**LITERATURE REVIEW**

**DDOS DETECTION IN KUBERNETES USING DEEP AND MACHINE LEARNING**

**Version: [1.5]**

|  |  |
| --- | --- |
| Project Code | F24-001 |
| Internal Supervisor | N/A |
| External Supervisor | Mr. Shoaib Raza |
| Project Manager | Asher Siddique |
| Project Team | Rija Anwar, Mahnoor Amjad |
| Submission Date | 11th December, 2024 |

**Document History**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Version*** | ***Name of Person*** | ***Date*** | ***Description of change*** |
| 1.0 | Mahnoor Amjad | 10th November, 2024 | Created the document |
| 1.1 | Mahnoor Amjad | 12th November, 2024 | Added the Abstract & Background |
| 1.2 | Mahnoor Amjad | 14th November, 2024 | Added the Literature Review |
| 1.3 | Mahnoor Amjad | 15th November, 2024 | Added the Problem Statement, Appendices, References and Bibliography |
| 1.4 | Mahnoor Amjad | 18th November, 2024 | Made final edits |
| 1.5 | Mahnoor Amjad | 10th December, 2024 | Added the final dates and edits |

**Distribution List**

|  |  |
| --- | --- |
| ***Name*** | ***Role*** |
| Mr. Shoaib Raza | External Supervisor |
| Asher Siddique | Project Manager |
| Mr. Abdul Aziz | Head of Department |
|  |  |
|  |  |

**Document Sign-Off**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Version*** | ***Sign-off Authority*** | ***Project Role*** | ***Sign-off Date*** |
| 1.5 | Sir Shoaib Raza | External Supervisor |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. Abstract........................................................................................................................................

2. Background and Justification.......................................................................................................

3. Problem Statement......................................................................................................................

4. Literature Review.........................................................................................................................

5. Appendices..................................................................................................................................

6. References...................................................................................................................................

7. Bibliography.................................................................................................................................

**1. ABSTRACT**

In the recent years, cloud technology has advanced across all disciplines and used worldwide. Micro-service architecture, infamous for deploying services as per the user demands has also spread notoriously. Cloud is known for deploying on-demand services and micro-services are known for being the most scalable as the user traffic/demand grows. This is why these both go hand-in-hand because as the traffic grows new services can be deployed on demand. As with all technology, the responsibility of making cloud a secure environment to protect the users’ interests and data is a primary concern.

The buildout of cloud native applications and micro-service architectures across various systems and tech in the industry has brought about an intensity in the need for a set of effective mechanisms to deal with DoS (Denial of Service) and DDoS (Distributed Denial of Service) attacks in cloud native environments.

Our review spans over two key research areas; classical machine learning for general anomaly detection in micro-services and containers, and deep learning approaches specifically targeting DoS/DDoS detection in Kubernetes environments.

**2. BACKGROUND AND JUSTIFICATION**

With the rising adoption of cloud native & micro-service architectures, Kubernetes has gained sufficient notability as a container orchestration platform that deploys micro-services as per user/customer demand. However, it is prone to DoS and DDoS attacks made vulnerable by its scalability and flexibility issues just like any other online system, but what makes it harder to detect is its distributed nature. This leads to severe service disruptions, loss of revenue, & compromised customer trust.

Classical machine learning has been employed for anomaly detection in cloud and containerized environments. Deep learning has been tried in detecting DoS/DDoS attack patterns. Challenges, however, remain in terms of real-time detection, scalability, minimizing false positives, and adapting to evolving attack patterns.

Our efforts aim to combine the logic of ML and DL to detect anomalous behavior that is indicative of DoS/DDoS attacks on cloud native applications through a feature collection technique tailored to Kubernetes.

**3. PROBLEM STATEMENT**

Our project revolves around detecting DoS and DDoS attacks that threaten the micro-service architecture in Kubernetes. Micro-services are modular, loosely coupled services often deployed across multiple systems at once, making the detection of said attacks more complex. We aim to design a system that will collect resource metrics from various Kubernetes nodes and deploy two Machine Learning models and two Deep Learning models to identify attack patterns using real-time monitoring.

So far, only one type of Deep Learning model (LSTM with auto-encoders) has been deployed. On the other hand, Classical Machine Learning models have been utilized for a more generalized concept of detecting anomalies based off on application and system logs. In this project, we aim to combine the domains of Machine Learning models and Deep Learning models and try to find an optimal solution for detecting attacks on cloud-native applications. The system will identify abnormal traffic patterns that signal the occurrence of these attacks using real-time data comprised of system calls, and resource metrics taken in real time from Kubernetes. To enhance accuracy and reliability, we intend to farm our own data through a simulation of DDoS attacks to generate independent and more precise sequences for comparison.

**4. LITERATURE REVIEW**

**4.1. MACHINE LEARNING (ML) MODELS FOR DOS AND DDOS DETECTION**

Classical Machine Learning has been explored and deployed to study patterns and detect anomalous trends or behaviors in Kubernetes environments.

**Anemogiannis [1] proposed a hierarchal system for anomaly detection by establishing baseline metrics for each Kubernetes resource based on activity over time. ML is used to study any deviations that occur in the multilayered system through the baseline metrics that act as thresholds. The abnormalities are scored according to severity with the top most crucial anomaly being flagged for immediate action and the lowest one being logged for trend analysis.**

**The hierarchal structure helps with the cascading issues, as a problem in one resource impacts others, allowing for a targeted and more efficient response to anomalies.**

Du et al. [2] describes a system that applies supervised learning to classify system behavior based on performance data. It takes labelled sample data to train models and deploys different algorithms (Support Vector Machines (SVM), Random Forests, Naive Bayes, and k-Nearest Neighbors (k-NN)) to classify the behaviors. It then injects faults like high CPU usage to derive real-time instances of anomalies to improve the accuracy of the model to detect varied typed of anomalies in live deployment. The system performs time series analysis across multiple containers running the same micro-service to identify the specific container causing the issues.

Zou et al.[3] proposed an enhanced version of Isolation Forest, where weights are assigned to each resource metric (e.g., CPU, memory) based on their relevance to anomaly detection. This weighted feature selection improves accuracy by adjusting the random selection process. The system automatically adjusts its monitoring interval, reducing overhead while maintaining real-time anomaly detection.

**4.2. DEEP LEARNING (DL) APPROACHES FOR DETECTION**

Deep Learning models have been deployed to capture data that helps identify Dos and DDoS attacks that a cloud native environment might suffer from.

Tien et al. [4] combines Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to detect anomalies in Kubernetes-managed Docker environments. CNNs capture spatial patterns in data, helpful for identifying unusual behaviors in resource metrics, while RNNs are adept at temporal patterns, tracking sequences of container activities over time. CNNs analyze structured data, like container resource usage, to spot unusual patterns. RNNs retain information from previous steps, allowing them to model temporal dependencies.

The model is trained on historical data to identify patterns in normal operations, and it flags deviations as potential anomalies. Additionally, it leverages an ensemble of deep learning methods to boost detection accuracy, reaching an effectiveness of around 96% for identifying abnormal behaviors.

Lin et al. [5] proposed a classified distributed learning model that classified applications based on their behavior patterns in system calls which helps tailor the anomaly detection to each application type and solving the issue of the diverse anomaly behaviors and limited training data. The CDL framework deploys an auto-encoder that reconstructs expected behavior patterns. The reconstruction error between the input and output layers is how attacks are detected; higher the error, the bigger the attack.

Forsberg [6] focuses on identifying problematic behavior and tracing the root cause of it in distributed micro-service environments. Identification is carried out through the use of Self-Organizing Maps (SOM) models, an unsupervised network technique for dimensionality reduction and pattern recognition in complex datasets. SOM models use performance metrics to identify any deviations and the anomalies are then classified based on these metrics. Root-cause analysis is done by a dependency graph of the micro-services, combined with inter-service causal analysis. An anomaly alignment algorithm further enhances detection accuracy by adjusting for timing discrepancies, which helps in matching the precise moments of anomaly occurrences between dependent services.

**4.3. COMBINATION OF ML & DL MODELS IN ATTACK DETECTION**

Some studies define approaches that make use of both ML and DL for anomaly or attack detection in a Kubernetes environment.

Harlicaj [7] provides us with two approaches combined in effort. The first one uses statistical models to detect baseline anomalies. These statistical techniques help identify deviations based on historical data patterns, useful for detecting web-based attacks. Classical ML classifiers (e.g., k-Nearest Neighbors or Decision Trees) are deployed for detection providing initial, low level identification with minimal overhead.

The second approach introduces a neural network model, specifically suited for detecting more complex, non-linear patterns in request traffic. This DL model, potentially a recurrent neural network (RNN) or convolutional neural network (CNN), enhances the ability to recognize attack signatures embedded within web requests. By training on labeled data, the DL model learns nuanced patterns, which helps in identifying multi-step attacks or subtle anomalies that might not be apparent with basic statistical methods.

**4.4. CURRENT GAPS**

Across the research papers, several common gaps emerge in anomaly and DDoS detection within Kubernetes and containerized environments. Some studies focus on deep learning models but encounter challenges related to computational demands, making them less suitable for real-time detection in resource-constrained settings. For instance, LSTM and auto-encoder-based approaches can be effective for capturing complex patterns but may strain processing resources in large-scale clusters.

On the other hand, certain papers rely heavily on traditional ML models, which can be efficient but may lack the sophistication needed to handle evolving, multi-step attacks or non-linear patterns in web traffic. While ML models like Isolation Forests and SVMs offer solid baseline performance, they may fall short when applied to complex, dynamic data in micro-service environments.

Additionally, while some approaches provide a multi-level detection framework to better suit Kubernetes’ layered structure, they often overlook temporal analysis and adaptive feature selection, which are essential for accurately detecting anomalies in rapidly changing systems. Finally, interpretability remains a limitation in several deep learning models, making it challenging for administrators to understand and trust automated decisions in security-critical applications.

**4.5. BASE PAPER AND FUTURE DIRECTIONS**

The paper, *“A Novel Approach to Feature Collection for Anomaly Detection in Kubernetes Environment”*, provides us with our basis for research and implementation. The paper presents a new method to collect features based upon which anomalies related to different attacks, resource contentions and/or performance bottlenecks are identified.

These features are resource metrics such as, memory, CPU, network I/O etc, being utilized across different Kubernetes resources. Darwesh et al. [8] introduces an event-driven data collection technique that collects and compiles data per every change in configuration, providing a more detailed real-time view of the system as compared to periodic polling and helps in identifying issues where resource allocation and configurations could shift rapidly due to auto-scaling and other dynamic actions.

The paper proposes that the system deploys multi-level analysis where anomalies are detected at every level i.e. container, node, pod. This hierarchal system proves effective for a Kubernetes architecture where anomaly detection can be localized as well as be carried out in a broader, systematic manner across the clusters.

It further proposes the integration of machine learning models, including Isolation Forests and Long Short-Term Memory (LSTM) networks, for processing the collected features. Isolation Forests are particularly useful for their efficiency in high-dimensional, sparse data typical in microservice telemetry, while LSTMs handle sequential dependencies and time series data to identify patterns indicative of anomalous behavior.

Darwesh et al. [8] prove their approach through the improvements in accuracy and a significant reduction in false positives, attributed to the optimized feature selection and the hierarchical anomaly detection strategy.

We will be building upon the real time monitoring that the paper explains which can be a significant step towards detecting anomalous behavior in any environment as well that of Kubernetes. The novelty in the feature collection proposed also helps greatly in identifying issues. By adapting the selected features to the environment’s current state, the system minimizes data collection overhead, making it efficient for cloud-native applications that are run on Kubernetes. The hierarchical anomaly detection structure aligns closely with the Kubernetes architecture, offering a unique perspective among conventional methods, which often do not account for Kubernetes’ layered setup.

**5. APPENDICES**

 **DoS (Denial of Service)**   
A type of cyber-attack where a single source floods a target system, such as a server or network, with excessive requests to overload and disrupt its normal functions.

 **DDoS (Distributed Denial of Service)**   
Similar to DoS, this attack involves multiple sources, often compromised devices, overwhelming a target system, making it inaccessible to legitimate users.

 **Micro-services**  
An architectural style for developing applications as a collection of small, loosely coupled, independently deployable services, each responsible for a specific business function.

 **Kubernetes**  
An open-source platform for automating the deployment, scaling, and management of containerized applications, widely used for orchestrating workloads in cloud environments.

 **Cloud Native Applications**  
Applications designed specifically for cloud environments, leveraging services like containers, micro-services, and DevOps practices to achieve scalability, resilience, and agility.

 **Auto-encoders**  
A type of artificial neural network used in unsupervised learning that learns efficient encodings by compressing input data into a lower-dimensional form and then reconstructing it back to the original format.

 **Machine Learning (ML)**  
A subset of artificial intelligence (AI) where algorithms learn patterns from data, allowing systems to make predictions or decisions without being explicitly programmed for each task.

 **Deep Learning (DL)**  
A specialized subset of ML that uses layered neural networks to model complex patterns in large datasets, particularly effective for tasks involving images, speech, and natural language.

 **Support Vector Machines (SVM)**   
A supervised ML algorithm used for classification and regression tasks, which finds the optimal hyperplane to separate data points of different classes in feature space.

 **Random Forests**  
An ensemble ML technique that uses multiple decision trees to make predictions, improving accuracy and robustness over single decision trees by averaging multiple predictions.

 **Naive Bayes**  
A probabilistic classifier based on Bayes' theorem, assuming independence between features. It's fast and effective for text classification and other tasks where feature independence can be assumed.

 **k-Nearest Neighbors (k-NN)**   
A simple ML algorithm used for classification and regression, where an object is classified based on the most common class among its k closest neighbors in the feature space.

 **Convolutional Neural Networks (CNNs)**   
A type of deep learning model particularly effective in image and video processing, utilizing convolutional layers to detect spatial hierarchies in data, like edges or shapes in images.

 **Recurrent Neural Networks (RNNs)**   
A type of neural network designed for sequential data, such as time series or language, where connections between nodes form cycles to retain memory of previous inputs.

 **Self-Organizing Maps (SOM)**   
An unsupervised learning technique that uses a type of artificial neural network to produce a low-dimensional, interpretable representation of high-dimensional data, often used for clustering and visualization.

**6. REFERENCES**

1. Anemogiannis, M. (2023). Hierarchical anomaly detection strategy for Kubernetes resources.
2. Du, M., et al. (2018). DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning.
3. Zou, H., et al. (2019). Docker container anomaly monitoring system based on optimized isolation forest algorithm.
4. Tien, C. W., Liao, J. W., Chang, S. C., & Kuo, S. Y. (n.d.). KubAnomaly: Anomaly detection for the Docker orchestration platform with neural network approaches.
5. Lin, Y., Tunde-Onadele, O., & Gu, X. (n.d.). CDL: Classified Distributed Learning for Detecting Security Attacks in Containerized App.
6. Forsberg, J. (2022). Automatic anomaly detection and root cause analysis for microservice clusters.
7. Harlicaj, E. (n.d.). Anomaly Detection of Web-Based Attacks in Microservices.
8. Darwesh, G., Hammoud, J., & Vorobeva, A. A. (n.d.). A novel approach to feature collection for anomaly detection in Kubernetes environment and agent for metrics collection from Kubernetes nodes.

**7. BIBLOGRAPHY**

1. Anemogiannis, M. (2023). Hierarchical anomaly detection strategy for Kubernetes resources.
2. Du, M., et al. (2018). DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning.
3. Zou, H., et al. (2019). Docker container anomaly monitoring system based on optimized isolation forest algorithm.
4. Tien, C. W., Liao, J. W., Chang, S. C., & Kuo, S. Y. (n.d.). KubAnomaly: Anomaly detection for the Docker orchestration platform with neural network approaches.
5. Lin, Y., Tunde-Onadele, O., & Gu, X. (n.d.). CDL: Classified Distributed Learning for Detecting Security Attacks in Containerized App.
6. Forsberg, J. (2022). Automatic anomaly detection and root cause analysis for microservice clusters.
7. Harlicaj, E. (n.d.). Anomaly Detection of Web-Based Attacks in Microservices.
8. Darwesh, G., Hammoud, J., & Vorobeva, A. A. (n.d.). A novel approach to feature collection for anomaly detection in Kubernetes environment and agent for metrics collection from Kubernetes nodes.