Anomalous Distributed Traffic Detection in Microservices with Clustering and Attention-based Graph Neural Networks

Javad Dogani

Department of Computer Science and Engineering and IT, School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran

Email: j.dogani@shirazu.ac.ir

1. INTRODUCTION

Microservice architecture represents a software design methodology in which a large-scale application is decomposed into smaller, independent units that interoperate via APIs or RPC [1] [1]. Each microservice embodies a self-contained module of functionality, enabling it to be developed, tested, and deployed in isolation, thereby promoting better adaptability and faster delivery [2] [2]. The proliferation of container-based architecture has expedited the deployment of microservices, offering a lightweight and portable approach to packaging and deploying microservices, simplifying the management of individual components' scalability and deployment [3, 4][3, 4]. Nevertheless, the growing adoption of microservices architecture and container technology has also engendered security challenges for cloud environments [5, 6]. In contrast to VM-based cloud infrastructure, microservice design confers a flexibility on service scalability and maintenance. Notably, all containers within a system share a single operating system kernel [6]. Thus, if one container is compromised by incorrect configuration, it may expose all other containers, increasing the complexity and risk introduced by containers [7]. Additionally, the update frequency of individual services can exceed hundreds of times per day [8], further complicating the dynamic and complex nature of cloud environments and potentially introducing new security vulnerabilities. Despite the many benefits of microservices, their inherent flexibility has the potential to impact the security posture of cloud environments, requiring the development of effective strategies for addressing associated risks.

One of the most significant impediments to adopting microservices architecture is ensuring the security of communication between individual microservices. [9, 10]. Furthermore, the growing complexity of container environments can create challenges in detecting and responding to security incidents. It is simply not feasible to manually manage security policies for the growing number of containers and microservices, necessitating policy automation to enhance the security of complex systems, such as those built on container-based microservices architecture [11]. The automated management of security policies is indispensable, allowing organizations to efficiently manage security policies at scale, promote consistency, comply with regulations, and respond swiftly to security incidents.

Microservices utilize the Remote Procedure Call (RPC) protocol to communicate with one another. Each microservice operates within its container and coordinates with other microservices using

RPC to execute their respective tasks. Nevertheless, any anomalies in RPC communication between microservices' containers, such as a sudden spike in volume or unusual communication patterns, may indicate an attempt to exploit system vulnerabilities [12]. Attackers may, for example, inundate a specific container with a massive number of requests to overload the system and launch a denial of service (DoS) attack. Therefore, it is critical to monitor RPC communication between microservices' containers and promptly identify any irregularities to ensure security [13]. Machine learning frameworks aid in this area by employing algorithms to learn from historical RPC traffic data and predict future traffic trends. By using machine learning algorithms, the system can detect anomalies in real time and notify the security team with all the necessary information to address the issue. Furthermore, machine learning algorithms can evaluate enormous amounts of data and identify intricate traffic patterns that conventional security tools may be unable to detect [14]. This significantly improves the accuracy and effectiveness of the detection process while reducing the likelihood of false positive alerts.

The microservices architecture presents a challenge for traditional machine learning models as they are limited to processing either graphs or time-series data, but not both simultaneously. The spatial dependencies between microservices are typically represented as graphs, while the time-series data represents the history of incoming traffic to the system over time [15]. To accurately detect anomalies in RPC traffic, it is essential to examine both the spatial dependencies and temporal fluctuations. However, the dynamic nature of microservices makes it challenging for a single global model to effectively capture the relationships between microservices and recognize unusual fluctuations in RPC traffic. To address this challenge, clustering techniques can be used to group relevant microservices and train a prediction model in each cluster, resulting in higher accuracy. In this regard, this proposal tries to propose a solution that combines clustering algorithms with Graph Convolution Networks (GCNs) [16] and an attention-based mechanism, called AEDGCN. GCNs are a type of neural network that can process graph data and learn complex spatial relationships between nodes in the graph, enabling the model to detect temporal dependencies in the input sequence and deliver more accurate predictions. By using clustering, attention-based mechanisms, and GCNs, the proposed model can examine the interconnections between microservices and the changes in RPC traffic over time, resulting in a more comprehensive and accurate understanding of the system.

2. RESEARCH OBJECTIVES

The increasing adoption of microservices architecture and container technology in cloud environments has resulted in new security challenges, especially in ensuring the security of communication between individual microservices. Detecting anomalies in RPC communication between microservices' containers is critical for identifying security threats, but traditional machine learning models have limitations in processing both the spatial dependencies and temporal fluctuations inherent in microservices. Therefore, there is a need to develop a solution that can accurately detect security threats in microservices architecture by combining clustering algorithms with Graph Convolution Networks (GCNs) and attention-based mechanisms.

The main objective of this research is to develop an effective solution for detecting security threats in microservices architecture by combining clustering algorithms with GCNs and attention-based mechanisms. The specific objectives include:

1. To investigate the security challenges associated with microservices architecture and container technology in cloud environments.
2. To examine the limitations of traditional machine learning models in processing both spatial and temporal data in a microservices architecture.
3. To develop an approach that combines clustering algorithms, GCNs, and attention-based mechanisms for detecting anomalies in RPC communication between microservices' containers.
4. To evaluate the proposed approach's effectiveness in detecting security threats in microservices architecture using real-world datasets.
5. To compare the performance of the proposed approach with traditional machine learning models and state-of-the-art methods for anomaly detection in a microservices architecture.

3. LITERATURE REVIEW

This section provides a review of literature related to graph-based anomaly detection on network traffic using machine learning approaches. Deep neural networks (DNN) have been utilized to model data and extract the underlying behaviour. Recurrent neural networks (RNN), a specific type of neural network, have been employed to model sequential data. Typically, such data is represented as a sequence of process events, known as a case. In a study [17], a graph convolutional neural network (GCNN) was utilized to detect irregular real-world remote procedure call (RPC) traffic. This study aimed to identify cybersecurity issues in thousands of RPCs generated by multiple microservices. In this work, a two-step process was utilized to trace and log the RPC traffic and detect anomalies. Firstly, the logged RPC traffic from active microservice functionality was analyzed, and correlations between RPC chain patterns in the data were identified using a density-clustering algorithm. These chain patterns represent a subsystem of the overall microservice functionality. A GCNN was then used to model each subsystem of the RPC traffic and learn the spatiotemporal dependencies of the traffic to solve the irregular RPC prediction problem. By utilizing these GCNNs, individual predictions can be made for each pre-existing sub-system. This approach was evaluated using two case studies, which involved real-world malicious traffic threat models, such as batch registration of bot accounts and account cracking.

In a related study [18], the researchers utilized a specific type of RNN called Long Short Term Memory (LSTM) neural network [19] to detect both long and short-term dependencies in cases. This LSTM-based framework was used to learn the patterns of cases and subsequently predict future events and their timestamps. The researchers evaluated the performance of this framework by training it on logged cases from two available datasets, and the results showed that it outperformed a previous methodology [20]. In addition, deep learning models such as Convolutional Neural Networks (CNN) have been used to model network traffic flow. The CNN model is particularly suited for modelling and analyzing graph constructs and imagery, and it can observe and learn the spatial relations of the traffic flow. For instance, in [21], the authors proposed a CNN model to learn network traffic as images to capture the spatial and temporal dynamics of the data and predict the network traffic speed. The effectiveness of this CNN algorithm was tested using two datasets composed of real-world transportation traffic. The authors of [22] proposed a traffic dispersion graph (TDG) methodology that uses a dynamic aspect to model the dependencies of the temporal dynamics of the TDGs over time. Anomalous traffic is detected by analyzing irregular network traffic occurring over time. The TDG method was evaluated using two data sets of traffic traces and was able to detect a cyber-attack with 100% accuracy.

The study [23] proposed a high-level attack detection framework for network communication data by using a hybrid CNN/LSTM deep learning model called STDeepGraph. The model uses a temporal communication graph to model the network communication structure and a distance graph kernel to map the communication into a high-dimensional space. The CNN component is used for extracting the spatial features of the network flow, and the LSTM for the temporal features. The model uses a softmax classification function to classify the network traffic as benign or malicious. The STDeepGraph was evaluated using real-world network attack data sets, and the results show that this method outperforms baseline methods in terms of accuracy and loss. The authors of [24] proposed a deep learning model that takes a graph representation of traffic-based data, transforms it over time, and learns the spatiotemporal dynamics of the data. The model was used to predict dynamic anomalies by measuring the non-Euclidean distance between the actual values and the output predictions. This was done by computing the affinity score of an existing data entity. A threshold value is established to detect anomalous behaviour. The model was evaluated using two available traffic-related data sets of network traffic and public transport traffic, and the results showed that the model had competitive results that were comparable to state-of-the-art techniques.

4- METHODOLOGY

The first phase of our method involves defining the distance between two RPC chains and applying the density-based clustering technique DBSCAN to identify unique RPC chain patterns. Performing clustering is vital because each microservice cluster could have its unique behaviour and patterns of communication, which can be tough to portray with a single global model. Building individual models per cluster can capture the distinctive flow and structure of each microservice properly, making it simpler to identify anomalous changes in traffic. In the second phase, we will adopt a spatiotemporal attention-based graph convolution network (GCN) approach for each RPC chain cluster, as opposed to a single unified model. This allows our method to dynamically weigh the importance of different nodes and edges in each RPC chain pattern, leading to improved prediction accuracy. To solve the prediction problems, we propose an Attention Based Spatio-Temporal Graph Convolutional Network considering External features. The goal of the attention mechanism is to select from all inputs information that is relatively important to the current task. By using attention in combination with GCN, our method can effectively capture the complex relationships between entities in microservices and make more accurate predictions about incoming RPC traffic. In the third phase of our method, the predicted traffic will be used to detect any unusual or malicious activities that might pose a threat to the security of the microservice architecture. This is achieved by monitoring the incoming traffic and comparing it with the predicted traffic. If there is any significant deviation from the predicted traffic, it could indicate the presence of an attack. The system then raises an alarm and further analysis can be performed to confirm the presence of an attack. This early detection of potential threats helps organizations to take proactive measures to mitigate the risk and prevent any harm to their microservice architecture.

Our method will be rigorously tested and evaluated through various experiments. Our approach to predicting RPC traffic in container-based microservices has been applied to a large dataset sampled from real-world Kubernetes production systems, which handle billions of daily active users. The combination of density-based clustering and spatial-temporal attention-based graph convolution network proves to be a reliable solution for predicting and detecting security threats in complex

microservice systems. Our approach will provide a comprehensive solution for securing microservices by combining the power of machine learning, clustering techniques, and attention mechanisms to monitor and predict RPC traffic in real-time.

We will use a microservices application called DeathStarBench, which includes several different microservice-based applications, such as a social networking app, a banking system, a media service, and a hotel reservation service [25]. We will choose to use the social networking application, named SocialNetwork and create a directed graph of the social network by registering users, establishing relationships between them, and representing them as nodes with edges representing their follow relationships. We then will use API requests and HTTP traffic generators to generate regular traffic and simulate cyber-attacks against the application. We will perform three separate experiments, each with a different type of cyber-attack. In each experiment, we will generate regular traffic and anomalous traffic caused by the cyber-attack. The regular traffic mainly consisted of RPC traffic returned in response to API requests to upload user posts, register new users, and log in to the application to view other users' timelines.

5- TIMELINE

Here is a possible timeline for completing the proposed research in 6 months:

Month 1:

- Conduct a comprehensive literature review on microservices architecture, container technology, security challenges, and machine learning techniques for anomaly detection in RPC traffic.
- Define the research problem and objectives.
- Develop a detailed research plan and timeline.

Month 2:

- Collect and preprocess data on RPC traffic in a microservices architecture.
- Implement clustering algorithms to group relevant microservices.
- Develop Graph Convolution Networks (GCNs) and an attention-based mechanism.

Month 3:

- Train prediction models in each cluster.
- Evaluate the performance of the prediction models using standard metrics.
- Refine the model to improve its performance.

Month 4:

- Test the proposed model on real-world data and evaluate its effectiveness in detecting anomalies in RPC traffic.
- Compare the proposed model with existing machine learning models.

Month 5:

- Write the results section and draft the discussion section of the research paper.
- Finalize the experimental setup and ensure the reproducibility of the results.

Month 6:

- Complete the manuscript by writing the introduction, methodology, and conclusion sections.
- Submit the research paper to a suitable conference or journal.

Of course, this timeline may need to be adjusted based on factors such as the availability of data, the complexity of the model, and unexpected challenges that may arise during the research process. It is essential to maintain flexibility and adapt the timeline as needed to ensure that the research is thorough and of high quality.

References

[1] M. Abdullah, W. Iqbal, and A. Erradi, "Unsupervised learning approach for web application auto-decomposition into microservices," *Journal of Systems and Software*, vol. 151, pp. 243-257, 2019/05/01/ 2019.

[2] M. Usman, S. Ferlin, A. Brunstrom, and J. Taheri, "A Survey on Observability of Distributed Edge & Container-Based Microservices," *IEEE Access*, vol. 10, pp. 86904-86919, 2022.

[3] A. Malviya and R. K. Dwivedi, "Designing Architecture for Container-As-A-Service (CaaS) in Cloud Computing Environment: A Review," in *Proceedings of 3rd International Conference on Machine Learning, Advances in Computing, Renewable Energy and Communication*, Singapore, 2022, pp. 549-563: Springer Nature Singapore.

[4] Z. Li, H. Jin, D. Zou, and B. Yuan, "Exploring New Opportunities to Defeat Low-Rate DDoS Attack in Container-Based Cloud Environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 695-706, 2020.

[5] S. Taherizadeh and M. Grobelnik, "Key influencing factors of the Kubernetes auto-scaler for computing-intensive microservice-native cloud-based applications," *Advances in Engineering Software*, vol. 140, p. 102734, 2020/02/01/ 2020.

[6] F. Minna and F. Massacci, "SoK: Run-time security for cloud microservices. Are we there yet?," *Computers & Security*, vol. 127, p. 103119, 2023/04/01/ 2023.

[7] N. Mateus-Coelho, M. Cruz-Cunha, and L. G. Ferreira, "Security in Microservices Architectures," *Procedia Computer Science*, vol. 181, pp. 1225-1236, 2021/01/01/ 2021.

[8] U. Qazi, M. Imran, and F. Ofli, "GeoCoV19: a dataset of hundreds of millions of multilingual COVID-19 tweets with location information," vol. 12, no. 1 %J SIGSPATIAL Special, pp. 6–15, 2020.

[9] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open RAN security: Challenges and opportunities," *Journal of Network and Computer Applications*, vol. 214, p. 103621, 2023/05/01/ 2023.

[10] A. Masood, D. S. Lakew, and S. Cho, "Security and Privacy Challenges in Connected Vehicular Cloud Computing," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2725-2764, 2020.
[11] A. Hannousse and S. Yahiouche, "Securing microservices and microservice architectures: A systematic mapping study," *Computer Science Review*, vol. 41, p. 100415, 2021/08/01/ 2021.
[12] M. Repetto, A. Carrega, and R. Rapuzzi, "An architecture to manage security operations for digital service chains," *Future Generation Computer Systems*, vol. 115, pp. 251-266, 2021/02/01/ 2021.
[13] C. Rajasekharaiah, "Securing Microservices on Cloud," in *Cloud-Based Microservices: Techniques, Challenges, and Solutions*, C. Rajasekharaiah, Ed. Berkeley, CA: Apress, 2021, pp. 179-202.
[14] M. Mahdavisharif, S. Jamali, and R. Fotohi, "Big Data-Aware Intrusion Detection System in Communication Networks: a Deep Learning Approach," *Journal of Grid Computing*, vol. 19, no. 4, p. 46, 2021/10/30 2021.
[15] C. Qiu, K. Yang, J. Wang, and S. Zhao, "AI-Empowered Network Root Cause Analysis for 6G," *IEEE Network*, pp. 1-9, 2023.
[16] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust Graph Convolutional Networks Against Adversarial Attacks," presented at the Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 2019. Available: https://doi.org/10.1145/3292500.3330851
[17] J. Chen, S. Lu, X. Weng, Z. Liang, and X. Wu, "Heterogeneity of antigen specificity between HLA-A*02:01 and other frequent Chinese HLA-A2 subtypes detected by a modified autologous lymphocyte-monocyte coculture," *Molecular Immunology*, vol. 114, pp. 389-394, 2019/10/01/ 2019.
[18] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, "Predictive Business Process Monitoring with LSTM Neural Networks," in *Advanced Information Systems Engineering*, Cham, 2017, pp. 477-492: Springer International Publishing.
[19] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.
[20] M. Polato, A. Sperduti, A. Burattin, and M. d. Leoni, "Time and activity sequence prediction of business process instances," *Computing*, vol. 100, no. 9, pp. 1005-1031, 2018/09/01 2018.
[21] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction," vol. 17, no. 4, p. 818, 2017.
[22] D. Q. Le, T. Jeong, H. E. Roman, and J. W.-K. Hong, "Traffic dispersion graph based anomaly detection," presented at the Proceedings of the 2nd Symposium on Information and Communication Technology, Hanoi, Vietnam, 2011. Available: https://doi.org/10.1145/2069216.2069227
[23] Y. Yao, L. Su, Z. Lu, and B. Liu, "STDeepGraph: Spatial-Temporal Deep Learning on Communication Graphs for Long-Term Network Attack Detection," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 120-127.

[24] J. Lee, H. Bae, and S. Yoon, "Anomaly Detection by Learning Dynamics From a Graph," *IEEE Access*, vol. 8, pp. 64356-64365, 2020.

[25] Y. Gan *et al.*, "An Open-Source Benchmark Suite for Microservices and Their Hardware-Software Implications for Cloud & Edge Systems," presented at the Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, Providence, RI, USA, 2019. Available: https://doi.org/10.1145/3297858.3304013