automated Robust Smart Contracts in Hyperledger

Advisors: Aniket Kate Ninghui Li *Student:* Sze Yiu Chau

1 Background

The world has seen a large amount of investments and developments going into the blockchain technology in recent years. In essence, a blockchain can be viewed as a distributed append-only log, the integrity of which is backed by the consensus of a network of nodes. Many properties of blockchain are considered to be desirable for the operation and collaboration of a variety of different industries. For example, the decentralized consensus and the append-only design of blockchain makes it difficult to tamper with historical transaction records and allows for great transparency.

Another important aspect of the blockchain technology is that it enables the creation of virtual entities that can automatically process and validate transactions on behalf of humans, which is a fundamental building block for a large-scale autonomous eCommerce. The business logic of such virtual entities are coded in the form of *smart contracts* that are agreed upon by the participating parties of some potential transactions. Given that smart contracts play a central role in transaction handling, we put the focus of this proposed research on their robustness.

2 Motivation and Problem Statement

In the face of high-speed and high-volume transactions, humans are increasingly the bottleneck of the economy: manual validation of contract conditions is time-consuming, error-prone, and expensive. This has led to the adoption of smart contracts. In a typical blockchain setup, a smart contract is a computer program that captures certain legal agreements, and can be executed automatically by validating nodes of the network. With adequate backing from cryptography, programmatic invocation of smart contracts can be seen as virtual entities legally acting on behalf of real-world entities in executing legal agreements. This is the main reason why smart contracts are considered to be one of the main enablers of autonomous eCommerce in the economy of the future.

We note that crafting smart contracts is itself also a human process, which makes it susceptible to errors. Recent incident with the DAO vulnerability[1] demonstrated that subtle bugs in smart contracts can potentially 1) lead to unexpected outcomes in the absence of adversaries; 2) be exploited by malicious parties and lead to severe monetary losses. The problem that we seek to solve in this proposed research is exactly on the robustness of smart contracts, specifically, on how to guarantee the robustness of smart contracts in an automated manner.

3 Vision

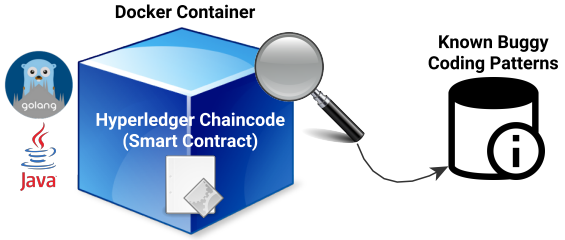
Our vision for the use of smart contracts with blockchain is simple: smart contracts should be made robust in the sense that 1) they cannot be abused by malicious parties; 2) they should faithfully capture the necessitated business logic. To this end, our ultimate goal is to be able to generate smart contracts automatically, along with proofs of robustness of the generated code, by leveraging formal methods. With the ability to automatically generate robust contract code given a formal legal agreement, the blockchain ecosystem can cater for a higher degree of automation, and seamlessly handle transactions at an even larger scale.

Research Team: The investigators have extensive expertise on crypto-currencies and blockchain technologies, as well as program analysis and formal methods for carrying out the proposed research.

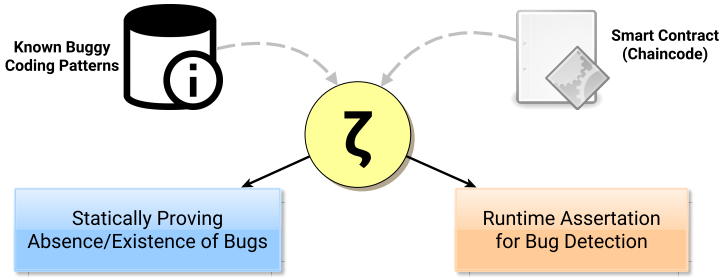
4 Approach and Research Plan

In order to realize the aforementioned vision, we divide the required research and development into multiple phases, which are to be executed in a sequential manner. In this proposal we focus on the first 2 phases. The artifacts produced in an earlier phase will be used in subsequent phases, gradually building towards the ultimate goal.

[1] `https://blog.ethereum.org/2016/06/17/critical-update-re-dao-vulnerability/`



(a) Phase 1: Investigate the language features and execution environment to identify potential buggy coding patterns.



(b) Phase 2: Design and implement a verifier that can statically/runtime assert the existence (or absence) of the known buggy patterns.

Figure 1: The first two phases of the proposed research, with a focus on robustness of smart contracts in Hyperledger.

Phase 1 - Pattern Identification: In this initial phase we would investigate the various features offered by the programming languages that are used to code smart contracts in Hyperledger[2], as well as the execution model and environment, with a focus on identifying potential error patterns. The Hyperledger uses Docker containers as a sandboxing mechanism for delivering and executing smart contracts (referred to in Hyperledger terminology as chaincode). Unlike in other blockchain technologies, for example, Ethereum, where smart contracts are first coded in some high-level programming languages and then compiled into bytecodes of its own virtual machine, chaincode in Hyperledger are executed according to the programming language of choice, without a Hyperledger-specific virtual machine. Chaincodes that have logical bugs might deviate from the expected behaviour and are susceptible to exploitations. We plan to draw intuitions on known buggy coding patterns (e.g. race conditions, missing exception handling, etc.) from previous studies on other blockchain platforms. The different choice of programming languages, network setup and execution environment in Hyperledger, however, might present new challenges.
Deliverable: A collection of erroneous coding patterns that can happen in Hyperledger chaincodes.

Phase 2: Towards High Robustness In this second phase we take as input the artifact produced from Phase 1, namely the collection of identified erroneous coding patterns, and develop a verifier that can vet chaincodes for potential logical bugs. The verification can happen statically in proving that a given chaincode does not exhibit any known buggy patterns. Additionally, the verifier can also execute the chaincode in an instrumented and sand-boxed manner to conduct runtime assertion, which is generally slower than static analysis but offers better accuracy, in order to gain further confidence on the robustness of a chaincode.
Deliverable: A verifier that can verify a given piece of chaincode contains known buggy coding patterns or not.

5 Impact

The blockchain community has seen a lot of developments lately, with now more than 700 cryptocurrencies available for trade online, many of them built on different premises and a consensus algorithm suitable for each of their intended applications. The lack of an open standard led to a fragmentation of the community, making it difficult to share data across different business domains and limits the possibilities for collaboration and innovation.

The emergence of Hyperledger fills the void in the community by providing a cross-industry open standard for distributed ledger, without relying on a central authority. It can be thought of as a clean slate redesign of a blockchain framework, with new insights based on the lessons learnt (e.g. permission and access control, choice of consensus algorithms and its impact on scalability, etc.) from various existing blockchain projects. Hyperledger has been well-received by members of different industries, citing the openness and collaborative principle embraced by the Linux Foundation who hosts the Hyperledger community, as well as the support and involvement of various vendors, developers and corporations, all make Hyperledger a serious contender for becoming the future standard of blockchain platform. Our proposed research takes advantage of the openness of the platform and will help lay a solid foundation in terms of both robustness and automaticity for coding business logic in smart contracts, which paves the way for a large-scale machine-to-machine eCommerce that drives the economy of the future.

[2] https://www.hyperledger.org/