Vulnerability Detection in Source Code via Subgraph Isomorphism Analysis

Douglas Takada, Freeman Trader, Tashi Stirewalt, and Assefaw Gebremedhin

ABSTRACT

Vulnerability detection is crucial for ensuring the security, reliability, and privacy of software systems. By representing source code as a property graph, where nodes represent code elements and edges represent relationships between them, we are able to perform a comprehensive vulnerability assessment. With the source code represented as a graph, we can leverage graph representation and machine-learning techniques to identify potential vulnerabilities. In this study, we introduce two things: a novel graph representation of source code based on the Abstract Syntax Tree (AST) and a novel approach to estimating subgraph isomorphism. We demonstrate the effectiveness of using graph analysis of source code for vulnerability detection. It indicates that such methods can be particularly valuable for detecting well-understood and documented vulnerabilities.

RELATED WORK

The research paper titled "Modeling and Discovering Vulnerabilities with Code Property Graphs" by Fabian Yamaguchi et al. [1] presents a novel method for representing source code known as a code property graph. This graph consolidates traditional program analysis concepts into a single data structure. While effective, this approach may face scalability issues when dealing with large code bases. In the paper "Abstracting Program Dependencies Using the Method Dependence Graph" by Haipeng Cai et al. [2], a new dependence abstraction called the method dependence graph (MDG) is introduced. It approximates the traditional fine-grained software dependence model at the method level. "Source Code Vulnerability Detection: Combining Code Language Models and Code Property Graphs" by Ruitong Liu et al. [3] proposes a unified model combining pre-trained code language models with code property graphs for vulnerability detection. However, this method might underestimate code semantics and encounter challenges capturing long-distance contextual information. The work on "Source Code Vulnerability Detection Using Deep Learning" [4] focuses on static security vulnerability detection in C# code using deep learning algorithms. Lastly, "An Analytical Review of the Source Code Models for Exploit Analysis" [5] investigates a new approach for detecting vulnerabilities that can be exploited in cyber attacks.

REFERENCES


ACKNOWLEDGEMENTS

This work is supported by funding for the VICTEROY Northwest Institute for Cybersecurity Education and Research (CySER) provided by The Office of the Undersecretary of Defense for Research and Engineering, in collaboration with the Air Force Research Laboratory and Griffiss Institute.