Pruvendo

Our expertise:

- Conducting an **informal audit** of the contract. This allows to realize the program architecture and dependencies, construct the call graph of contracts methods, find the potential security flows and prepare ourselves for the specification creation step.
- Summarising results of informal audit we develop the **high-level specification** - free but technical description of what system does which high level properties it should have and describe the main user scenarios of exploitation
- Development of **low-level specification** - which is bound with the implementation and follows it
- **Translation of solidity code into Ursus** (Coq embedded language) - preparing the contract to formal verification
- **QuickChick** - running sophisticated randomised tests. That is taking the propositions formulated on the previous steps inside Coq ecosystem, we use special methods to run property-based checker, which apply brute forcing algorithms to find the counter examples.
- **Deductive formal verification**

Our cases:

GOSH - the first git on blockchain. We created formally verified software deployment system to guarantee security of software supply chain.

Everscale blockchain. We performed formal verification of most critically important smart contracts. Such, the Multisig contract whose verification allowed to find five critical bugs missed by many audits.

Flex DEX - developed specification and performed formal verification for the first DEX with On-chain Order Book.

H2Q - GameFi fully created using formally verified smart contracts

CEO - Sergey Egorov

CTO - Andrey Lyashin

We strongly recommend to use formal verification at least at the feature freeze stage, to simplify detection of architectural problems and vulnerabilities and also speed up code validation, shortening the development cycle.

This is crucial for smart contracts, exploit in which can lead to money and reputational losses especially with nontrivial logic.

✉ team@pruvendo.com

www. pruvendo.com