Resilient and Trustworthy Transactive Platform for Smart and Connected Communities

Aron Laszka, Anastasia Mavridou, Abhishek Dubey

Internet of Things, Cyber-Physical Systems, and Data Sciences are fueling the development of innovative solutions for various applications in Smart and Connected Communities (SCC). These solutions are often data-driven, which makes them vulnerable to data integrity attacks. Additionally, the increasing dependence on dynamic data-driven support systems implies that any operational breach in the underlying SCC networks or services due to attacks or failures will have cascading effects. Lastly, such attacks will not only cripple the SCC operations but also escalate customers' privacy concerns, influencing the extent to which they are willing to share data.

These challenges have led to increasing focus on SCC platforms that provide participants the capability to not only exchange data and services in a decentralized and perhaps anonymous manner, but also provide them with the capability to preserve an immutable and auditable record of all transactions in the system. Such transactive platforms are actively being suggested for use in Healthcare, Smart Energy Systems, and Smart Transportation Systems. These platforms can provide support for privacy-preserving and anonymizing techniques, such as differential privacy, fully homomorphic encryption, and mixing [2,3]. Further, the immutable nature of records and event chronology in these platforms provides high rigor and auditability. Lastly, the decentralized nature of these platforms ensures that any adversary needs to compromise a large number of node to take control of the system.

Blockchains form a key component of these platforms because they enable participants to reach a consensus on any state variable in the system, without relying on a trusted third party or trusting each other. Distributed consensus not only solves the trust issue, but also provides fault-tolerance since consensus is always reached on the correct state as long as the number of faulty nodes is below a threshold. Further, blockchains also enable performing computation in a distributed and trustworthy manner in the form of smart contracts. However, while the distributed integrity of a blockchain ledger presents unique opportunities, it also introduces new assurance challenges that must be addressed before protocols and implementations can live up to their potential. For instance, smart contracts deployed in practice are riddled with bugs and security vulnerabilities. A recent automated analysis of 19,336 smart contracts deployed in practice found that 8,333 of them suffered from at least one security issue. Although this study was based on smart contracts deployed on the public Ethereum blockchain, the analyzed security issues were largely platform agnostic. Security vulnerabilities in smart contracts present a serious issue for two main reasons. Firstly, smart-contract bugs cannot be patched. By design, once a contract is deployed, its functionality cannot be altered even by its creator. Secondly, once a faulty or malicious transaction is recorded, it cannot be removed from the blockchain (“code is law” principle). The only way to roll back a transaction is by performing a hard fork of the blockchain, which requires consensus among the stakeholders and undermines the trustworthiness of the platform.

In practice, these vulnerabilities often arise due to the semantic gap between the assumptions that contract writers make about the underlying execution semantics and the actual semantics of smart contracts. To tackle these security risks and vulnerabilities, we introduce a formal, finite-state machine (FSM) based language for designing smart contacts, and we provide a tool [4] for generating code from the high-level, graphical, and FSM-based language to low-level smart contract code (Solidity [1]). The advantages of our approach, which helps developers to create secure contracts rather than to fix existing ones, are threefold. First, we provide a formal model with clear semantics and an easy-to-use graphical editor, thereby decreasing the semantic gap and eliminating the issues arising from it. Second, rooting the whole process in rigorous semantics allows the integration of formal analysis tools, which can be used to verify safety and security properties, thereby enabling the development of correct-by-design smart contracts. Third, we model and analyze the interactions between the decentralized smart contract and external IoT actors. Specifically, we provide and enforce rigorous interaction semantics that enables us to provide end-to-end assurance guarantees on the whole system.

[1] Mavridou, A. and Laszka, A. (2018). Designing Secure Ethereum Smart Contracts: A Finite State Machine Based Approach. *22nd International Conference on Financial Cryptography and Data Security* (FC 2018).

[2] Laszka, A., Dubey, A., Walker, M., and Schmidt, D. (2017). Providing Privacy, Safety, and Security in IoT-Based Transactive Energy Systems using Distributed Ledgers. *7th International Conference on the Internet of Things* (IoT 2017).

[3] Bergquist, J., Laszka, A., Monika S., and Dubey, A. (2017). On the Design of Communication and Transaction Anonymity in Blockchain-Based Transactive Microgrids. *arXiv preprint arXiv: 1709.09601*.

[4] FSoliM tool. Accessible online at https://cps-vo.org/group/SmartContracts