SUMMARY

As modern software systems increase in complexity and size, the challenges of finding and fixing software defects, especially vulnerabilities, have increased. Traditional approaches, like static code analysis tools and manual code reviews, are usually not effective for large-scale and complex codebases. These approaches normally yield a high false-positive rate and cannot detect runtime vulnerabilities, which means critical defects remain undetected and system security is compromised.

It addresses the challenge through the presentation of VulGraphNet, a new defect prediction model using a combined approach of static and dynamic analysis with Graph Neural Networks. This model will rely on a multi-level graph representation: Control Flow Graphs, Data Flow Graphs, and Abstract Syntax Trees to capture interdependent relations within the code in complex ways. VulGraphNet provides a more holistic solution to finding defects resulting from complex interactions among various software components by integrating static analysis, which identifies structural vulnerabilities in the code, and dynamic analysis, which monitors runtime behavior.

Among the main contributions, this work develops machine learning techniques in integrating Graph Neural Networks for improving feature extraction. GNNs naturally fit in analyzing graph-structured data, thus empowering the model to learn such complex patterns and associations

across different parts of a program. This deep feature learning allows the identification of potential security vulnerabilities that may go unnoticed using conventional approaches, especially those stemming from intricate dependencies and context-sensitive behaviors. Besides, VulGraphNet also employs multi-scale feature fusion techniques, enhancing the generalization ability of the model and making it applicable to more software projects, from small-scaled systems to large, distributed applications.

In this paper, some publicly available vulnerability datasets are used to conduct several experiments on the proposed VulGraphNet model. The results have shown outstanding performance with both detection accuracy and coverage. This is because combining static and dynamic analysis reduces false positives and false negatives, which increases the precision and recall of the detection of defects. This leads to the most effective and reliable method of vulnerability prediction, which is considered a very important process in software development regarding security breaches and maintenance cost reduction.

The results of the present study bring into relief the relevance of integrating different techniques of analysis, which is necessary to overcome the limitations identified within current methodologies for vulnerability detection. The integration of dynamic analysis with machine learning-based static analysis enables a more robust framework in the detection of

complex vulnerabilities depending on both the structure and execution behavior of the code. In the end, this research provides a contribution to the advancement of software security by providing an accurate, efficient, and scalable tool for defect detection at an early stage.

In the end, VulGraphNet represents a significant enhancement to software vulnerability detection by harnessing the strengths of static and dynamic analysis, Graph Neural Networks, and multi-scale feature fusion for more accurate and scalable vulnerability detection systems. This work will contribute not only to enhancing the efficiency of software defect identification but also pave the way for further research into automated software security tools.

Keyword: Software Vulnerability Detection, Graph Neural Networks (GNN), Static Analysis, Graph Attention Networks (GAT).