IMPROVING THE SECURITY OF SMART CONTRACTS THROUGH THE INTEGRATION OF MACHINE LEARNING: CHALLENGES AND PROSPECTS

Oleksandr Tereshchenko
PhD student of the Software Engineering Department
Odesa Polytechnic National University, Ukraine

In today's digital world, smart contracts have become a fundamental element of blockchain ecosystems, offering automation, immutability and decentralization of transactions. However, with the growth of their popularity and application, the security issue of smart contracts comes to the fore, as vulnerabilities in their code can lead to significant financial losses and undermine trust in blockchain technologies. Traditional code analysis methods, while effective in certain contexts, have limitations that can make it difficult to detect complex vulnerabilities. In this context, machine learning opens up new horizons of possibilities, offering tools for deeper analysis and understanding of smart contract code, which can significantly improve the detection and correction of potential vulnerabilities.

The purpose of the work is to study the potential of machine learning methods in identifying and eliminating vulnerabilities of smart contracts to increase their security in blockchain ecosystems.

Representatives of traditional methods are formal verification, symbolic execution, fuzzing, and intermediate representation. The method of formal verification consists in mathematically proving the correctness of algorithms. This approach allows for the guaranteeing of the security of smart contracts by identifying and eliminating certain classes of vulnerabilities. Although this method has high reliability of the detected results and the ability to prove the absence of error classes, it also has its drawbacks. It requires significant effort to formalize specifications and may not be suitable for complex systems, and is not always practical for a wide range of smart contracts.

Another method, symbolic execution, analyzes the program with symbolic rather than concrete input to identify potential errors. This approach is capable of detecting complex vulnerabilities, but may suffer from the "state explosion" problem. Another method, fuzzing, involves generating random input data to a program to detect non-standard behavior. This method is effective in detecting vulnerabilities, but has a high number of false positives [1].

An intermediate representation used to analyze and optimize smart contract code simplifies the analysis of complex code. Despite its usefulness, it does not detect vulnerabilities by itself without additional analysis tools [2].

On the other hand, machine learning offers a new approach to code analysis, allowing us to quickly adapt to new types of vulnerabilities and handle large volumes of code. However, large amounts of annotated data are required for effective machine learning training, and interpretation of the results can be difficult. Existing approaches that use models such as LSTM (Long Short-Term Memory) and GNN (Graph Neural Networks) face certain limitations. In particular, these methods may not take into account key variables in the code of smart contracts, which leads to insufficient semantic modeling and not always satisfactory detection results [3].

One of the main problems is the "black box" of neural networks, which makes it difficult to interpret the results. In most cases, deep learning cannot pinpoint the exact location or line of code where a vulnerability might be, unlike traditional detection tools.

The development of deep learning technologies and their application to the analysis of smart contracts continues to open up new opportunities for improving the security of blockchain applications. At the same time, it is important to take into account the existing limitations and work to overcome them, in particular, by improving interpretability and integration with time-tested analysis methods [4].

Improving interpretability: One of the areas of development is improving the interpretability of deep learning models. This may include developing methods that can pinpoint specific areas of code that are potentially vulnerable to attack, thereby making the results more convincing and useful to developers.

Integration with expert rules: Another important aspect is the integration of deep learning with expert rules defined in traditional detection tools. This can help improve the accuracy of vulnerability detection, as expert rules are already well adapted to identify specific types of vulnerabilities.

Hence, the main challenges associated with the application of machine learning include limited interpretability and the need for large volumes of annotated data for effective learning. However, the prospects for the development of deep learning technologies open up new opportunities for improving the security of smart contracts, in particular through integration with expert rules and the development of methods that improve the interpretability of results. Thus, the integration of machine learning into the development and audit processes of smart contracts opens up new horizons for ensuring their security, while requiring further research and development in this area.

References:

1. Making smart contracts smarter / Luu L. et al. // Proceedings of the 2016 ACM SIGSAC conference on computer and communications security / ACM. Vienna, 2016. P. 254.
2. Tereshchenko O. , Komleva N. Vulnerability Detection of Smart Contracts Based on Bidirectional GRU and Attention Mechanism // ICTERI 2023. Communications in Computer and Information Science. Cham, 2023. Vol. 1980. P. 276-287.
3. Komleva N. , Tereshchenko O. Requirements for the development of smart contracts and an overview of smart contract vulnerabilities at the Solidity code level on the Ethereum platform // Herald of Advanced Information Technology. 2023., No. 1 Vol. 6. P. 54-68.
4. Learning to repair software vulnerabilities with generative adversarial networks / Harer J. et al. // Advances in Neural Information Processing Systems / NIPS. San Francisco, 2018. P. 7933.