Compliance Checking For Decentralized Applications

(Lightning Talk Abstract)

Flavio Corradini[1], Alessio Galassi[1], Alessandro Marcelletti[1], and Barbara Re[1]

University of Camerino, Camerino, Italy
`{name.surname}@unicam.it`

The blockchain adoption has expanded rapidly in various sectors, including governance, finance, and gaming, driven by its ability to improve transparency and ensure data integrity [1]. This is made possible especially by Decentralized Applications (DApps) that use smart contracts to encode business logic and enable the automated and trustless execution of predefined activities [2]. However, the implementation of smart contracts can diverge from the intended logic due to logical flaws or unforeseen scenarios in their execution [1]. Logical flaws may arise from incorrect assumptions, unhandled conditions, or unintended interactions between different contract functions [3]. Unforeseen scenarios stem from external factors such as network congestion, gas fee fluctuations, or user behaviors that were not taken into account during development. These issues may lead to unexpected behaviors, especially in smart contracts with intricate dependencies or dynamic conditions in their business logic.

In such a context, analysis techniques must be considered to spot unexpected behavior in order to check that smart contracts faithfully execute the desired business logic [4]. Several techniques have been proposed to monitor and verify smart contract implementations and executions [5]. Among the other approaches based on symbolic execution, formal verification, fuzzing, and static analysis has been considered. However, these approaches only focus on the verification of smart contract code implementation without considering actual executions and their produced data. They are unable to identify unexpected behaviors that may arise even within proper code. Such behaviors can instead be identified by observing the data produced with the execution of smart contracts, such as transactions, blocks, and events. Moreover, the immutability of the data stored on the blockchain provides a permanent record of information [6].

In this direction, compliance checking is a technique that allows defining compliance requirements and consequent compliance rules to check over the data under analysis [7]. It empowers auditors by supporting and streamlining auditing operations. Compliance checking is a promising solution that allows for the assessment of the adherence of rules to data resulting from the execution of a smart contract [8]. Indeed, smart contract execution data reflects the business logic of DApps, as process execution data captures the operational behavior of business processes in traditional business process management systems [9]. However, while compliance checking has potential applications in blockchain scenarios, its adoption poses a main limitation since traditional rules specification language [10, 11] do not explicitly refer to the blockchain data characterization. This limitation arises from the domain-specific concepts and logic of DApps, which require dedicated solutions for their analysis. We propose **a Domain-Specific Language (DSL), named CoBlock (*Compliance Checking language for Blockchain*)**, for defining compliance rules in terms of blockchain characterizations, treated as first-class citizens. The main advantage of the proposed DSL is that it allows the auditors to focus on blockchain characterizations relevant for DApp auditing, addressing the expressiveness limitations of existing compliance rule languages. CoBlock language is defined by a context-free grammar, thereby inheriting the associated properties of decidability, including the existence of algorithmic parsing techniques and the resolvability of membership problems [12]. Indeed, context-free languages are recognized by pushdown automata and can be parsed in polynomial time using algorithms such as Earley's algorithm [13]. CoBlock enables the creation of rules allowing the expression of blockchain concepts and characterizations relevant for DApps auditing.

This enables **a tailored framework that leverages CoBlock for compliance checking on blockchain**. Using a DSL for compliance checking on smart contract execution data provides auditors with a structured way to define compliance rules. Compared to manually analyzing blockchain logs or using general-purpose programming languages, CoBlock offers a more targeted approach for expressing compliance requirements. However, designing and implementing a DSL is a non-trivial task; to facilitate these operations, we present a framework for the DSL that provides a formal grammar, a parser, and a rule-checker engine to ensure correctness and usability.

We demonstrate our proposal using the CoBlock language on two real-world DApps, namely Augur and PancakeSwap. Augur is a blockchain-based betting platform that we use to demonstrate how its logic can be expressed and evaluated with CoBlock. PancakeSwap is a decentralized exchange platform that we use to show the language at work on a more comprehensive set of data. We chose the mentioned DApps as their logic was analyzed in the literature, and their logs were extracted and made available [14]. Those blockchain logs contain meaningful information. Overall, the evaluation results demonstrate that the CoBlock framework successfully captures compliance requirements by leveraging blockchain-specific characterizations. The rules formulated using the CoBlock language and evaluated through the tool show the entire framework's ability to detect compliant and non-compliant executions by expressing compliance requirements in real-world DApp execution data. The compliance rules not only spot unexpected behaviors, but their results provide relevant insights on the operational status of DApps, which can trigger further analysis or business logic enhancements.

References

1. P. Zheng, Z. Jiang, J. Wu, Z. Zheng, Blockchain-Based Decentralized Application: A Survey, IEEE Open J. Comput. Soc. 4 (2023) 121–133.
2. W. Viriyasitavat, L. D. Xu, D. Niyato, Z. Bi, D. Hoonsopon, Applications of Blockchain in Business Processes: A Comprehensive Review, IEEE Access 10 (2022) 118900–118925.
3. J. Chen, X. Xia, D. Lo, J. Grundy, X. Luo, T. Chen, Defining Smart Contract Defects on Ethereum, IIEEE Trans. Software Eng. 48 (1) (2022) 327–345.
4. F. Corradini, A. Marcelletti, A. Morichetta, B. Re, L. Ruschioni, A Digital Twin Approach for Blockchain Smart Contracts, in: SANER-C, IEEE, Rovaniemi, Finland, 2024, pp. 1–11.
5. G. Wu, H. Wang, X. Lai, M. Wang, D. He, S. Chan, A comprehensive survey of smart contract security: State of the art and research directions, Journal of Network and Computer Applications 226 (2024) 103882.
6. Z. Zheng, S. Xie, Dai, An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends, in: 2017 IEEE International BigData Congress, IEEE, Honolulu, HI, USA, 2017, pp. 557–564.
7. W. M. P. Van Der Aalst, Process Mining: A 360 Degree Overview, in: W. M. P. Van Der Aalst, J. Carmona (Eds.), Process Mining Handbook, Vol. 448, Springer, Cham, 2022, pp. 3–34, series Title: Lecture Notes in Business Information Processing.
8. C. Di Ciccio, G. Meroni, P. Plebani, Business Process Monitoring on Blockchains: Potentials and Challenges, in: S. Nurcan, I. Reinhartz-Berger, P. Soffer, J. Zdravkovic (Eds.), Enterprise, Business-Process and Information Systems Modeling, Vol. 387, Springer, Cham, 2020, pp. 36–51, series Title: Lecture Notes in Business Information Processing.
9. M. Weske, Business Process Management: Concepts, Languages, Architectures, Springer, Berlin, Heidelberg, 2019.
10. D. Knuplesch, M. Reichert, L. T. Ly, A. Kumar, S. Rinderle-Ma, Visual Modeling of Business Process Compliance Rules with the Support of Multiple Perspectives, in: D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, W. Ng, V. C. Storey, J. C. Trujillo (Eds.), Conceptual Modeling, Vol. 8217, Springer, Berlin, Heidelberg, 2013, pp. 106–120, series Title: Lecture Notes in Computer Science.
11. M. Pesic, H. Schonenberg, W. M. Van Der Aalst, DECLARE: Full Support for Loosely-Structured Processes, in: EDOC, IEEE, Annapolis, MD, USA, 2007, pp. 287–287.
12. M. Sipser, Introduction to the theory of computation, Sigact News 27 (1) (1996) 27–29.
13. J. Earley, An efficient context-free parsing algorithm, Commun. ACM 13 (2) (1970) 94–102.
14. F. Corradini, A. Marcelletti, A. Morichetta, B. Re, A Data Extraction Methodology for Ethereum Smart Contracts, in: PerCom Workshops, IEEE, Biarritz, France, 2024, pp. 524–529.