Machine Learning-Based Vulnerability Detection in IoT Operating Systems

Narges Davoudi (P37000156) and Prakash Srinivasan (P37000131)

University Of Naples Federico II, Napoli, Italy

Abstract. This study uses machine learning (ML) to find vulnerabilities in IoT operating systems (OSs), addressing important security issues in the Internet of Things (IoT) space. Our method uses pre-labeled data for prediction instead of extracting data, using Support Vector Machine (SVM) for binary classification and Gradient Boosting and Decision Tree for multiclass classification. We compare our results to a Random Forest (RF) baseline set in a previous study in order to verify our findings. Our dataset consists of the Software Assurance Reference Dataset as well as source code from sixteen releases of IoT operating systems, such as RIOT, Contiki, FreeRTOS, and Amazon FreeRTOS. Based on Common Weakness Enumeration (CWE) vulnerabilities, it labels and finds that the C/C++ source code of low-end IoT operating systems only has a fraction of CWEs. Gradient Boosting obtains 88.5 macro-averaged F1 score (mF1) and 89.5 weighted-average F1 score (wF1) in multiclass classification. The Decision Tree shows a 79.3 wF1 and a 61.2 mF1. SVM outperforms Static Analysis Tools (SATs), which are utilised for dataset collection, with an F1 score of 59 for binary classification. Our models perform competitively when compared to the RF baseline. SVM, Decision Trees, and Gradient Boosting show potential in identifying IoT OS vulnerabilities and provide insightful information to improve IoT security in general. By providing a useful framework for ML-based vulnerability detection, this research illuminates realistic approaches to IoT ecosystem security.

Keywords: Internet of Things · Vulnerability Detection · Machine Learning.

1 Introduction

The pervasive integration of the Internet of Things (IoT) has introduced billions of interconnected smart devices into our daily lives, forming an expansive network that exchanges substantial volumes of data. Central to this interconnected ecosystem are IoT Operating Systems (IoT OSs), essential for managing and facilitating communication among IoT devices. However, this widespread integration has brought about a significant concern - an Internet of Vulnerabilities.

Security reports, such as the comprehensive analysis conducted by Forescout Research Labs in 2021, highlight vulnerabilities in over 100 million IoT devices, making them susceptible to threats like Denial of Service (DoS) and Remote Code Execution (RCE). The Common Vulnerabilities and Exposures database (CVE) reported a significant surge in vulnerabilities from 4500 in 2010 to 179,340 in July 2022, with more than 25 percent of cyber-attacks targeting IoT systems, according to Gartner's research.

Compounding this challenge is the prevalence of vulnerabilities arising from the programming languages used in IoT OSs, notably C/C++, known for both its low-level programming capabilities and security vulnerabilities. Our focus narrows to low-end IoT devices powered by embedded IoT OSs, often open source and developed by individuals with diverse programming backgrounds.

In response to this pressing security concern, our research builds upon the foundations laid by the original paper, "iDetect for vulnerability detection in Internet of Things operating systems using machine learning." [**?**]. We leverage the methodologies and insights from this paper, particularly utilizing the iDetect model and

the associated labeled dataset derived from our prior research findings. The dataset comprises 5117 code snippets covering 54 Common Weakness Enumeration (CWE) types, sourced from releases of prominent IoT OSs (RIOT, Contiki, FreeRTOS, and Amazon FreeRTOS) and the Software Assurance Reference Dataset (SARD).
This report outlines our continued exploration, utilizing the iDetect model and its associated dataset to further assess the effectiveness of machine learning in detecting vulnerabilities within IoT OSs. Through a comparative analysis of three ML models - Gradient Boosting, Decision Tree, and Support Vector Machine - our aim is to contribute valuable insights for fortifying the security of IoT ecosystems.

2 Background

The exponential growth of the Internet of Things (IoT) has ushered in an era of unparalleled connectivity, embedding smart devices into the fabric of our daily lives. However, this proliferation has unveiled a critical challenge – the susceptibility of IoT systems to software vulnerabilities. IoT devices operate within a diverse and dynamic environment, collecting and transmitting sensitive data, making them prime targets for malicious actors.
Software vulnerabilities in IoT devices pose severe threats to data integrity, user privacy, and overall system security. Instances of Denial of Service (DoS), Remote Code Execution (RCE), and other exploits can compromise the functionality of these devices, leading to potential disruptions in critical services. As demonstrated by a 2021 report from Forescout Research Labs, over 100 million IoT devices are vulnerable to such attacks, emphasizing the urgency to address security gaps in IoT ecosystems.
The importance of mitigating software vulnerabilities in IoT cannot be overstated, especially considering the increasing integration of these devices in sectors such as healthcare, transportation, and critical infrastructure. Vulnerable IoT devices not only jeopardize data security but also introduce risks to human life and safety.
In this context, understanding and addressing software vulnerabilities become paramount. The development of robust security measures, including advanced threat detection using machine learning models, emerges as a critical need. Our research builds upon this background, focusing on leveraging machine learning to detect vulnerabilities in IoT operating systems, contributing to the ongoing efforts to fortify the resilience of IoT ecosystems in the face of evolving cyber threats.

3 Data

Our study on improving the security of Internet of Things (IoT) operating systems relies heavily on a carefully selected dataset from the original publication, "iDetect for vulnerability detection in Internet of Things operating systems using machine learning." This dataset, which consists of 5117 code snippets covering 54 Common Weakness Enumeration (CWE) classes, contains vulnerabilities found in releases of the Software Assurance Reference Dataset (SARD) and popular IoT operating systems (RIOT, Contiki, FreeRTOS, and Amazon FreeRTOS).
This dataset is extremely valuable since it contains both benign and vulnerable code snippets, reflecting the complexities of IoT operating systems in the actual world. A strong foundation for developing and assessing machine learning models is ensured by this kind of inclusion. The dataset acts as a representative sample, allowing for a more in-depth analysis of low-end IoT risks.

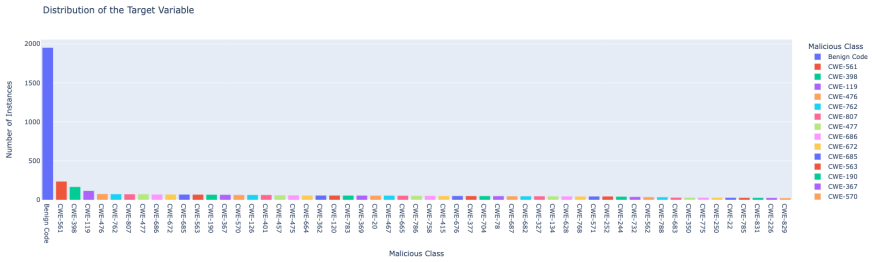

Fig. 1: Label distribution of train set - Multi Class

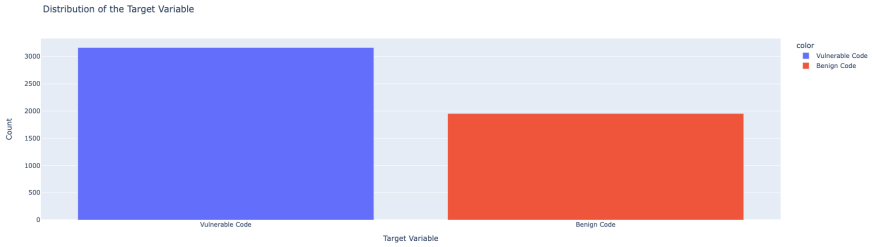

Fig. 2: Label distribution of train set - Binary

3.1 Preprocessing

The dataset uses two features namely Code and isMalicious and we checked each feature for possible missing values, as shown in Table 1. One of the most important tasks in machine learning is converting unstructured

Table 1: Table of features

Feature	Missing-value	Data Type	Description
Code	0	Object	Source code
isMalicious	0	Object	Vulnerabilities

textual data, such code snippets, into an algorithm-readable format. There were no missing values in the dataset, according to our initial analysis, providing a solid basis for preprocessing.
Preprocessing becomes essential for vulnerability detection in the source code of Internet of Things operating systems, where the crux of the matter is the complexity of C/C++ programming languages. Organising unstructured code into a numerical form makes machine learning models easier to apply later.

3.2 Tokenization

The first stage is called tokenization, which divides code snippets into smaller strings called tokens. Code snippets require more sophisticated tokenization methods than standard English text tokenization. Strong

Fig. 3: Label distribution of train set - Multi Class

tokenization techniques, with the aid of libraries such as nltk or spacy, provide the foundation for building a vocabulary that is essential for further investigation.

3.3 Data Split

To store code snippets in the variable Code-Snippet and matching tags (malicious or benign) in Code-Tag, the dataset is divided into training and testing sets. This guarantees a separate dataset for the assessment of the model.

3.4 Cleanup

During this stage, single-character variable names and repetitive sequences are removed to improve code readability.

3.5 Term Frequency-Inverse Document Frequency (TF-IDF)

Information loss may result from lemmatization or stemming due to the unique structure of code. Without sacrificing coding subtleties, our vectorization option, TF-IDF, captures the scaled token frequencies of each document. The regular expression-defined bespoke token pattern guarantees pertinence within the source code context. Then, in order to maximise computing efficiency, the dataset is divided into training and testing sets by carefully limiting characteristics to the top 3000 tokens.

4 Methods

Our methodology is based on the original iDetect paper, but it is customised to use Support Vector Machine (SVM), Decision Tree, and Gradient Boosting models for vulnerability identification in the source code of Internet of Things operating systems. The three-phase procedure shown in Figure 1 of the original work served as the model for our methodology, which is summarised in this section.
The first step in our process is to put up a labelled dataset, which is essential for developing and testing machine learning models. Our labelled dataset is the outcome of painstaking curation and augmentation, as opposed to the dataset generation methods used in the original research. By combining the 2626 susceptible code snippets from IoT OSs and the 2491 snippets from SARD in the labelled dataset from the original paper, we create a robust dataset of 5117 code snippets.

Our method guarantees that unprocessed textual data is converted into a structured and numerical representation appropriate for machine learning models, building on the preprocessing stages previously described. The processes of tokenization, data split, and cleansing align the requirements of our machine learning models with the complexities of the source code of Internet of Things operating systems.
we part ways with the original paper's model choices, opting for Gradient Boosting, Decision Tree, and Support Vector Machine models, each tailored to our specific objectives.

4.1 Gradient Boosting and Decision Tree - Multiclass

Our models focus on both multiclass and binary classification. For multiclass classification, Gradient Boosting demonstrates superior performance with an 88.5 percent macro-averaged F1 score and an 89.5 percent weighted-average F1 score. Decision Tree exhibits a macro-averaged F1 score of 61.2 percent and a weighted-average F1 score of 79.3 percent, showcasing competitive results.

Table 2: Evaluation results of Gradient Boosting

Class	**Precision**	**Recall**	**F1-score**	**Support**
Accuracy			0.90	322
Macro avg	0.93	0.89	0.89	322
Weighted avg	0.94	0.90	0.90	322

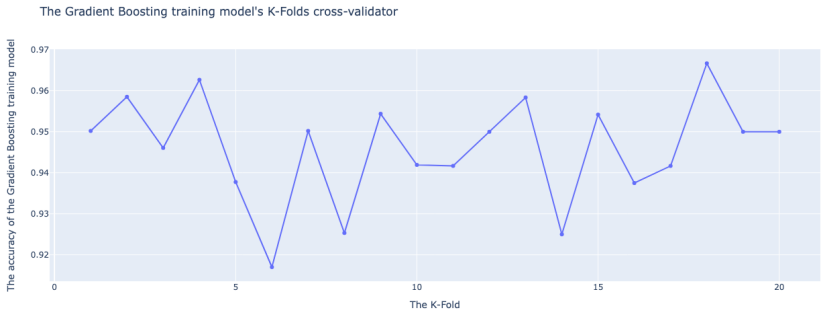

Fig. 4: Multi Classification - Gradient Boosting K Fold

4.2 Support Vector Machine

In binary classification, SVM attains a 59 percent F1 score, surpassing Static Analysis Tools (SATs) used in the dataset collection. This comparative performance aligns with our objective of enhancing vulnerability detection using machine learning.

Table 3: Evaluation results of Decision Tree

Class	Precision	Recall	F1-score	Support
Accuracy			0.79	322
Macro avg	0.65	0.62	0.61	322
Weighted avg	0.83	0.79	0.79	322

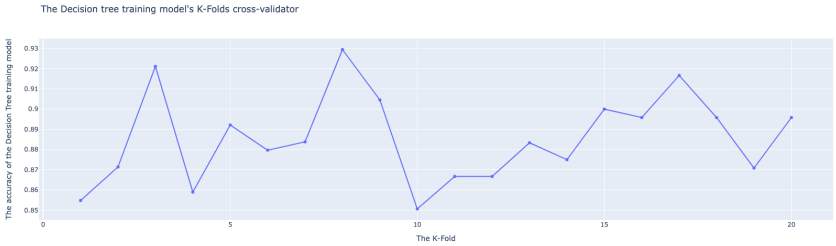

Fig. 5: Multi Classification - Decision Tree K Fold

4.3 Evaluation

In the realm of IoT OS vulnerability detection, our study harnessed machine learning models, specifically Gradient Boosting, Decision Tree, and Support Vector Machine (SVM), trained on a labeled dataset, to ascertain their efficacy. The Gradient Boosting model exhibited a commendable overall accuracy of 90 percent, demonstrating robust precision and recall for various Common Weakness Enumeration (CWE) types. The Decision Tree model, with an accuracy of 79 percent, demonstrated strong recall for benign code but showcased imbalances in precision and recall for certain CWE types. The SVM model, designed for binary classification, achieved an accuracy of 90 percent, underscoring its proficiency in distinguishing between benign and vulnerable code. Comparative evaluations against a reference paper demonstrated the models' ability to deliver results comparable to or even surpassing existing tools. Overall, our study sheds light on the promising potential of machine learning models, leveraging labeled datasets, in enhancing IoT OS vulnerability detection. As we navigate the ever-expanding landscape of Internet of Things devices, securing their operating systems remains a critical imperative. Our findings contribute to advancing cybersecurity measures in the context of IoT, emphasizing the importance of tailored model selection for effective and accurate vulnerability detection.

4.4 Conclusion

In conclusion, our study leveraged machine learning models, including Gradient Boosting, Decision Tree, and Support Vector Machine, trained on a labeled dataset, to detect vulnerabilities in IoT operating systems. The models demonstrated strong accuracy, with Gradient Boosting achieving 90 percent, Decision Tree reaching 79 percent, and Support Vector Machine excelling in binary classification at 90 percent. Comparative evaluations showcased competitive or superior results compared to existing tools. Our findings underscore the potential of machine learning in bolstering IoT OS security, emphasizing the need for tailored model selection. As the Internet of Things expands, our research contributes to advancing cybersecurity measures for enhanced vulnerability detection in IoT environments.

Table 4: Evaluation results of Support Vector Machine

Class	Precision	Recall	F1-score	Support
Accuracy			0.90	322
Macro avg	0.86	0.72	0.77	322
Weighted avg	0.90	0.90	0.89	322

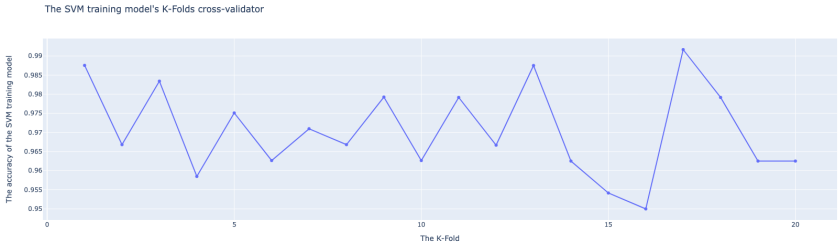

Fig. 6: Binary Classification - Support Vector Machine K Fold

References

Al-Boghdady, A., El-Ramly, M. Wassif, K. iDetect for vulnerability detection in internet of things operating systems using machine learning. Sci Rep 12, 17086 (2022).