Proactive Threat Detection in Kubernetes Cluster

Rahul Kinnera
Polytechnic Institute
Purdue University
West Lafayette, IN
rkinnera@purdue.edu

***Abstract*—The rapid adoption of Kubernetes for container orchestration has introduced complex security challenges that traditional methods struggle to address. This paper explores the integration of Zeek, an advanced network traffic analysis tool, with the Malware Information Sharing Platform (MISP) for proactive threat detection in Kubernetes environments. Leveraging Zeek's traffic inspection and MISP's threat intelligence capabilities, the proposed solution identifies and mitigates network-based threats. Our experimental results demonstrate enhanced detection accuracy and operational scalability, validating the framework's effectiveness in securing dynamic Kubernetes clusters.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The rapid growth in cloud environments has fundamentally transformed application management and deployment within organizations. This transformation is closely linked to the increasing adoption of cloud-native architectures, which has led to an exponential rise in the size and complexity of these environments [1]. As a result, modern networks are generating large volumes of traffic, which poses significant challenges for effective monitoring [2]. This increase in traffic not only enhances operational capabilities but also expands the attack surface, making organizations more vulnerable to sophisticated and dynamic cyber threats [3].

Proactive cybersecurity measures are essential for organizations to mitigate risks effectively. It is widely recognized that having a robust cybersecurity policy in place proactively is far more effective than reactive measures, which can allow attackers to exploit vulnerabilities before they are detected [4]. Proactive detection methods enable organizations to identify potential threats early, thereby reducing the dwell time of attackers within a system and minimizing potential damage.

As network traffic continues to scale, the complexity of detecting malicious activities also increases. Traditional Intrusion Detection Systems (IDS) struggle to keep pace with the volume and speed of data generated by large-scale infrastructures [5]. Recent advancements in Machine Learning (ML) and Artificial Intelligence (AI) have shown promise in addressing these challenges; however, these methods often require substantial computational resources, which can escalate operational costs and limit their deployment in cost-sensitive environments [6].

Kubernetes emerges as a powerful container orchestration platform that offers solutions to these challenges. Its inherent scalability and ability to deploy containerized applications make it a cost-effective option for managing complex cloud environments [1]. Furthermore, integrating tools like Zeek for network traffic analysis and MISP for threat intelligence can create a robust framework for proactive threat detection within Kubernetes clusters [3]. This integration can help avoid the high infrastructure costs associated with ML/AI while ensuring efficient resource utilization and adaptability [1].

This work investigates the integration of Zeek and MISP within Kubernetes in order to address the pressing need for scalable and cost-efficient threat detection. By leveraging the orchestration capabilities of Kubernetes, the proposed framework will merge real-time network monitoring with actionable threat intelligence. The resulting system will detect anomalies and correlate them with external intelligence for comprehensive protection of large-scale cloud environments. This paper assesses the sufficiency of this approach and pinpoints a probably practical alternative to the resource-consuming systems based on ML/AI.

II. Background and Related Work

Over the past few years, many research works have surfaced that propose intrusion detection solutions in cloud environments by integrating with other technologies like artificial intelligence and machine learning. However, the deployment of IDS has been considered as a major issue in terms of cloud security [7]. Lata and Singh [7] found the existing literature suggests that deploying IDS on every single VM while improving the security posture increases computational overhead, whereas deploying IDS on vSwitch or Hypervisor made context information collection difficult. Also, the placement of the IDS server within or outside the cloud server also provides different results in terms of threat detection[8].

Next, the requirement of resource-intensive ML models needs to be questioned, and recent studies imply the possibility of accurately detecting time series data without the use of machine learning techniques[9]. Studies have proven that preliminary anomaly detection using correlational data mining techniques is effective for time-series-based network logs[10]. While the papers work on different experimental setups, the results achieved in these articles suggest an exploration for IDS without involving AI or ML.

Kubernetes offers a scalable and cost-effective method for deploying IDS. In the context of Kubernetes, cluster optimization and monitoring are vital for maintaining performance

and security. Hadikusuma et al. emphasize the necessity of employing robust monitoring frameworks such as Prometheus and Grafana, which facilitate real-time monitoring and visualization of cluster metrics [11]. These tools not only enhance resource management but also play a crucial role in identifying anomalies that may indicate security breaches. However, these tools focus on Kubernetes-specific data, and to protect large cloud environments, it is necessary to utilize network logs from a holistic cloud environment point of view. IDS should ultimately focus not just on the Kubernetes cluster but also detect threats related to the entire cloud deployment

Literature also suggests that Zeek performed well in IDS-only mode when compared to Snort and Suricata [12]. Zeek IDS is capable of detecting intrusions at high speeds (Gigabits per second (Gbps)) and in large volumes of traffic, making it suitable for our proposal [13]. Using Zeek IDS with its scripting possibilities makes managing threats, logging, and even after-detection tasks easier[13]. Moreover, the Malware Intelligence Sharing Platform (MISP) serves as a critical component in the proactive threat detection landscape. It enables organizations to share threat intelligence effectively, enhancing their ability to detect and respond to emerging threats. The sharing of indicators of compromise (IoCs) through platforms like MISP allows for a more informed and rapid response to potential attacks, as highlighted by the work of Vakilinia and Sengupta, who discuss the importance of information sharing in mitigating sophisticated cyber threats[14].

Background literature ultimately points out that integrating Zeek and MISP Data as a Kubernetes cluster forms a robust framework for enhancing security in Kubernetes clusters. This multifaceted approach enables organizations to not only detect and respond to threats more effectively but also to anticipate potential vulnerabilities before they can be exploited

III. Proposed Work

A. High Level Workflow

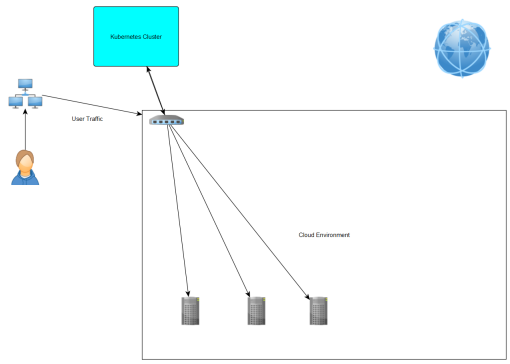

Fig. 1. Assumed workflow of intelligent IDS K8s based cloud environment

Based on the existing literature, Fig 1 shows a high-level architecture or workflow of how threat-intelligent IDS-based Kubernetes cluster will be deployed in a use-case scenario. Users connect to the cloud environment via layer two or three switches depending upon the cloud deployment under consideration. We presume these switches have some level of software-defined networking capabilities. Utilizing those features, the switches route critical traffic to a Kubernetes cluster deployed locally in a local server or private cloud. These assumptions promote the separation of duties and also encourage defense in depth as we are outsourcing network packet analysis, relieving loads that a switch can handle.

B. Kubernetes Cluster

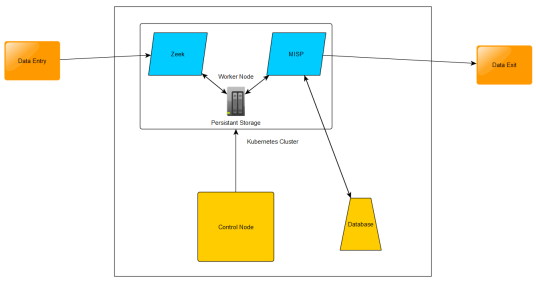

Fig. 2. Structure of Kubernetes Cluster

We propose a multi-node Kubernetes cluster with a database for proactive threat detection, like Fig 2. The Kubernetes cluster receives network traffic from the network switches, analyzes the data, and sends policies that must be enforced to the switches to mitigate threats. Kubernetes cluster consists of a control node, a worker node and a database. The functions of these components are as follows:

1) Control Node: The control node has a typical configuration with docker, containerd, kubelet, kubeapi, and kubectl packages installed in the VM. The control node is initialized with addresses and capacity to handle the worker node, and metrics API has to be enabled to track resource utilization. The control node is not directly connected to the incoming traffic to ensure controller safety and avoid the creation of rogue pods.

2) Worker Node: The worker node has the same packages as the control node and is connected to the control plane. An additional package installed in the worker node includes Kompose.io. Kompose.io converts Dockerfiles into YAML Manifests, allowing us to deploy Docker images as Kubernetes pods. The worker node consists of the pods, namely Zeek and MISP. MISP (Malware Intelligence Sharing Platform) and Zeek are deployed as stateful pods using Kompose-generated YAML manifests and connected to the control node. The MISP pod is connected to the database instance, and once the MISP pod is up and running, threat intelligence data is stored in the database under the MISP schema. Zeek captures network traffic from one of the main NICs (e.g., *eth0*) and stores it as pcap files, or it can process live network traffic directly, depending on personal choice. These pods are connected through a shared persistent storage where MISP's IoCs are sent to Zeek as a threat intelligence file (Fig 3 1). Based on the

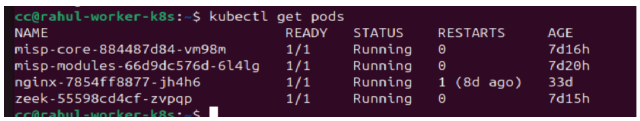

Fig. 3. MISP and Zeek Pods

file, Zeek classifies packets by reading and including the intel file in Zeek's configuration. Later, Zeek pods were restarted, which enabled the changes made to the configuration file. Similarly, the anomalies detected by Zeek are also stored in the

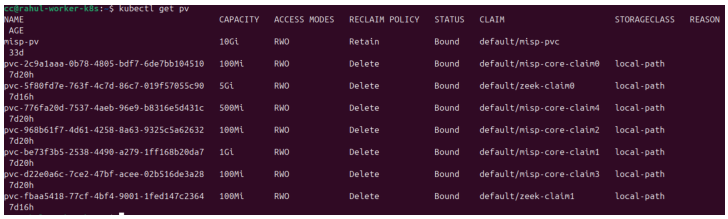

Fig. 4. Persistent Volumes and shared PV zeekclaim1

shared storage in the form of http.log, conn.log, and dns.log. These logs are converted into MISP interpretable format using Python-based API, and further context is added to enrich the MISP data periodically.

C. Database

MISP data is hosted on the database node. The MISP API connects to the database, fetches data when needed by the Zeek pod, and converts it into Zeek interpretable form. The algorithm that explains the workflow of MISP API has been mentioned in algorithm 1.

IV. Experimental Setup

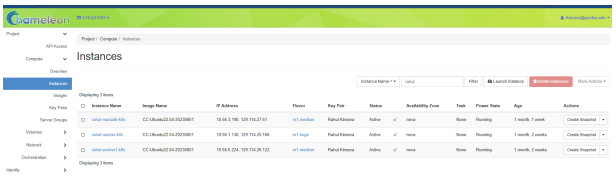

Fig. 5. Experimental Setup for Kubernetes Cluster

As per Fig 3., The chameleon cloud has 3 VMs. m1.large flavor is utilized for the control node, and m1.medium is used for the worker node and database. m1.medium flavor contains 2 VCPUs, 4GB RAM, and 40 GB Disk Storage, whereas m1.large flavor contains 4 VCPUs, 8 GB RAM, and 40 GB Disk Storage. MariaDB was deployed in the database node, which hosts MISP-related data. As part of the experimental setup, nping or simple DdoS attack using tools like Low Ion Cannon Orbiter is conducted onto the Kubernetes cluster. Ideally, the attack is mitigated by threat detection by proactively blocking the packets received from malicious IPs.

Algorithm 1 Import Threat Intelligence from MISP to Zeek

Require: MISP API Endpoint, API Key, Zeek Shared Directory Path

Ensure: Zeek-compatible Threat Intelligence File (`intel.dat`)

Fetch Indicators from MISP:
Set `MISP_API_ENDPOINT` and `API_KEY`.
Execute a `GET` request to retrieve indicators (e.g., IPs, domains).
Parse the JSON response into a list of attributes.
Format Indicators for Zeek:
for each indicator in the attribute list **do**
  if type is `"ip-src"` or `"ip-dst"` **then**
    Set `type = Intel::ADDR`.
  else if type is `"domain"` **then**
    Set `type = Intel::DOMAIN`.
  end if
  Append the formatted indicator to `intel.dat` in TSV format.
end for
Store `intel.dat` in Shared Storage:
Save `intel.dat` to `/shared/zeek/intel.dat`.
Configure Zeek to Read `intel.dat`:
Update `local.zeek` configuration to include `intel.dat`.
Restart Zeek pods to apply the changes.
Monitor Alerts:
Process network traffic through Zeek.
Check `intel.log` for detected matches.

V. Results and Discussion

So far, the proposed architecture setup has been the toughest to achieve as multiple issues have been encountered with containerizing Zeek and MISP. Multiple issues that were discovered during these projects are as follows:

1) Lack of Documentation for this setup: While previously created docker images of Zeek and MISP existed on GitHub, little to no documentation was available that would ensure a smooth conversion of Dockerfiles into stateful Kubernetes pods. This results in many package mismatches and reinstalling software from scratch.
2) Lack of Resources: Worker Node storage was not sufficient to host MISP-based data as a basic requirement for hosting MISP data requires 4 VCPU, 8GB RAM, and 320 GB disk space or more.
3) Network issues: While Zeek was connected to obtain data from the eth0 interface, Zeek could not capture packets due to issues with Chameleon's shared net IP configuration. This results in no output in the persistent volume. Also, MISP works only from shared net IPs but it doesn't function with the associated floating IP as chameleon prevents access apart from SSH on that IP.

Ideally, results would display a working Kubernetes cluster, and we would gather resource utilization using Metrics

API. Based on the metrics, we would observe how resource-intensive this setup is and compare the resource metrics of anomaly detection using ML. We would expect the ML to provide accuracy similar to our Kubernetes setup, but it would be less cost-effective than our proposed solution.

VI. Conclusion

Integration of Zeek and MISP in these clusters of Kubernetes reflects scalability, aside from being proactive in modern cloud threat detection. This framework solves such big challenges caused by dynamic and complex nature presented by Kubernetes-using Zeek on one hand to bring in real-time analysis in network traffic, and MISP, on the other hand, providing a strong sharing platform for threat intelligence. Having deployed these tools in place with Kubernetes ensures that security in the cloud is cost-effective yet adaptive to emerging threat landscapes.

VII. Future Work

There is much more to do to complement this framework, considering that Zeek and MISP will be integrated into the Kubernetes clusters. Of the work to pursue, one recommendation is visual representation with Grafana for intrusion detection results. This would also let organizations have much better visibility into trends, anomalies, and attack patterns from Prometheus as a data source and allow them to make decisions more quickly. Automation of threat intel feed ingestion from MISP via Zeek with the help of Kubernetes CronJobs or CI/CD pipelines keeps the system updated without human intervention. This can be further complemented by the integration of advanced alerting mechanisms and solutions for SIEM correlation across a variety of log sources, enhancing situational awareness and response times.

It may be further extended with more behavioral analysis and threat-hunting capabilities to support Advanced Persistent Threat detection. Further extension of the framework in monitoring multi-cloud or hybrid Kubernetes environments would make it even more applicable to organizations with a distributed infrastructure. Other important directions to be pursued in order to make the framework suitable for large-scale deployments include performance optimization, such as resource tuning and horizontal scaling. The addition of user-friendly interfaces for Zeek and MISP configurations, strategies for long-term data retention for compliance and forensic analysis, will make the system more user-friendly and robust. It could also grow into an all-round solution for contemporary cloud-native security issues when real-world traffic validation of the framework and the integration of more threat intelligence sources are performed.

Acknowledgment

Further work will be required to proceed to for a successful implementation. Will keep you posted on updates

CNIT 623 - Intro to cloud infrastructure final Project Report

References

[1] "Exploring the Cloud Computing Loop in the Strategic Alignment Model," SpringerLink. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-94541-5_12

[2] "Cloud Energy Consumption," Encyclopedia of Cloud Computing, Wiley Online Library. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/9781118821930.ch25

[3] "The Role of Institutional Pressures and Top Management Support in the Intention to Adopt Cloud Computing Solutions," Emerald Insight. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/jeim-09-2014-0087/full/html

[4] "Enhancement of Big Data Security in Cloud Computing Using RSA Algorithm," IEEE Conference Publication, IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/10410830

[5] "CLOUD COMPUTING STRATEGY AND IMPACT IN BANKING/FINANCIAL SERVICES," Computer Science and Information Technology. [Online]. Available: https://aircconline.com/csit/papers/vol11/csit111704.pdf

[6] "e-Health Cloud: Opportunities and Challenges." [Online]. Available: https://www.mdpi.com/1999-5903/4/3/621

[7] "Intrusion Detection System in Cloud Environment: Literature Survey & Future Research Directions," ScienceDirect. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667096822000775

[8] "Intrusion Detection System (IDS) Server Placement Analysis in Cloud Computing," IEEE Conference Publication, IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/7571954

[9] "Is Machine Learning Necessary for Cloud Resource Usage Forecasting?," ACM Digital Library, ACM. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3620678.3624790

[10] "Improving Network Security through Traffic Log Anomaly Detection Using Time Series Analysis," SpringerLink. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-16626-6_14

[11] "Survey Paper: Optimization and Monitoring of Kubernetes Cluster using Various Approaches," SemanticScholar. [Online]. Available: https://pdfs.semanticscholar.org/b80b/d0730cb8510c087ca98b7d45b3bff2d199a3.pdf

[12] "Which Open-Source IDS? Snort, Suricata, or Zeek," ScienceDirect. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1389128622002420

[13] "Refinements in Zeek Intrusion Detection System," IEEE Conference Publication, IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9785047

[14] "Privacy Preserving Cyber Threat Information Sharing and Learning for Cyber Defense," IEEE Conference Publication, IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8666477