НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ноябрь–декабрь 2024 Том 24 № 6 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
November–December 2024 Vol. 24 No 6 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

doi: 10.17586/2226-1494-2024-24-6-1007-1015

Enhancing Kubernetes security with machine learning: a proactive approach to anomaly detection

Ghadeer Darwesh[1], Jaafar Hammoud[2], Alisa A. Vorobeva[3]✉

[1,2,3] ITMO University, Saint Petersburg, 197101, Russian Federation

[1] ghadeerdarwesh32@gmail.com, https://orcid.org/0000-0003-1116-9410
[2] hammoudgj@gmail.com, https://orcid.org/0000-0002-2033-0838
[3] vorobeva@itmo.ru✉, https://orcid.org/0000-0001-6691-6167

Abstract

Kubernetes has become a cornerstone of modern software development enabling scalable and efficient deployment of microservices. However, this scalability comes with significant security challenges, particularly in detecting specific attack types within dynamic and ephemeral environments. This study presents a focused application of Machine Learning (ML) techniques to enhance security in Kubernetes by detecting Denial of Service (DoS) attacks and differentiating between DoS attacks, resource overload caused by attacks, and natural resource overloads. We developed a custom monitoring agent that collects telemetry data from various sources, including real-world workloads, actual attack scenarios, simulated hacking attempts, and induced overloading on containers and pods, ensuring comprehensive coverage. The dataset comprising these diverse sources was meticulously labeled and preprocessed, including normalization and temporal analysis. We employed and evaluated various ML classifiers, with Random Forest and AdaBoost emerging as the top performers, achieving F1 macro scores of 0.9990 ± 0.0006 and 0.9990 ± 0.0003, respectively. The novelty of our approach lies in its ability to accurately distinguish between different types of resource overloads and provide robust detection of DoS attacks within Kubernetes environments. These models demonstrated a high degree of accuracy in detecting security incidents, significantly reducing false positives and false negatives. Our findings highlight the potential of ML models to provide a targeted, proactive security framework for Kubernetes, offering robust protection against specific attack vectors while maintaining system reliability.

Keywords

Kubernetes security, microservices, machine learning, anomaly detection, containerization, cybersecurity, telemetry data, real-time threat detection

For citation: Darwesh G., Hammoud J., Vorobeva A.A. Enhancing Kubernetes security with machine learning: a proactive approach to anomaly detection. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2024, vol. 24, no. 6, pp. 1007–1015. doi: 10.17586/2226-1494-2024-24-6-1007-1015

УДК 004.056

Повышение безопасности Kubernetes с использованием машинного обучения: проактивный подход к обнаружению аномалий

Гадир Дарвиш[1], Жаафар Хаммуд[2], Алиса Андреевна Воробьева[3]✉

[1,2,3] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация

[1] ghadeerdarwesh32@gmail.com, https://orcid.org/0000-0003-1116-9410
[2] hammoudgj@gmail.com, https://orcid.org/0000-0002-2033-0838
[3] vorobeva@itmo.ru✉, https://orcid.org/0000-0001-6691-6167

Аннотация

Введение. Kubernetes — ключевая платформа для масштабируемого и эффективного развертывания микросервисов. С увеличением масштабируемости возрастает сложность выявления и своевременного обнаружения специфических типов атак в динамичных средах Kubernetes. **Метод.** В работе предложен подход для повышения безопасности Kubernetes, позволяющий детектировать атаки типа «отказ в обслуживании» (Denial

of Service, DoS), основанный на использовании методов машинного обучения. Подход базируется на данных, полученных от пользовательского агента мониторинга, осуществляющего сбор телеметрической информации из различных источников, включая реальные рабочие нагрузки, сценарии атак, имитацию взлома и перегрузку ресурсов в контейнерах и подах. Полученные данные размечаются и обрабатываются, включая нормализацию и временной анализ для создания полноценного набора данных. **Основные результаты.** В ходе экспериментов протестированы различные классификаторы машинного обучения. Наиболее высокие показатели качества получены с использованием алгоритмов Random Forest и AdaBoost, дающие макро F1-оценки 0,9990 ± 0,0006 и 0,9990 ± 0,0003 соответственно. Разработанный подход позволяет эффективно отличать перегрузки ресурсов, вызванные атаками от естественных перегрузок, и обеспечивает точное выявление DoS-атак. Предложенная модель машинного обучения демонстрирует высокую точность в обнаружении инцидентов безопасности, существенно снижая количество ложных срабатываний. **Обсуждение.** Полученные результаты показывают, что модели машинного обучения могут стать основой для создания проактивной системы безопасности Kubernetes, которая обеспечит надежную защиту от специфических векторов атак, сохраняя при этом стабильность системы. Полученные результаты могут быть полезны исследователям и специалистам в области кибербезопасности приложения Kubernetes.

Ключевые слова

безопасность Kubernetes, микросервисы, машинное обучение, обнаружение аномалий, контейнеризация, кибербезопасность, телеметрические данные, обнаружение угроз в реальном времени

Ссылка для цитирования: Дарвиш Г., Хаммуд Ж., Воробьева А.А. Повышение безопасности Kubernetes с использованием машинного обучения: проактивный подход к обнаружению аномалий // Научно-технический вестник информационных технологий, механики и оптики. 2024. Т. 24, № 6. С. 1007–1015 (на англ. яз.). doi: 10.17586/2226-1494-2024-24-6-1007-1015

Introduction

The microservices architecture has emerged as a transformative approach in software development, enabling the decomposition of monolithic applications into smaller, independently deployable services. This architectural style promotes scalability, flexibility, and rapid deployment, addressing the demands of modern software applications [1]. By leveraging containerization technologies such as Docker and Kubernetes, microservices can be orchestrated efficiently, providing a robust framework for developing and maintaining complex, large-scale applications [2]. However, the distributed and dynamic nature of microservices introduces significant security challenges. Traditional security mechanisms often fail to adequately protect microservices due to their inability to adapt to the continuous integration and deployment cycles inherent in these environments.

Security in Kubernetes extends beyond traditional perimeter defenses. In a containerized environment, each pod represents a potential entry point for attackers. Vulnerabilities in container images, misconfigurations in Kubernetes manifests, or even compromised nodes can lead to catastrophic breaches if left unchecked [3, 4]. In this context, Machine Learning (ML) models can be trained to recognize patterns and anomalies in data, making them highly effective at detecting attacks that may otherwise go unnoticed by traditional Intrusion Detection Systems (IDS). The ability of ML models to learn and adapt makes them particularly suited for the dynamic and complex nature of cybersecurity. Both supervised and unsupervised ML algorithms are used in this domain. Supervised learning can classify whether network traffic is normal or potentially harmful, while unsupervised learning can identify previously unseen attack patterns [5–7].

However, the application of ML in cybersecurity is not without challenges. One of the main challenges is the quality and quantity of data required to train effective ML models. Cybersecurity datasets need to be large and diverse to encompass the wide range of potential attacks, and they also need to be labeled accurately to train supervised learning algorithms. This often requires significant resources and expertise [8]. Another challenge is the interpretability of ML models. Many effective ML models, such as neural networks, are often described as “black boxes” because their internal workings are not easily interpretable by humans. This can make it difficult to understand why a particular prediction was made, which is often important in cybersecurity contexts [9, 10].

In this article, we explore the concept of using machine learning to enhance security in Kubernetes environments by detecting DoS attacks. Our approach not only identifies DoS attacks but also differentiates between resource overloads caused by attacks and natural fluctuations in resource usage, a critical distinction for reducing false positives. We collect telemetry data from multiple sources, including real-world workloads, actual attack scenarios, simulated hacking attempts, and induced overloads on containers and pods. By leveraging machine learning algorithms to analyze this diverse data, we aim to detect anomalous behaviors indicative of potential attacks. This proactive approach empowers organizations to identify and mitigate attacks in Kubernetes environments before they escalate into full-blown breaches.

Related Works

Recent advancements in container technology have spurred significant research into securing these environments. This section reviews notable works that have contributed to enhancing the security of containerized applications, particularly in detecting attacks such as DoS and differentiating between attack-induced resource overloads and natural system overloads.

Researchers in [11] introduced a real-time Host-based IDS for Linux containers. Their approach monitors system calls from the host kernel to detect anomalies in container

behavior. The system achieved a high detection rate of 100 % with a low false positive rate of around 2 %.

In [12], researchers developed an online anomaly detection system for Docker containers using an optimized isolation forest algorithm. By assigning weights to resource metrics and using weighted feature selection, their system improves detection accuracy. This approach effectively detects anomalies in both simulated and real cloud environments with minimal performance overhead, which is crucial for distinguishing between attack-related overloads and natural variations in resource usage.

Researchers in [13] proposed a probabilistic real-time IDS for Docker containers. Their IDS uses n-grams of system calls and probabilistic models like Maximum Likelihood Estimator and Simple Good Turing to detect malicious applications. The system achieved accuracy ranging from 87 % to 97 % on various datasets, demonstrating its potential in detecting a wide range of attacks, including DoS.

Researchers in [14] evaluated the performance of anomaly-based IDS at the container level for multi-tenant applications. They used the Bag of System Calls technique and a sliding window with eight machine learning algorithms. Decision Tree and Random Forest algorithms provided the best results, achieving an F-Measure of 99.8 %. The study also found that Decision Tree is faster and consumes less Central Processing Unit (CPU) and memory compared to Random Forest, which is advantageous for real-time attack detection in resource-constrained environments.

In [15], researchers presented an approach to evaluate intrusion detection effectiveness in container-based systems using attack injection. They used a TPC-C workload with a database engine running as a container and monitored its system calls. The approach was effective in different scenarios, consistently detecting most attacks with precision values showing more variance, which could be beneficial in scenarios involving varied types of attacks like DoS.

A study in [16] focused on container vulnerability exploit detection, evaluating static and dynamic detection schemes using 28 real-world vulnerability exploits. Static scanning detected only 3 out of 28 vulnerabilities, while dynamic anomaly detection schemes detected 22 exploits. This highlights the importance of dynamic detection methods in identifying attack-induced anomalies that static methods might miss.

Researchers in [17] introduced Compiler Description Language (CDL), a classified distributed learning framework for detecting security attacks in containerized applications. CDL integrates online application classification with anomaly detection to address the challenge of insufficient training data for dynamic, short-lived containers. CDL improved detection rates and reduced false positive rates significantly compared to traditional methods, making it a promising approach for distinguishing between different causes of resource overloads in Kubernetes environments.

In [18], researchers analyzed security attacks and detection techniques for Docker containers, highlighting the need for effective security measures due to the high efficiency and widespread use of Docker in development and deployment. Their study proposed a detailed analysis of existing security mechanisms and attacks, presenting a detection framework that proved effective in experimental evaluations, particularly in identifying DoS attacks.

Lastly, in [19], researchers conducted a comprehensive analysis of Docker container attack and defense mechanisms. They identified significant gaps in existing defenses, particularly in handling dynamic attack landscapes. Their evaluation framework, using an extensive dataset of 51 real-world vulnerabilities, demonstrated that static scanning tools and dynamic anomaly detection approaches both have limitations, with high false positive rates and inadequate training data being major issues. This underscores the need for more robust detection mechanisms that can accurately differentiate between attack-induced and natural resource overloads.

Problem Statement: Challenges in Detecting Attacks in Kubernetes and Microservices Environments

Adopting Kubernetes for its agility and scalability also introduces significant security challenges. This section examines the key challenges in detecting attacks in these environments and why traditional security measures may be insufficient.

1. Complexity and Dynamism: Kubernetes environments are highly dynamic, with containers continuously being created, scaled, and terminated in response to varying workloads. In a microservices architecture, where applications consist of numerous independently deployable services, each running in its own container, the complexity is further amplified [12, 19]. This dynamism increases the difficulty in distinguishing between normal operational behaviors and potential attacks, such as DoS attacks or resource overloads caused by malicious activities. Differentiating these from natural, benign overloads in the system is a critical challenge for effective attack detection.
2. Attack Surface Expansion: The proliferation of containers and microservices significantly expands the attack surface in Kubernetes environments. Attackers have numerous entry points to exploit, ranging from vulnerabilities in container images and misconfigurations in Kubernetes manifests to compromised nodes and insecure Application Programming Interfaces. Traditional security tools often focus on perimeter defenses and lack the visibility needed into containerized workloads, leaving Kubernetes environments vulnerable to sophisticated attacks, including DoS attacks that can be easily mistaken for legitimate resource spikes [18, 19].
3. Ephemeral Nature of Containers: Containers are ephemeral by design, meaning they are short-lived and can be easily replaced or terminated. Traditional security measures, which rely on static configurations and long-lived assets, struggle to adapt to the transient nature of containers. Consequently, security teams may find it difficult to maintain visibility into the security posture of Kubernetes workloads and respond effectively to attacks, particularly those involving intentional resource exhaustion that mimics natural system behavior [19].

4. Lack of Contextual Awareness: In Kubernetes environments, security events and telemetry data are generated at a high volume and velocity. Without proper context distinguishing between normal operational behavior and attack-induced anomalies becomes a daunting task. For example, an unexpected surge in resource usage might either be a result of a legitimate operational demand or a DoS attack. Traditional security measures often lack the intelligence to analyze telemetry data in real-time and identify subtle deviations from normal behavior that indicate potential attacks [9, 10]. This lack of contextual awareness increases the likelihood of false positives and false negatives, undermining the effectiveness of security measures in Kubernetes environments.
5. Scalability and Performance Impact: As Kubernetes clusters scale to accommodate growing workloads, traditional security measures may introduce scalability and performance overhead. Agent-based security solutions commonly used in traditional environments may struggle to keep up with the dynamic nature of Kubernetes deployments leading to increased resource consumption and performance degradation [6]. This challenge is particularly pronounced when trying to detect DoS attacks and differentiate them from legitimate high-load scenarios where the security system must operate efficiently without hindering overall system performance.

These challenges underscore the need for advanced, adaptive security solutions that can effectively detect and differentiate between various types of attacks, including DoS attacks, in Kubernetes and microservices environments. By leveraging machine learning techniques, this study aims to address these challenges, providing a robust framework for proactive attack detection and mitigation in these complex, dynamic systems.

Dataset Collection with Custom Monitoring Agent

In this section, we detail the dataset used for training and evaluating our machine learning models for attack detection in Kubernetes environments. The dataset is comprehensive, incorporating telemetry data collected from a variety of sources to ensure the models can effectively differentiate between DoS attacks, resource overloads caused by attacks, and natural system overloads.

Custom Monitoring Agent. To gather comprehensive telemetry data from Kubernetes nodes and applications, we developed a custom monitoring agent tailored to our specific requirements. This agent is designed to collect a diverse range of system-level and application-level metrics, providing valuable insights into resource utilization, network activity, and application behavior. The custom monitoring agent is deployed across all nodes in the Kubernetes cluster, ensuring thorough data collection and monitoring coverage.

Data Collection Sources. The dataset comprises telemetry data collected from multiple sources, including:

— **Real-world Workloads.** Metrics were gathered from production Kubernetes clusters running under typical operational conditions. This real-world data provides a baseline for normal system behavior and natural resource usage patterns.
— **Actual Attack Scenarios.** We intentionally induced DoS attacks and other resource-exhausting activities in controlled environments. This data helps in training the models to recognize the specific signatures and patterns associated with such attacks.
— **Simulated Hacking Attempts.** To further enrich the dataset, we simulated various attack vectors that could potentially target Kubernetes environments. These simulations were designed to mimic real-world hacking activities, including attempts to overload resources or exploit vulnerabilities.
— **Induced Overloading on Containers and Pods.** We also conducted experiments to deliberately overload containers and pods in a non-malicious manner. This data is crucial for teaching the models to distinguish between malicious overloads caused by attacks and benign overloads resulting from legitimate high-demand scenarios.

Data Collection from Kubernetes Nodes. Our monitoring agent collects an extensive array of system-level metrics from each Kubernetes node at regular intervals. These metrics include CPU usage, disk I/O, network traffic, memory utilization, process activity, and TCP/UDP socket statistics. By capturing these metrics, we obtain a holistic view of the cluster health and performance, enabling the identification of anomalous behavior that may indicate potential attacks.

Data Collection from Applications. In addition to monitoring Kubernetes nodes, our custom agent gathers metrics directly from the target applications. These application-level metrics encompass CPU and memory usage, open file descriptors, and detailed HyperText Transfer Protocol (HTTP) request statistics. Monitoring application behavior in real-time allows us to detect deviations from normal operation such as unusual spikes in resource consumption. Specifically, we identify and analyze patterns in HTTP request handling by categorizing them into predefined threat models, which helps in distinguishing between benign anomalies and potential attacks. This approach enhances the precision of our detection mechanisms, ensuring that the identified anomalies are meaningful and actionable.

Timestamped Data Collection. All metrics collected by our monitoring agent are timestamped to facilitate temporal analysis of system behavior. Timestamped data enables the correlation of events across different components of the Kubernetes environment, aiding in the identification of patterns indicative of security incidents, including those that may evolve gradually or occur in bursts such as DoS attacks.

Dataset Integrity and Quality. Ensuring the integrity and quality of the collected dataset is paramount for the effectiveness of our machine learning models. We employ rigorous data validation techniques to address missing values, outliers, and data quality issues, ensuring that our models are trained on clean and reliable data. High-quality datasets enhance the accuracy and robustness of the machine learning models leading to more reliable detection of attacks and reducing the chances of false positives and negatives.

Experiments and Model Evaluation

The success of machine learning models in detecting attacks within Kubernetes environments relies heavily on the quality of the dataset, the preprocessing steps, and the evaluation methods used. In this section, we outline the experiments conducted to train and evaluate our models, focusing on their ability to detect DoS attacks, differentiate between attack-induced resource overloads, and natural resource spikes.

Data Preprocessing. The initial phase of our machine learning project involves comprehensive data processing. This crucial step includes loading the dataset, cleansing unnecessary columns, transforming time-related data into discrete components, and partitioning the dataset into training and testing subsets. Utilizing the pandas library, we leveraged its robust DataFrame structure to efficiently manipulate our structured data.

Firstly, we removed the 'id' column, which does not contribute to our analysis. Subsequently, the 'time' column was converted to a datetime format, from which 'hour', 'minute', and 'second' components were extracted using pandas date-time functionality. This step ensures that temporal data can be effectively utilized in model training, which is crucial for detecting patterns related to DoS attacks that may occur at specific times or under certain conditions.

After data preprocessing, the dataset was split into features (X) and the target variable (y). We applied normalization to the features using the StandardScaler from scikit-learn, ensuring that each feature contributes equally to the model training. The dataset was then divided into training and testing sets to facilitate model evaluation on unseen data, enhancing the model generalizability [20].

Model Selection and Training. We employed a diverse array of classifiers from the scikit-learn library, including Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Support Vector Classifier, K-Nearest Neighbors Classifier, Gradient Boosting Classifier, AdaBoost Classifier, and Extra Trees Classifier. Each classifier was trained using default parameters on the training set, learning the underlying patterns in the data to make accurate predictions.

Given the focus on distinguishing between DoS attacks, attack-induced resource overloads, and natural overloads, special attention was given to models that could capture complex patterns and interactions within the data. For instance, ensemble methods like Random Forest and AdaBoost were particularly effective in this regard due to their ability to handle a variety of data distributions and interactions.

Model Evaluation. Upon training, the models were evaluated on the testing set comparing the predictions against the actual values to calculate accuracy. However, to ensure a more robust performance estimate, we implemented cross-validation. StratifiedKFold cross-validation with five splits was employed, preserving the percentage of samples for each class in each fold. This method provides a comprehensive performance measure by averaging the results across multiple rounds of training and testing.

In addition to accuracy, we calculated the F1 macro scores for each model using cross-validation. The F1 macro scores accounts for both precision and recall, providing a balanced measure of model performance across all classes, including the detection of DoS attacks and differentiation between resource overloads (Fig. 1). Fig. 1 shows the F1 macro scores obtained for each machine learning classifier over multiple cross-validation folds. Each subplot demonstrates how a specific classifier performs consistently across different folds, providing insights into the robustness and generalizability of the models. The high consistency observed across folds indicates minimal variance in the performance, underlining the reliability of the classification models in detecting DoS attacks and distinguishing between resource overloads. The models were also evaluated based on their ability to minimize false positives and false negatives, which is crucial for maintaining the reliability of security systems in Kubernetes environments.

Novel Contributions. One of the novel aspects of our work is the ability of our machine learning models to distinguish between attack-induced and natural resource overloads. This is particularly important in Kubernetes environments where resource usage can fluctuate naturally due to varying workloads. By accurately identifying these differences, our models reduce the risk of false positives, thereby enhancing the operational efficiency of the security system. This ability to differentiate adds a layer of precision that is often lacking in traditional security approaches, making our solution particularly suited for dynamic, cloud-native environments.

The results from the cross-validation show that the models generally achieved high F1 macro scores, with the Random Forest Classifier and AdaBoost Classifier achieving top performance (0.9990 ± 0.0006 and $0.9990 \pm \pm 0.0003$, respectively). These results indicate strong model performance across various classifiers with minimal variability across folds, highlighting the robustness of our approach.

Confusion Matrices. To further elucidate model performance, we generated confusion matrices for each classifier (Fig. 2). Fig. 2 presents the confusion matrices derived from the cross-validation predictions of each classifier. The rows represent the actual class labels ('attack' and 'no attack'), while the columns show the predicted class labels. Diagonal elements indicate the number of correctly classified instances for each class, and the off-diagonal elements represent misclassifications. These matrices provide a visual assessment of the types of errors made by each classifier, helping to identify whether a model is prone to mistaking natural resource spikes for DoS attacks or vice versa. This analysis allows for targeted refinement of the models to reduce both false positives and false negatives. These matrices offer detailed insights into how well each model discriminates between the classes, highlighting areas where the model excels and where it requires improvement. For example, the confusion matrices help identify whether a model is particularly prone to mistaking natural overloads for DoS attacks or vice versa, allowing us to refine the models further.

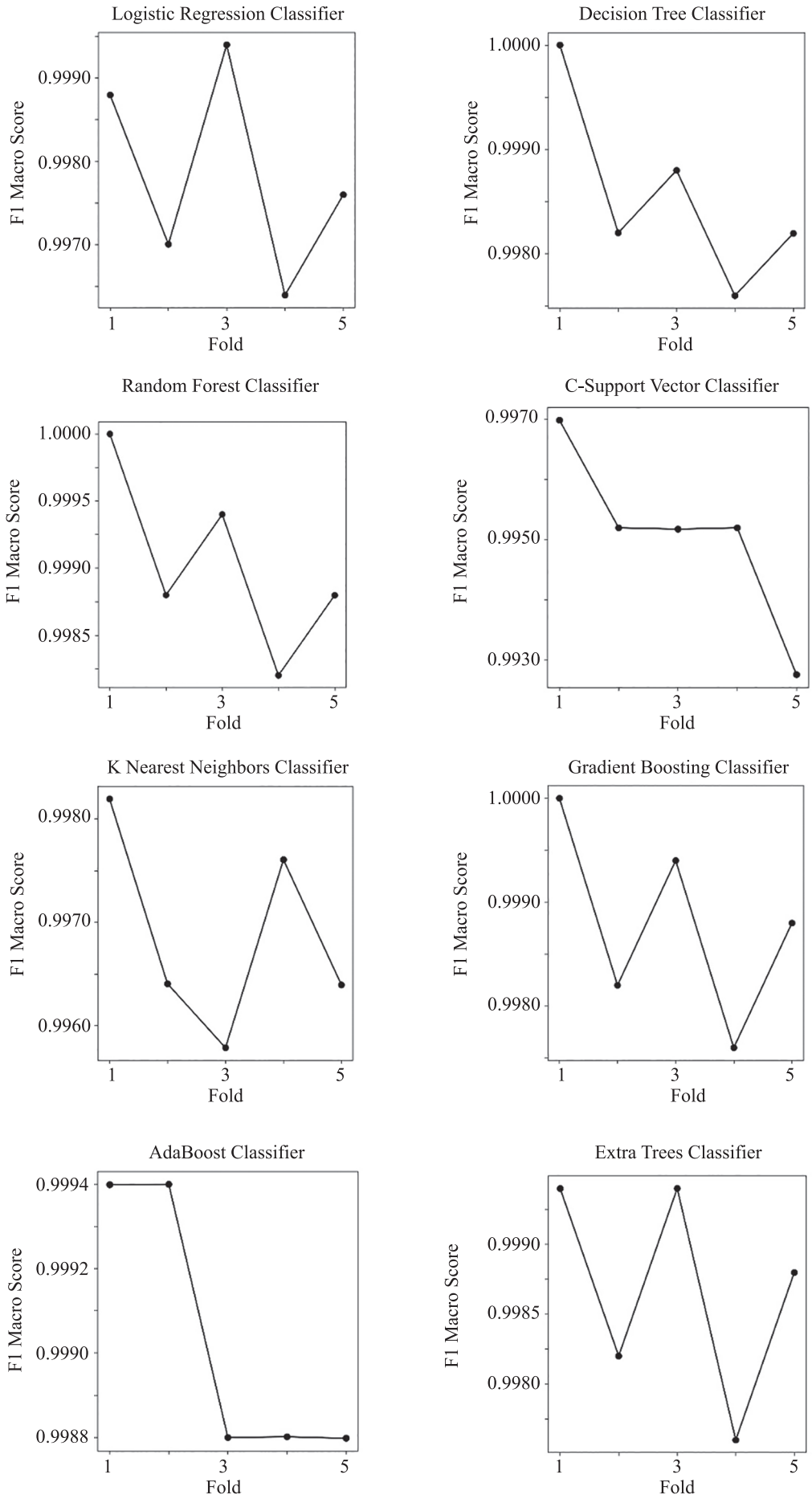

Fig. 1. F1 macro scores across cross-validation folds for various machine learning classifiers

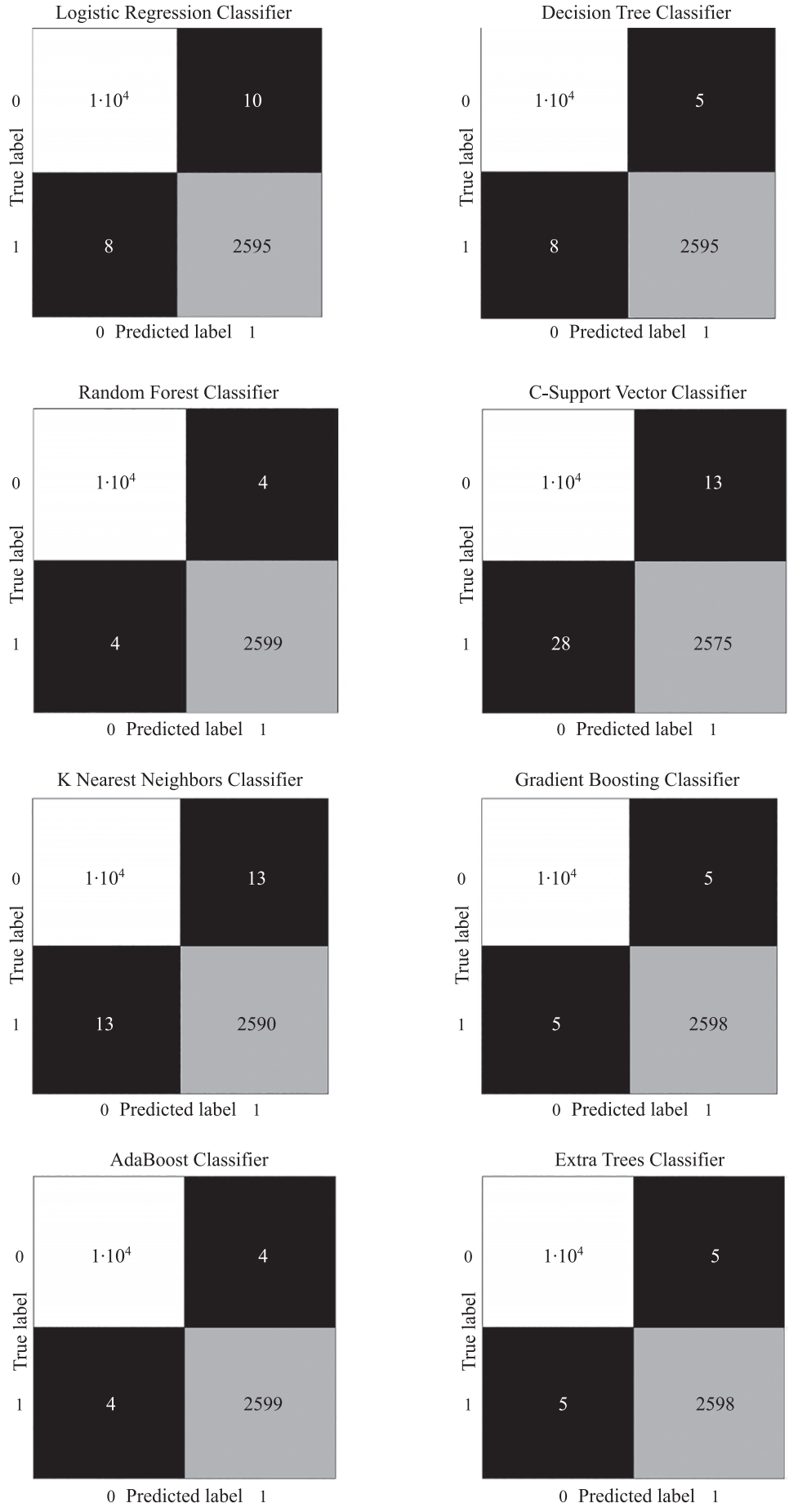

Fig. 2. Confusion matrices for various machine learning classifiers applied to the dataset

Conclusion

The dynamic and distributed nature of Kubernetes and microservices architecture necessitates advanced security measures that extend beyond traditional approaches. Our study has demonstrated that machine learning can significantly enhance security in Kubernetes environments by providing robust, real-time detection of attacks, including Denial of Service attacks. Our models are particularly effective in distinguishing between attack-induced resource overloads and natural fluctuations in resource usage, a critical capability for reducing false positives and maintaining operational efficiency.

By leveraging telemetry data collected from multiple sources, including real-world workloads, simulated hacking attempts, and induced overloading scenarios, we have developed a comprehensive dataset that enables our Machine Learning (ML) models to accurately detect and differentiate between various types of attacks. The integration of these models into production environments offers a proactive and adaptive approach to security, well-suited to the continuous integration and deployment cycles of modern software applications.

This research highlights the importance of integrating ML-based security solutions to maintain the integrity and reliability of Kubernetes deployments. The rigorous data processing and robust model evaluation techniques, such as cross-validation, confusion matrices, and F1 macro scores, have ensured the development of highly accurate and reliable machine learning models.

Looking forward, future work should focus on enhancing the interpretability of ML models, making it easier for security teams to understand and act upon the predictions made. Additionally, expanding the dataset to include even more diverse and complex scenarios will further improve the models ability to detect and respond to sophisticated attacks. Through these efforts, the security of Kubernetes environments can be continually improved, ensuring that they remain resilient in the face of evolving cyber threats.

References

1. Nobre J., Pires E.J., Reis A. Anomaly detection in microservice-based systems. *Applied Sciences*, 2023, vol. 13, no. 13, pp. 7891. https://doi.org/10.3390/app13137891
2. De Lauretis L. From monolithic architecture to microservices architecture. *Proc. of the 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 2019, pp. 93–96. https://doi.org/10.1109/issrew.2019.00050
3. Darwesh G., Hammoud J., Vorobeva A.A. A novel approach to feature collection for anomaly detection in Kubernetes environment and agent for metrics collection from Kubernetes nodes. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2023, vol. 23, no. 3, pp. 538–546. https://doi.org/10.17586/2226-1494-2023-23-3-538-546
4. Ghadeer D., Jaafar H., Vorobeva A.A. Security in kubernetes: best practices and security analysis. *Vestnik UrFO. Security in the Information Sphere*, 2022, no. 2(44), pp. 63–69.
5. Jacob S., Qiao Y., Ye Y., Lee B. Anomalous distributed traffic: Detecting cyber security attacks amongst microservices using graph convolutional networks. *Computers & Security*, 2022, vol. 118, pp. 102728. https://doi.org/10.1016/j.cose.2022.102728
6. Peralta-Garcia E., Quevedo-Monsalbe J., Tuesta-Monteza V., Arcila-Diaz J. Detecting structured query language injections in web microservices using machine learning. *Informatics*, 2024, vol. 11, no. 2, pp. 15. https://doi.org/10.3390/informatics11020015
7. Vinayakumar R., Alazab M., Soman K.P., Poornachandran P., Al-Nemrat A., Venkatraman S. Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 2019, vol. 7, pp. 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334
8. Zhang L., Cushing R., de Laat C., Grosso P. A real-time intrusion detection system based on OC-SVM for containerized applications. *Proc. of the 2021 IEEE 24th International Conference on Computational Science and Engineering (CSE)*, 2021, pp. 138–145. https://doi.org/10.1109/cse53436.2021.00029
9. Raj P., Vanga S., Chaudhary A. *Cloud-Native Computing: How to Design, Develop, and Secure Microservices and Event-Driven Applications*. John Wiley & Sons, 2022, 352 p.
10. Torkura K.A., Sukmana M.I.H., Meinel C. Integrating continuous security assessments in microservices and cloud native applications. *Proc. of the 10th International Conference on Utility and Cloud Computing, (UCC'17)*, 2017, pp. 171–180. https://doi.org/10.1145/3147213.3147229
11. Abed A.S., Clancy C., Levy D.S. Intrusion detection system for applications using linux containers. *Lecture Notes in Computer Science*, 2015, vol. 9331, pp. 123–135. https://doi.org/10.1007/978-3-319-24858-5_8
12. Zou Z., Xie Y., Huang K., Xu G., Feng D., Long D. A docker container anomaly monitoring system based on optimized isolation

Литература

1. Nobre J., Pires E.J., Reis A. Anomaly detection in microservice-based systems // Applied Sciences. 2023. V. 13. N 13. P. 7891. https://doi.org/10.3390/app13137891
2. De Lauretis L. From monolithic architecture to microservices architecture // Proc. of the 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). 2019. P. 93–96. https://doi.org/10.1109/issrew.2019.00050
3. Darwesh G., Hammoud J., Vorobeva A.A. A novel approach to feature collection for anomaly detection in Kubernetes environment and agent for metrics collection from Kubernetes nodes // Научно-технический вестник информационных технологий механики и оптики. 2023. Т. 23. № 3. С. 538–546. https://doi.org/10.17586/2226-1494-2023-23-3-538-546
4. Ghadeer D., Jaafar H., Vorobeva A.A. Security in kubernetes: best practices and security analysis // Вестник УрФО. Безопасность в информационной сфере. 2022. № 2(44). С. 63–69.
5. Jacob S., Qiao Y., Ye Y., Lee B. Anomalous distributed traffic: Detecting cyber security attacks amongst microservices using graph convolutional networks // Computers & Security. 2022. V. 118. P. 102728. https://doi.org/10.1016/j.cose.2022.102728
6. Peralta-Garcia E., Quevedo-Monsalbe J., Tuesta-Monteza V., Arcila-Diaz J. Detecting structured query language injections in web microservices using machine learning // Informatics. 2024. V. 11. N 2. P. 15. https://doi.org/10.3390/informatics11020015
7. Vinayakumar R., Alazab M., Soman K.P., Poornachandran P., Al-Nemrat A., Venkatraman S. Deep learning approach for intelligent intrusion detection system // IEEE Access. 2019. V. 7. P. 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334
8. Zhang L., Cushing R., de Laat C., Grosso P. A real-time intrusion detection system based on OC-SVM for containerized applications // Proc. of the 2021 IEEE 24th International Conference on Computational Science and Engineering (CSE). 2021. P. 138–145. https://doi.org/10.1109/cse53436.2021.00029
9. Raj P., Vanga S., Chaudhary A. Cloud-Native Computing: How to Design, Develop, and Secure Microservices and Event-Driven Applications. John Wiley & Sons, 2022. 352 p.
10. Torkura K.A., Sukmana M.I.H., Meinel C. Integrating continuous security assessments in microservices and cloud native applications // Proc. of the 10th International Conference on Utility and Cloud Computing, (UCC'17). 2017. P. 171–180. https://doi.org/10.1145/3147213.3147229
11. Abed A.S., Clancy C., Levy D.S. Intrusion detection system for applications using linux containers // Lecture Notes in Computer Science. 2015. V. 9331. P. 123–135. https://doi.org/10.1007/978-3-319-24858-5_8
12. Zou Z., Xie Y., Huang K., Xu G., Feng D., Long D. A docker container anomaly monitoring system based on optimized isolation

forest. *IEEE Transactions on Cloud Computing*, 2022, vol. 10, no. 1, pp. 134–145. https://doi.org/10.1109/tcc.2019.2935724
13. Srinivasan S., Kumar A., Mahajan M., Sitaram D., Gupta S. Probabilistic real-time intrusion detection system for docker containers. *Communications in Computer and Information Science*, 2019, vol. 969, pp. 336–347. https://doi.org/10.1007/978-981-13-5826-5_26
14. Cavalcanti M., Inacio P., Freire M. Performance evaluation of container-level anomaly-based intrusion detection systems for multi-tenant applications using machine learning algorithms. *Proc. of the 16th International Conference on Availability, Reliability and Security (ARES'21)*, 2021, pp. 1–9. https://doi.org/10.1145/3465481.3470066
15. Flora J., Gonçalves P., Antunes N. Using attack injection to evaluate intrusion detection effectiveness in container-based systems. *Proc. of the IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2020, pp. 60–69. https://doi.org/10.1109/prdc50213.2020.00017
16. Tunde-Onadele O., He J., Dai T., Gu X. A study on container vulnerability exploit detection. *Proc. of the IEEE International Conference on Cloud Engineering (IC2E)*, 2019, pp. 121–127. https://doi.org/10.1109/ic2e.2019.00026
17. Lin Y., Tunde-Onadele O., Gu X. CDL: Classified distributed learning for detecting security attacks in containerized applications. *Proc. of the 36th Annual Computer Security Applications Conference (ACSAC'20)*, 2020, pp. 179–188. https://doi.org/10.1145/3427228.3427236
18. Huang L., Ma D., Li S., Zhang X., Wang H. Text level graph neural network for text classification. *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3444–3450. https://doi.org/10.18653/v1/d19-1345
19. Haq M.S., Nguyen T.D., Tosun A.S., Vollmer F., Korkmaz T., Sadeghi A.-R. SoK: A comprehensive analysis and evaluation of docker container attack and defense mechanisms. *Proc. of the IEEE Symposium on Security and Privacy (SP)*, 2024, pp. 4573–4590. https://doi.org/10.1109/sp54263.2024.00268
20. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay É. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011, vol. 12, pp. 2825–2830.

forest // IEEE Transactions on Cloud Computing. 2022. V. 10. N 1. P. 134–145. https://doi.org/10.1109/tcc.2019.2935724
13. Srinivasan S., Kumar A., Mahajan M., Sitaram D., Gupta S. Probabilistic real-time intrusion detection system for docker containers // Communications in Computer and Information Science. 2019. V. 969. P. 336–347. https://doi.org/10.1007/978-981-13-5826-5_26
14. Cavalcanti M., Inacio P., Freire M. Performance evaluation of container-level anomaly-based intrusion detection systems for multi-tenant applications using machine learning algorithms // Proc. of the 16th International Conference on Availability, Reliability and Security (ARES'21). 2021. P. 1–9. https://doi.org/10.1145/3465481.3470066
15. Flora J., Gonçalves P., Antunes N. Using attack injection to evaluate intrusion detection effectiveness in container-based systems // Proc. of the IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC). 2020. P. 60–69. https://doi.org/10.1109/prdc50213.2020.00017
16. Tunde-Onadele O., He J., Dai T., Gu X. A study on container vulnerability exploit detection // Proc. of the IEEE International Conference on Cloud Engineering (IC2E). 2019. P. 121–127. https://doi.org/10.1109/ic2e.2019.00026
17. Lin Y., Tunde-Onadele O., Gu X. CDL: Classified distributed learning for detecting security attacks in containerized applications // Proc. of the 36th Annual Computer Security Applications Conference (ACSAC'20). 2020. P. 179–188. https://doi.org/10.1145/3427228.3427236
18. Huang L., Ma D., Li S., Zhang X., Wang H. Text level graph neural network for text classification // Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2019. P. 3444–3450. https://doi.org/10.18653/v1/d19-1345
19. Haq M.S., Nguyen T.D., Tosun A.S., Vollmer F., Korkmaz T., Sadeghi A.-R. SoK: A comprehensive analysis and evaluation of docker container attack and defense mechanisms // Proc. of the IEEE Symposium on Security and Privacy (SP). 2024. P. 4573–4590. https://doi.org/10.1109/sp54263.2024.00268
20. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay É. Scikit-learn: Machine learning in Python // Journal of Machine Learning Research. 2011. V. 12. P. 2825–2830.

Authors

Ghadeer Darwesh — PhD Student, ITMO University, Saint Petersburg, 197101, Russian Federation, sc 57226287648, https://orcid.org/0000-0003-1116-9410, ghadeerdarwesh32@gmail.com
Jaafar Hammoud — PhD Student, ITMO University, Saint Petersburg, 197101, Russian Federation, sc 57222044000, https://orcid.org/0000-0002-2033-0838, hammoudgj@gmail.com
Alisa A. Vorobeva — PhD, Associate Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, sc 57191359167, https://orcid.org/0000-0001-6691-6167, vorobeva@itmo.ru

Авторы

Дарвиш Гадир — аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, sc 57226287648, https://orcid.org/0000-0003-1116-9410, ghadeerdarwesh32@gmail.com
Хаммуд Жаафар — аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, sc 57222044000, https://orcid.org/0000-0002-2033-0838, hammoudgj@gmail.com
Воробьева Алиса Андреевна — кандидат технических наук, доцент, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, sc 57191359167, https://orcid.org/0000-0001-6691-6167, vorobeva@itmo.ru

Received 20.06.2024
Approved after reviewing 08.10.2024
Accepted 15.11.2024

Статья поступила в редакцию 20.06.2024
Одобрена после рецензирования 08.10.2024
Принята к печати 15.11.2024