Optimising Digital Forensics Investigations in Containers as a Service Environments

Internship
MSc in Cybersecurity

Louise Elie
Student ID: 21195137

School of Computing
National College of Ireland

Supervisor: Kamil Mahajan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Louise Elie		
Student ID:	21195137		
Programme:	MSc in Cybersecurity	**Year:**	2023-2024
Module:	Internship		
Supervisor:	Kamil Mahajan		
Submission Due Date:	September 16th, 2024		
Project Title:	Optimising Digital Forensics Investigations in Containers as a Service Environments		
Word Count:	7250 words	**Page Count**	20 pages

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: [illegible]

Date: September 13th, 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI Acknowledgement Supplement

Internship

Optimising Digital Forensics Investigations in Containers as a Service Environments

AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

Tool Name	Brief Description	Link to tool
SciSpace	Find additional papers related to a research question.	https://typeset.io/
ChatGPT	Chatbot to assist in diverse types of tasks.	https://chatgpt.com/

Description of AI Usage

This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

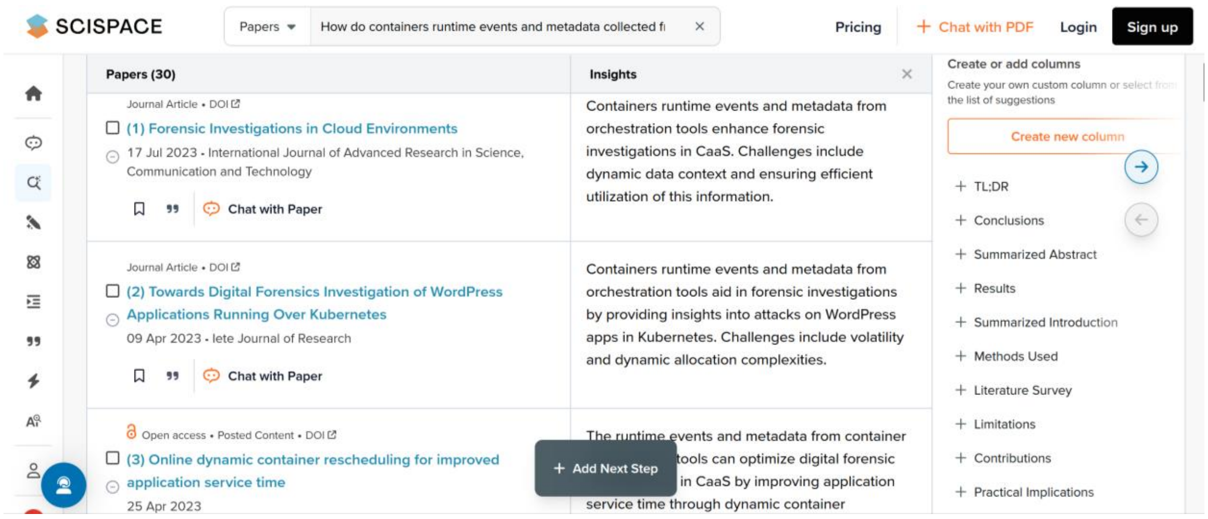
SciSpace	
Due to the gap in the related work, this tool was used to find papers which could propose solutions to a research question.	
Insert the research question**.**	Lists papers and their insights

ChatGPT	
Operational errors occurred during the testing process; this tool helped in troubleshooting.	
Insert the message error**.**	Provides suggestions for solving the problem.

Evidence of AI Usage

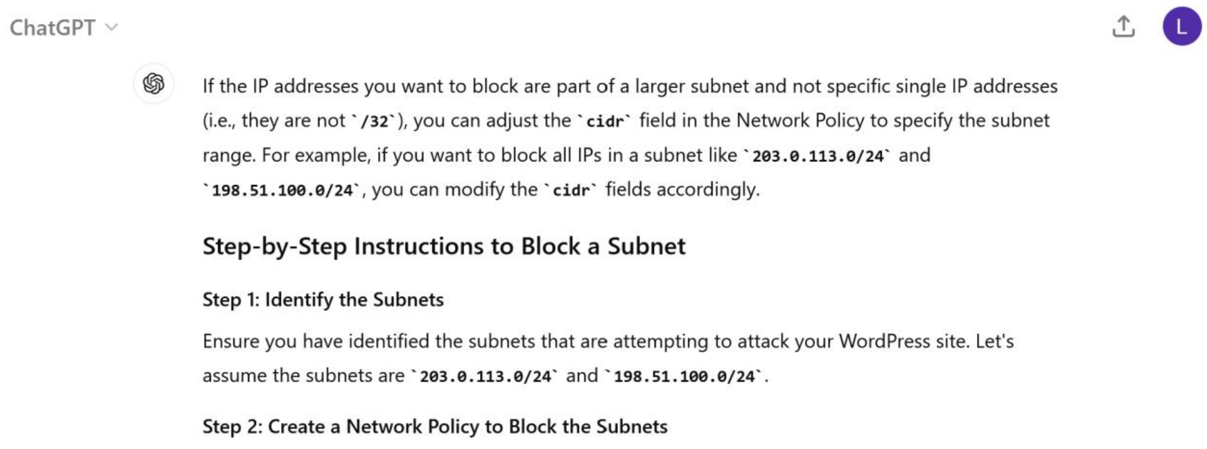
This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

Additional Evidence:



Papers (30)	Insights
Journal Article • DOI (1) Forensic Investigations in Cloud Environments 17 Jul 2023 • International Journal of Advanced Research in Science, Communication and Technology	Containers runtime events and metadata from orchestration tools enhance forensic investigations in CaaS. Challenges include dynamic data context and ensuring efficient utilization of this information.
Journal Article • DOI (2) Towards Digital Forensics Investigation of WordPress Applications Running Over Kubernetes 09 Apr 2023 • Iete Journal of Research	Containers runtime events and metadata from orchestration tools aid in forensic investigations by providing insights into attacks on WordPress apps in Kubernetes. Challenges include volatility and dynamic allocation complexities.
Open access • Posted Content • DOI (3) Online dynamic container rescheduling for improved application service time 25 Apr 2023	The runtime events and metadata from container orchestration tools can optimize digital forensic in CaaS by improving application service time through dynamic container

Additional Evidence:



If the IP addresses you want to block are part of a larger subnet and not specific single IP addresses (i.e., they are not `/32`), you can adjust the `cidr` field in the Network Policy to specify the subnet range. For example, if you want to block all IPs in a subnet like `203.0.113.0/24` and `198.51.100.0/24`, you can modify the `cidr` fields accordingly.

Step-by-Step Instructions to Block a Subnet

Step 1: Identify the Subnets

Ensure you have identified the subnets that are attempting to attack your WordPress site. Let's assume the subnets are `203.0.113.0/24` and `198.51.100.0/24`.

Step 2: Create a Network Policy to Block the Subnets

Optimising Digital Forensics Investigations in Containers as a Service Environments

Louise Elie
Student ID: 21195137

Abstract

Cloud computing is a popular technology used among individuals and companies. One of its models, named Containers as a Services (CaaS), combines the dynamic nature of containers with the advantages of a managed and scalable infrastructure. This paper focuses on the performance of digital forensics in CaaS environments, where traditional forensic procedures must adapt to the challenges posed by the ephemeral nature of containers and the volatile data associated with public cloud environments. With a significant gap in the related work, this study contributes to the field by analysing container runtime events and metadata from Kubernetes, a container orchestration tool, to optimise forensics investigations in public cloud systems. This research proposes a containerised infrastructure hosted on Azure with the CaaS model named Azure Kubernetes Service (AKS) to conduct digital forensics. The proposed architecture exposed a vulnerable WordPress application deployed with AKS. To simulated real-world scenarios, brute force attacks using X Brute Forcer and Denial of Service (DoS) attacks using Apache JMeter were performed on an external Kali Linux machine. The results demonstrated that the data collected from containers runtime events and metadata confirmed evidence of the attacks. The forensic investigations were efficient as rules for threat detection were configurated, along with mitigation solutions.

Keywords **–** digital forensics, containers security, Kubernetes, public cloud

1 Introduction

The past ten years have seen a rapid evolution of cloud computing, becoming a foundational technology for both individuals and organisations. Nowadays, everyone uses services hosted in the cloud, and some companies rely entirely on it (Mosca *et al.*, 2014). Behind this concept of “cloud” computing, infrastructure resources are hosted and managed by cloud service providers like Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP). Many types of cloud models are now available, this paper will focus on Containers as a Service (CaaS). In this architecture, the end-users have access to these lightweight isolated environments designed to run specific software applications. This model offers containers as resources through a container orchestration tool.

Kubernetes is a container orchestration tool that can be used to manage containers. Its role is to automate the management of containerised applications, including deployments, scalability, and change of configurations. Kubernetes guarantees the operation of multiple containers across various hosts. Figure 1 illustrates the different key elements: nodes, which are physical or virtual (e.g., local servers or instances managed on public clouds) machines running the control plane (Kubernetes); pods, which are the smallest units that hosts one or several containers; other resources include services which expose an application running on pods as a network service, and deployments which control the updating and scaling of pods. On cloud environments, the control planes, also known as master nodes, are managed by the cloud service provider. Thus, some data such as audit logs may not be directly accessible.

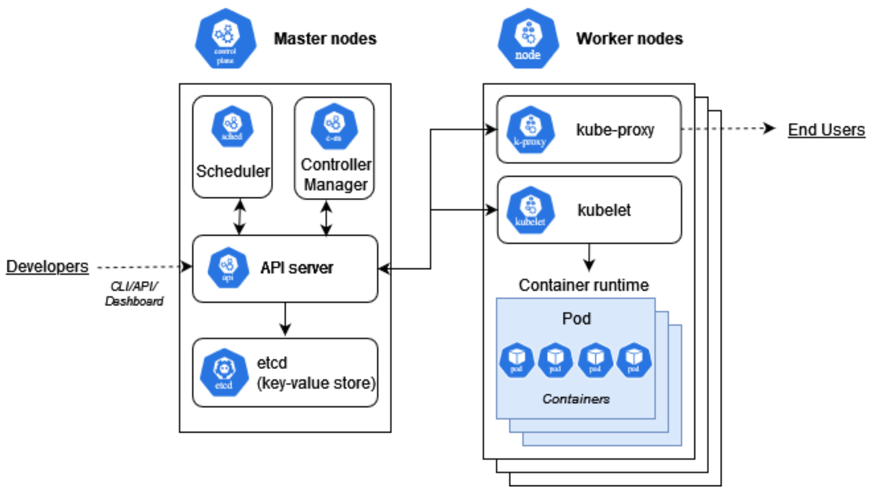


Figure 1: Simplified Kubernetes infrastructure

Any system is a target; thus, containers are vulnerable to cyber-attacks. On managed containerised infrastructure, in public cloud environments, digital forensics is one of the main concerns (Tosatto, Ruiu and Attanasio, 2015). Conducting forensics on cloud computing and containers is more complicated and challenging than doing it on local environment with physical devices. In fact, it can be complex to locate data because of these components' scalable nature (Singh and Chatterjee, 2017). Furthermore, the resources can be scaled up or down in addition to being volatile.

In this context, the challenge is to be able to trace back incidents, as data may be moved, deleted, or even inaccessible without the assistance of cloud service provider (Sultan, Ahmad and Dimitriou, 2019). Evidence of an attack could be found using data related to containers and the orchestration tool, specifically runtime events and metadata. Since no prior research has been done combining digital forensics on these two complex environments: containers and public cloud; in the next section, the literature review demonstrates a gap in the field’s knowledge. This leads to the following research question that guides this study.

How can containers runtime events and metadata collected from a container orchestration tool contribute to optimising the efficiency of digital forensic investigations in Containers as a Service (CaaS) environments, and what are the key challenges in leveraging this data in a dynamic context?

This research question aims to explore whether containers data (e.g., runtime events, metadata) from a container orchestration tool can optimise forensic investigation in a public cloud environment. The research will explore tools and approaches on CaaS environments to facilitate forensic procedures. The term efficiency of the research question is defined by specific metrics related to the attacks performed such as increased CPU and memory or multiple login page access attempts, as well as general forensics metrics like accuracy, detection time, and false positives/negatives. The goal is for organisations and forensic investigators to improve their ability to recognise threats in these dynamic contexts and respond to incidents more quickly. The results can also help law enforcement entities and cloud service providers since these procedures may facilitate compliance with regulatory requirements and improve legal conflicts.

The paper discusses digital forensics challenges in Containers as a Service environments, along with the role of containers runtime events and metadata in section 2 related work. The research methodology presenting the different steps to validate the hypothesis is outlined in section 3. Section 4 describes the lab design, including the architecture diagram and the flow for the proposed technique. The implementation, which explains the resources needed for the testing process to collect and analyse data is detailed in section 5. Section 6 demonstrates the evaluation part of the research with a discussion of the main findings. Finally, the last section presents the conclusion and suggestions for enhancing the research for future work.

2 Related Work

In the past few years, there has been a growing interest in using the advantages of public cloud environments for managing containerised infrastructures. On the cloud market, solutions have been developed such as Azure Kubernetes Service (AKS) by Microsoft Azure and Amazon Elastic Kubernetes Service (EKS) by Amazon Web Services (AWS). Nonetheless, these solutions can add more complexity when recovering from cyber-attacks considering the dynamic nature of containers and the potential inaccessibility of data managed by cloud service providers. To give a better understanding of the role of container runtime events and metadata in public cloud container environments for forensic investigations, this section presents the challenges associated with public clouds and containers and analyses existing forensics research conducted on similar architectures.

2.1 Public Cloud Computing Challenges in Forensics

Cloud computing is an innovative solution contributing to scalable capabilities and ease of use for its users. Several types of services are offered on cloud computing: from standard services such as Infrastructure as a Service (IaaS) to security model like Firewall as a Service (FaaS), or even Containers as a Service (CaaS) – which the main topic of this research. In this context, the control plane of the container orchestration tool along with the instance where it is deployed, are managed by the cloud service provider. Singh and Chatterjee (2017) provide an analysis of cloud computing and its associated security challenges. Digital forensics is one of the main concerns. In traditional forensic procedures, the physical device is investigated to determine the attack's root cause and to trace the actions committed by the attacker. This technique is more challenging in a cloud computing environment because of its dynamic nature. Indeed, data is not permanently stored on physical devices; it can be moved, and its volatile characteristic complicates the forensic process. In a similar paper written by Edington Alex and Kishore (2017) which objective was to propose a forensic framework in this complex cloud environment, additional elements highlighted difficulties related to digital forensics. Both users and investigators have restricted access to their infrastructure. For instance, log data like process logs, network logs, or system logs can be difficult to get without the cooperation of the cloud service provider.

Herman *et al.*, (2020) have defined cloud computing forensic science as "the reconstruction of past cloud computing events through the identification, acquisition, preservation, examination, interpretation, and reporting of potential digital evidence". In their publication, the authors provided technical details on data collection and data analysis. They also cover the legal aspect and gives concerns on the absence of standardised tools, practices, and procedures. Shah and Malik (2014) identified these issues when the cloud was progressively being widely used. They proposed a multi-layer approach to address these challenges. Their methodology included a front end for the API interface and data acquisition technique, a middle layer for logs, and a back end for the presentation of collected evidence. Their case studies on cross-site scripting attacks and external intrusions could also provide insights on how container runtime events and metadata might be effectively leveraged in real-world forensic scenarios.

2.2 Security Concerns on Containerisation

Regarding container security, the study by Sultan, Ahmad and Dimitriou (2019) discusses containers vulnerabilities, threats, and potential solutions. In their open research section, the need for a structured method in digital forensics in this domain is important, but due to its dynamic nature it remains particularly challenging. Indeed, containers are lightweight, portable, and isolated software environments that can be rapidly scaled up or down depending on resource demands, adding to the difficult of conducting efficient forensic investigations. The NIST guide on application containers security written by Souppaya, Morello and Scarfone (2017) highlights the importance to assess vulnerabilities, secure configuration, and add automation to respond to security challenges in containerised environments. Containers, registries, images, and orchestrators are considered as potential targets to cyber-threats.

Moreover, Tosatto, Ruiu and Attanasio (2015) identified challenges in container orchestration systems. These clusters manage containers deployments and monitor their activity. This adds complexity as they must ensure security across a dynamic and distributed infrastructure. While containers provide isolation, there is still a risk of cross-contamination of data between containers, which can complicate forensic investigations. In a rapidly changing environment, ensuring the integrity and availability of data can be difficult as events may be lost or overwritten during scaling operations. Also, some monitoring solutions do not integrate well with these containerised environments, making it difficult to collect runtime events and metadata necessary for forensic analysis. This paper provided insights in container orchestration tools, which is a main element of the research to collect data after an attack for forensic investigations.

2.3 Existing Containers Forensics Analysis

A few papers are specialised on digital forensics applied to containers and container orchestration tools. The paper written by Hyder *et al.* (2023) was not publicly available and a copy was sent following contact with the corresponding author. The authors describe the demand for effective forensic methods to identify attacks on containerised applications. After a dictionary attack on a local Kubernetes infrastructure hosting a WordPress application, the paper investigates techniques to identify security incidents with the use of log monitoring and alerting tools. This study inspired the lab design, further detailed in section 4. Focusing on Kubernetes, Bagheri *et al.* (2023) proposed a non-disruptive proactive attack mitigation strategy. This approach involves rapid data following potential security incidents to optimise the efficiency of forensic investigations. This leads to a quicker analysis and response. Runtime events are leveraged with the use of a large dataset of 231k alerts based on real-world APT attacks.

Another analysis was conducted directly on Docker containers in a conference paper by Franco *et al.* (2023). The paper identified resource utilisation as an efficiency parameter for the investigation of on an attack. During a crypto jacking attack on Docker containers, the authors used honeypots to collect resources and network data for their forensic investigation. The study discovered that resource patterns such as elevated CPU and RAM usage, temperature spikes and abnormal network traffic were a sign of unauthorized crypto mining. Watts *et al.* (2019) also used Docker, and the monitoring tool Prometheus, to optimise data acquisition in the context of digital forensics. The research also collected performance metrics from Prometheus such as memory data and operating events. These parameters contribute to the analyse of container behaviour during forensic investigations. The paper also noted the short lifetime of containers, impacting data collection in these dynamic environments. Gharaibeh *et al.*, (2024) presented a tool designed for the collection and analysis of container checkpoints. By capturing container states, this approach contributes to the integrity of collected evidence. The paper also identified gaps in existing research regarding the collection of digital evidence from container environments, particularly in the context of incident response and forensic analysis.

2.4 Research Niche

Table I compares the different papers mentioned above, in order of appearance, with their strengths and limitations related to the contribution of this research.

Table 1: Literature contribution to the research question

Reference	Ranking[1]	Contribution	Limitations
Singh and Chatterjee (2017)	A	Presentation of challenges in cloud environments for digital forensics.	No further research on the identified open issues.
Edington Alex and Kishore (2017)	Cited by 123	List of various challenges in cloud forensics relevant to CaaS environments, adding complexity to data acquisition and integrity.	Limited discussion on metadata utilisation and no specificity on containers.

[1] https://www.core.edu.au/conference-portal

Herman *et al.* (2020)	Cited by 36	Overview of forensic challenges in cloud computing and highlighting the importance to maintain the integrity of metadata.	Limited to the identification of challenges.
Shah and Malik (2014)	Cited by 27	Proposes a structured architecture for cloud forensics, which can be adapted to CaaS environments.	Lack of methodology details for implementing the forensic process.
Sultan, Ahmad and Dimitrou (2017)	Cited by 270	Guide on containers security, including forensics investigations in the future research section.	Absence of real-world application.
Souppaya, Morello and Scarfone (2017)	Cited by 95	Presents threats affecting containers, along with mitigations.	Scope does not cover cloud environments.
Tosatto, Ruiu and Attanasio (2015)	Cited by 129	Overview of containerisation, its challenges, and discusses how containers allow for rapid scaling and dynamic resource allocation.	Does the implications of runtime events and metadata in forensic investigations
Hyder *et al.* (2023)	C	Similar experiment conducted on a local Kubernetes environment with an exposed WordPress application. Brute force attacks were targeting WordPress.	Forensic investigations were not performed on a cloud environment.
Bagheri *et al.* (2023)	B	Introduces a proactive, non-disruptive attack mitigation strategy that leverages runtime events and alerts.	Does not involve the maintaining of forensic integrity and accuracy in dynamic context.
Franco *et al.* (2023)	B	Insights on identifying performance metrics to indicate a potential sign of a cyber-attack.	Forensic investigations were not performed on a container orchestration tool.
Watts *et al.* (2019)	Cited by 25	Provides evidence that monitoring tools like Prometheus can gather performance data from containers.	Limited scope of metrics, security-specific events could provide insights in the container behaviour.
Gharaibeh *et al.* (2024)	B	Emphasizes the importance of checkpointing container memory and filesystem for forensic purposes.	Limited scope on runtime events and lack of integration with orchestration tools.

Related work has demonstrated that, due to its scalable and evolving nature, performing digital forensics on cloud computing remains challenging. Containers on cloud computing add even more complexity. With the increasing adoption of containerisation technologies, organisations tend to deploy applications in these lightweight and portable environments. Combined with the benefits of cloud computing, it is also possible to use models such as Containers as a Service (CaaS). When performing container forensics on public cloud environments, it is complex to collect and store data as containers are ephemeral and data is volatile. Moreover, a few research papers have been developed on containers forensics. However, no previous work has been done on collecting containers runtime events and metadata from container orchestration tools in a public cloud environment. This research encompasses cloud security, container security and digital forensics. Methodologies and best practices will be detailed in this paper to optimise digital forensics in CaaS models and benefit forensic investigators and cloud service providers.

3 Research Methodology

Following the literature review, a research methodology was defined to propose a solution to validate the hypothesis that containers runtime events and metadata can optimise the efficiency of digital forensics in public cloud infrastructures. As shown in Figure 2, the research methodology consists of five steps namely state of the art, lab design, data collection, data analysis, and evaluation and results.

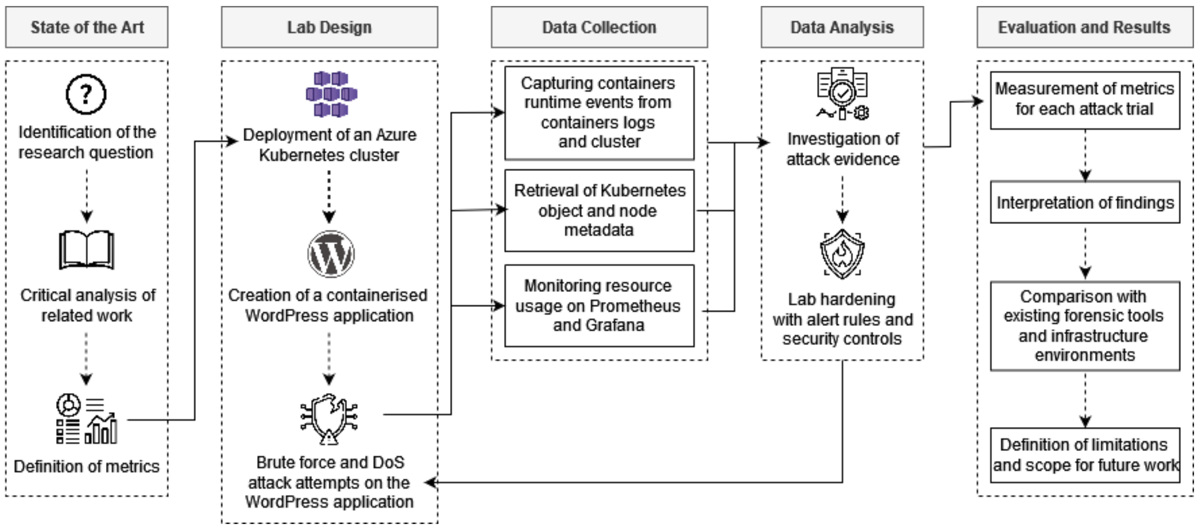

State of the Art
Identification of the research question
Critical analysis of related work
Definition of metrics
Lab Design
Deployment of an Azure Kubernetes cluster
Creation of a containerised WordPress application
Brute force and DoS attack attempts on the WordPress application
Data Collection
Capturing containers runtime events from containers logs and cluster
Retrieval of Kubernetes object and node metadata
Monitoring resource usage on Prometheus and Grafana
Data Analysis
Investigation of attack evidence
Lab hardening with alert rules and security controls
Evaluation and Results
Measurement of metrics for each attack trial
Interpretation of findings
Comparison with existing forensic tools and infrastructure environments
Definition of limitations and scope for future work

Figure 2: Research methodology

The first step, *State of the Art* has been carried out in the previous sections of this paper. The introduction presented the research question. The scalable nature of the public cloud and the ephemeral status of containers represent challenges for digital forensics. To identify related work, the previous section provided a critical analysis of the literature review. No previous work has investigated attack evidence in Containers as a Service (CaaS) environments.

The second step, *Lab Design* involved the creation of the testing environment to collect data from Kubernetes, the container orchestration tool. Azure was the chosen public cloud, on which a Kubernetes cluster was deployed. The vulnerable target was a WordPress site installed and configured on this cluster. To perform digital forensics on this environment, attacks were conducted on the exposed application from an external Kali Linux virtual machine. Open-source tools were used from the attacker machine: X Brute Forcer for brute force attacks and Apache JMeter for DoS attacks. The lab design was inspired by a paper written by Hyder *et al.* (2023). The authors deployed a WordPress site on a local Kubernetes environment to perform digital forensics and find evidence of a brute force attack attempt.

The third step, *Data Collection* gathered activity information following the attacks on the Kubernetes cluster. As stated in the research question, the two primary sources were container runtime events and metadata. To collect this data, pod logs on the terminal and on Azure portal provided insights on container activity, such as login attempts to the *wp-login.php* WordPress file. At the node level, Kubernetes events captured recent observations, like unavailable containers or inactive endpoints, but also a description of object and node metadata. For instance, node metadata displayed the resources allocated to each container. Metrics on resource usage were also visualised in Grafana, which collects logs from Prometheus, a monitoring tool.

The fourth step, *Data Analysis* identified potential attack evidence from the collected data. The testing included three trials each for brute force and DoS attacks, thus six trials in total. After each trial, hardening solutions were also implemented to compare results with rate-limiting, IP address blocking, login attempts restrictions, and alert rules. The data was structured in a table with the following parameters to categorise the attack: timestamp, event type (e.g., login attempt, network activity) and description (e.g., WordPress target file, traffic on port 80), resource utilisation (CPU, memory, network), number of attempts, duration, and metadata details. Another table gathered information from the attack trials based on defined efficiency parameters. These elements are resource utilisation, failed attempts, service availability, and investigation duration. The analysis also involved comparing the two attack types, and the events indicating these attacks.

The fifth step, *Evaluation and Results* measured the metrics identified from the analysed data to provide statistics such as failed login attempts per second for each brute force attack, CPU memory utilisation and service availability during DoS attacks, and packet rates for both types of attacks. Findings from the collected data and implemented hardening solutions were interpreted to demonstrated that data from containers runtime events and metadata can optimise the efficiency of digital forensics. For instance, a spike in resource usage indicated a potential DoS attack, multiple logins attempt in a brief period suggested brute force attacks, and mitigation solutions based on these observations can help anticipate attacks. A theoretical comparison with existing tools such as Falco provided additional information on digital forensics applied to Kubernetes, as well as for other public cloud platforms like AWS and local environments. Limitations from the testing process were identified, along with the definition of scope for future work.

4 Design Specification

A realistic lab environment is essential to conduct forensics investigations in a public cloud-hosted containerised environment (also known as CaaS - Containers as a Service environments). The following subsections identify and present the high-level design for the implemented architecture (in Subsection 4.1) as well as the associated requirements for collecting and analysing evidence following the attacks (in Subsection 4.2).

4.1 System Architecture

Different components are required to set up the lab design: a public cloud infrastructure, a container orchestration tool, a container-based web application, log monitoring and alerting, and open-source tools to simulate attacks on the web application. Microsoft Azure was the provided public cloud, it has a large toolset adapted to this research. Among its services, Azure Kubernetes Services (AKS) has the role of the managed container orchestration tool. Figure 3 presents the architecture diagram deployed on Azure.

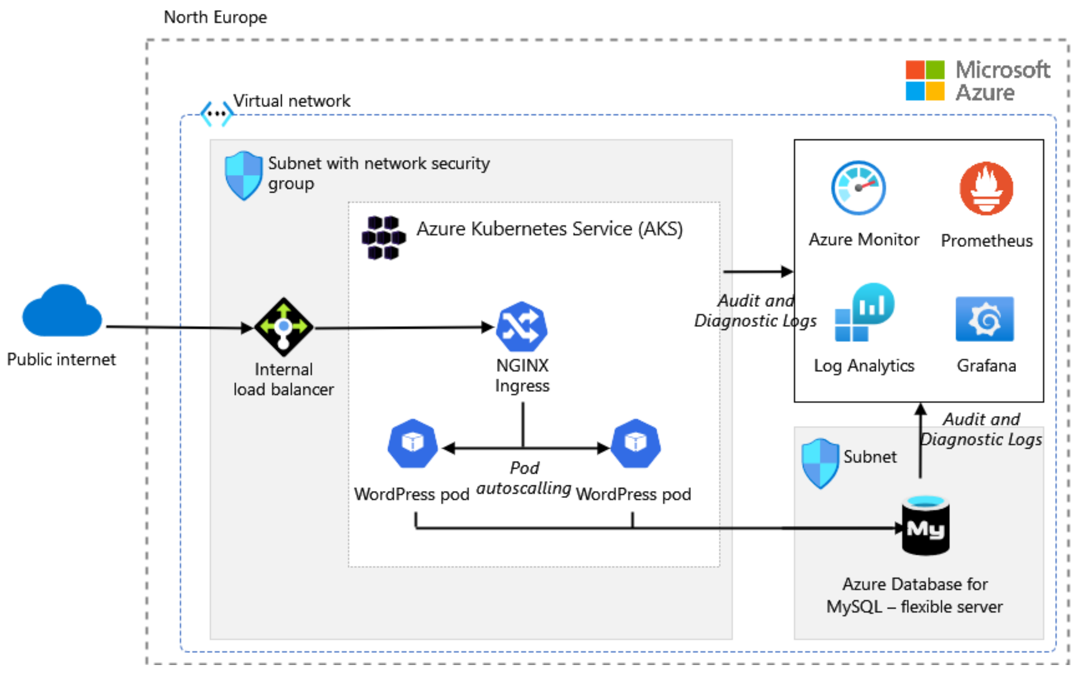


Figure 3: Architecture diagram

On this diagram, an AKS cluster with Kubernetes v1.28.9 was deployed in a private subnet. A NGINX ingress controlled the traffic with the external access from the internet, through an internal load balancer, and the internal resources. Widely used as a solution for the testing process for its simple configuration, WordPress v6.5.3 was implemented on autoscaling AKS pods to provide high availability in a containerised environment. Another subnet hosted the managed a MySQL v8.0.21 database to store WordPress media. The aim is to conduct digital forensics on Containers as a Service environments; thus, the chosen scenario is to target an exposed containerised WordPress application. This infrastructure was analysed with Azure managed tools such as Azure Monitor, Log Analytics, Prometheus, and Grafana. These tools are further detailed in the following subsection.

4.2 Flow of the Proposed Technique

The brute force and the Denial of Service (DoS) attacks are used as an attack model as part of the experiment. These attacks were performed from a local Kali Linux 2024.2 virtual machine which is not in the Azure environment. Figure 4 illustrates the overall workflow of the proposed technique for the attack sequence and data collection. Before launching attacks, it is important to underlie that Azure has a shared responsibility model[2]. The public cloud provider is not responsible of its clients' architecture. In the context of penetration testing, activities on resources such as containers are permitted services[3].

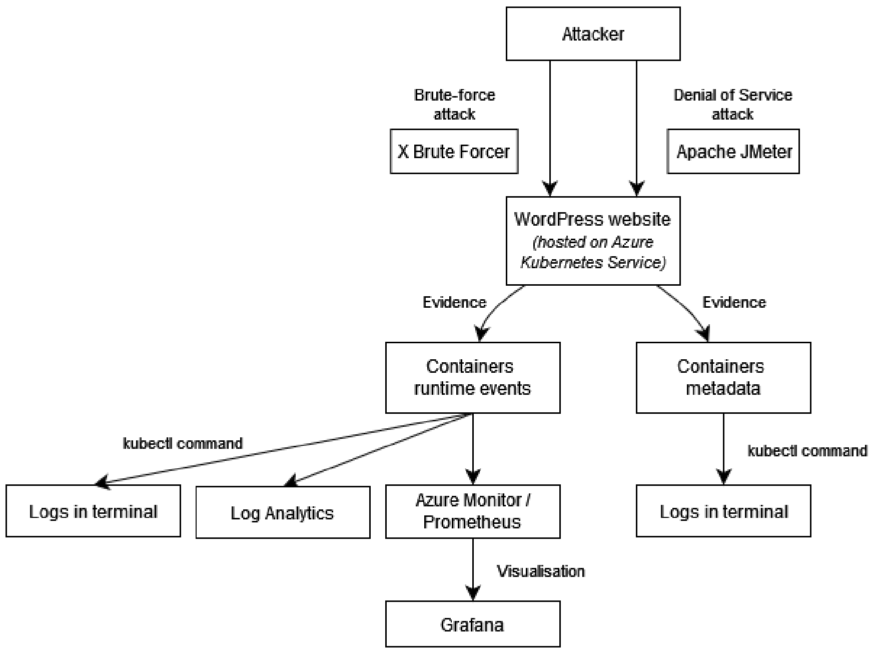


Figure 4: Attack sequence and information gathering

The attacks were launched using open-source tools: X Brute Forcer v1.2 for brute force attacks, and Apache JMeter v5.6.3 for DoS attacks. Both targeted the WordPress login page hosted on the AKS pods. To analyse the data for digital forensics, containers metadata was collected from Kubernetes objects descriptions using Azure Cloud Shell. Containers runtime events were gathered from logs in Azure Cloud Shell, Log Analytics in the Azure portal, and Grafana with metrics sourced from Prometheus.

5 Implementation

For this research, the implementation of the proposed solution was carried out using Azure's managed services. Indeed, to demonstrate the efficiency of containers runtime events and metadata in forensic investigations, a vulnerable web application was deployed on a container orchestration tool via a cloud service provider. Table 2 provides the list of the main tools and technologies used, based on the previously built architecture and attack flow.

Once the cluster was deployed with the vulnerable WordPress application, the attacks were launched from the Kali Linux machine. The first challenge was ensuring that the cloud service provider would not block the attacks. To confirm this, the web application was

[2] https://learn.microsoft.com/en-us/azure/security/fundamentals/shared-responsibility

[3] https://www.microsoft.com/en-us/msrc/pentest-rules-of-engagement

deployed without any security hardenings. As mentioned in the research methodology, each attack was performed in three trials to compare the results. With each trial, security hardenings were progressively added based on the observations.

Table 2: Overview of tools and technologies participating in the testing process

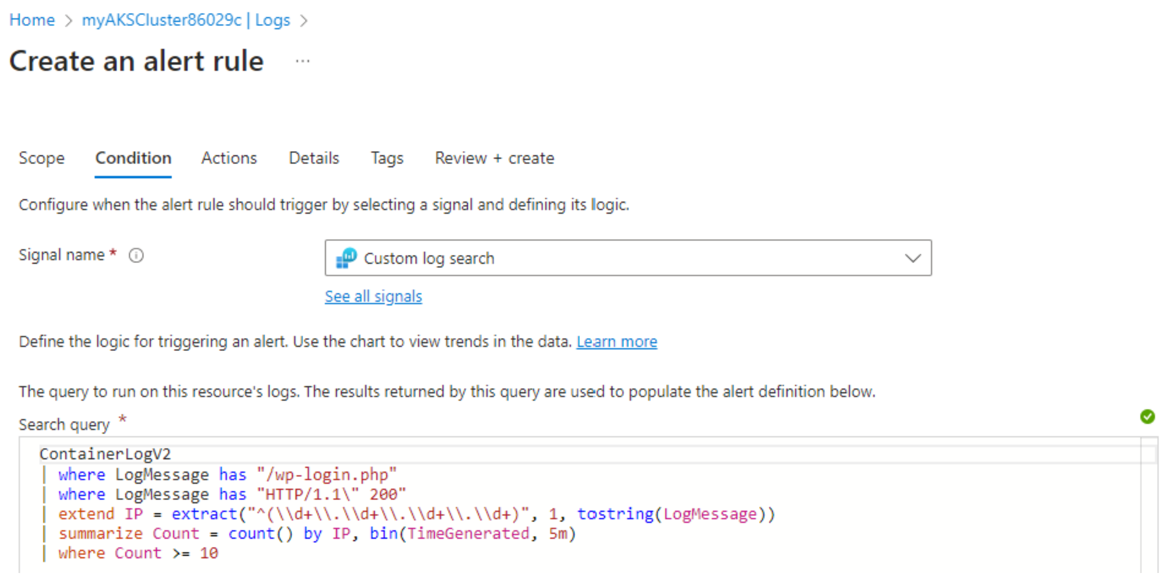
Tools and Technologies	Role
Azure Cloud Shell	Terminal used to deploy, configure, and manage the Kubernetes cluster in the Azure public cloud environment.
Bash	Scripting language opted in Azure Cloud Shell.
Kubernetes (AKS - Azure Kubernetes Service)	Kubernetes v1.28.9 deployed via AKS in Azure Cloud Shell. It has the role of the container orchestration tool.
kubectl	Command-line tool for interacting with the Kubernetes objects and events in the cluster.
Helm	Packet manager used to deploy an ingress controller and WordPress.
YAML	Language for Kubernetes manifests, describing object metadata and runtime events. It also manages resources such as network policies and rate-limiting configurations.
WordPress	Vulnerable web application hosted on the AKS cluster and accessible via internet.
MySQL (Azure Database for MySQL)	Database storing WordPress media content.
X Brute Forcer and Apache JMeter	Open-source tools installed on a local Kali Linux machine to perform brute force and Denial of Service (DoS) attacks, respectively.
Prometheus and Grafana	Monitoring and visualisation tools for gathering information on resource utilisation.
Azure Log Analytics	Azure tool used to create rules based on log queries from data collected by Azure Monitor.
Microsoft Excel	Spreadsheet software to analyse the collected data and create visualisations.

Brute force attacks require a list of passwords to attempt against targeted users. As part of WordPress vulnerability assessment and detailed in the paper written by Kyaw, Sioquim and Joseph (2015), the following URL was fetched to list the existing users who had published on the WordPress application: *<wordpress_ip_addess>/wp-json/wp/v2/users*. To ensure a successful match with one of the users' passwords, a password from a list of the 500 worst passwords[4] was added for the creation of a new user. X Brute Forcer was then launched from the attacker machine, targeting the WordPress IP address with this list of weak passwords.

Based on the elements gathered from the AKS cluster, and further detailed in the evaluation section, the WordPress add-ons WPS Limit Login was implemented to block

[4] https://www.skullsecurity.org/wiki/Passwords

access the login page when multiple login attempts occur in a brief period of time. As shown in Figure 5, an alert rule was also configured to detect repeated patterns in container runtime events, indicating a potential brute force attack.


Home > myAKSCluster86029c | Logs >

Create an alert rule ···

Scope Condition Actions Details Tags Review + create

Configure when the alert rule should trigger by selecting a signal and defining its logic.

Signal name * : Custom log search

See all signals

Define the logic for triggering an alert. Use the chart to view trends in the data. Learn more

The query to run on this resource's logs. The results returned by this query are used to populate the alert definition below.

Search query *

```
ContainerLogV2
| where LogMessage has "/wp-login.php"
| where LogMessage has "HTTP/1.1\" 200"
| extend IP = extract("^(\\d+\\.\\d+\\.\\d+\\.\\d+)", 1, tostring(LogMessage))
| summarize Count = count() by IP, bin(TimeGenerated, 5m)
| where Count >= 10
```

Figure 5: Login attempt alert rule

Regarding the DoS attacks, Apache JMeter was used for load testing. After specifying the WordPress IP address, the target file (e.g., *wp-login.php*), and a thread group of 3000 users, the attack was executed over an average duration of three minutes, during which the site became unavailable due to resource overload.

Since tools like Azure Web Application Firewall (WAF) were not provided in this research environment to mitigate DoS attacks, rate-limiting, IP blocking, and alerts were configured based on data collected from the AKS cluster. Unlike the customed alerts created for brute force attacks, pre-designed Prometheus alerts were used, as rules related to the pod resource usage were included. Figures 6 and 7 display the configurations on rate-limiting and IP blocking added to harden the AKS cluster.

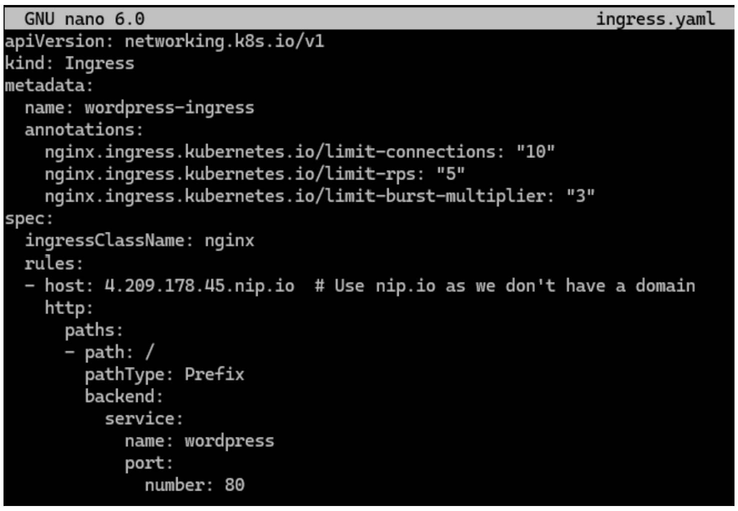
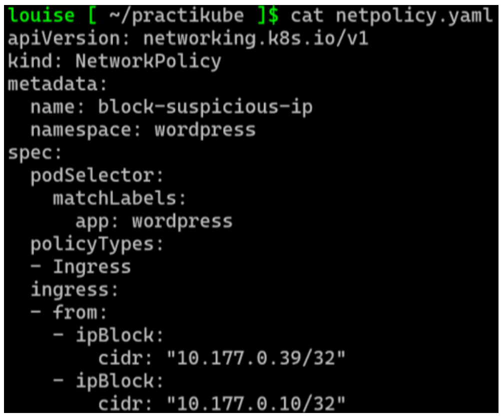

```
  GNU nano 6.0                                             ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: wordpress-ingress
  annotations:
    nginx.ingress.kubernetes.io/limit-connections: "10"
    nginx.ingress.kubernetes.io/limit-rps: "5"
    nginx.ingress.kubernetes.io/limit-burst-multiplier: "3"
spec:
  ingressClassName: nginx
  rules:
  - host: 4.209.178.45.nip.io  # Use nip.io as we don't have a domain
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: wordpress
            port:
              number: 80
```

Figure 7: Rate-limiting configuration



```
louise [ ~/practikube ]$ cat netpolicy.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: block-suspicious-ip
  namespace: wordpress
spec:
  podSelector:
    matchLabels:
      app: wordpress
  policyTypes:
  - Ingress
  ingress:
  - from:
    - ipBlock:
        cidr: "10.177.0.39/32"
    - ipBlock:
        cidr: "10.177.0.10/32"
```

Figure 8: IP blocking rule

6 Evaluation

The aim of this experiment was to answer the research question of whether container runtime events and metadata can contribute to optimising the efficiency of digital forensics on a Containers as a Service (CaaS) environment. To achieve this, brute force, and Denial of Service (DoS) attacks were performed on an exposed containerised application hosted in a public cloud environment. Considering the ephemeral status of containers and the limitations of a lab environment managed by a cloud service provider, the objective of this research was to identify whether traces of evidence could remain and be accessible after a cyberattack. Metrics were defined to identify the efficiency parameters; they reflect how forensic investigations can be conducted using containers runtime events and metadata.

6.1 Pertinence of Container Runtime Events in Attack Detection

As part of the forensics investigations, it is essential to identify elements that can determine the cause and the impact of the attack. As mentioned in the design specification section, container runtime events were collected to assess whether the information could contribute to the forensic investigations. By providing real-time information about the state and behaviour of containers, this data is essential for identifying anomalies and understanding the sequence of events leading to an incident. To provide a comparative analysis, brute force, and Denial of Service (DoS) attacks were conducted, each with three trials and progressively implemented security hardenings. Runtime events were collected from Kubernetes events with information on the state changes and decisions made by the control plane (e.g., pod creation, deletion, and errors), the cluster terminal for logs from applications inside containers, and Grafana for performance metrics.

Combined with metrics defining the efficiency parameters, it was observed that specific container runtime events and metadata, such as multiple attempts to log in and spike in resource utilisation, are indicators of these cyber-attacks. Moreover, the analysis involving security measures on the second and third trials demonstrated that real-time monitoring of container runtime events reduced the time required to detect and respond to cyber-attacks. Table 3 presents the combined collected data from the container runtime events following both attacks, highlighting relevant pattern associated with these threats.

Table 3: Collected data on efficiency parameters

Metrics	Brute force attack	DoS attack
Total attempts	1500	9000
Avg. CPU	68%	86%
Avg. Memory	215 MB	603 MB
Avg. Packet Rate	407 p/s	510 p/s
Avg. Attack Duration	90s	360s
Avg. Failed attempts	99,9% (1 success in trial 1)	When the pod is restarting
Avg. Service availability	Login page blocked from 10 to 20min for trials 2 and 3	2 restarts of pod during the attack
Avg. Investigation Duration	Alert detection in 5min	Alert detection in 5min
Common events	Login attempts	Network activity

As shown in Figure 9, it was observed on the container logs a repetitive attempt access to the WordPress login page (e.g., *wp-login.php*) in a brief period of time. On the second trial, a security add-on was implemented along with an alert rule. The login page was not accessible after a certain number of attempts. This result highlights a decrease in the login attempts per second across trials indicating that the attack is less effective.

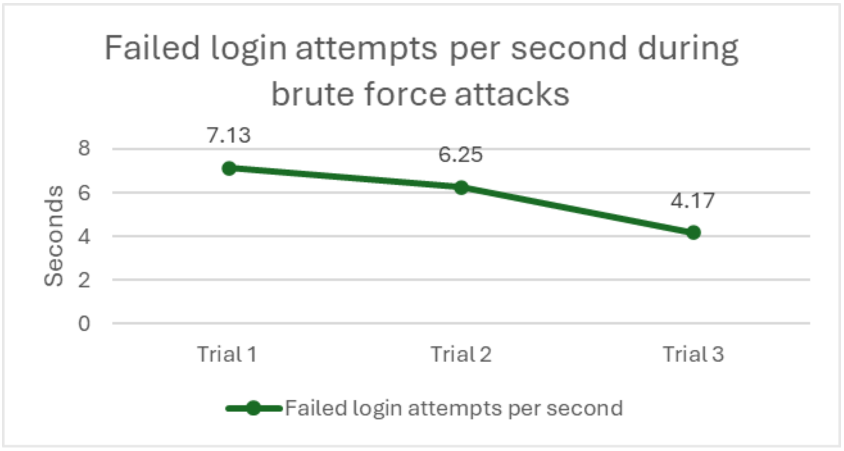


Figure 9: Failed login attempts per second on brute force attacks

Following DoS attacks, CPU usage for the three trials were above 80% with a spike at 90%. Memory usage was also elevated with a minimum of 500MB and a spike at 800 MB for the second trial. Important levels of resource utilisation can indicate the impact on the system, and consequently the severity of the attack. The second trial indicated spikes in CPU and memory usage. It can suggest an effective attack mitigation, such as rate limiting or request filtering. However, the third trial showed comparable results to the first trial as presented in Figure 10. The current resource allocation in the node was not sufficient for the pods managing the WordPress site. Node metadata displays this type of information. Indeed, CPU throttling, and memory could be scaled up to these pods in the node to ensure the availability of the web application. However, on a regular basis these pods do not require all the allocated resources. Consistent pod restarts across trials suggested a strained recovery mechanism. Pods were restarted to handle high traffic, due to the prolonged DoS attacks. Additionally, a high packet rate was observed and coincided with the performance degradation indicating the attack's impact on the system.

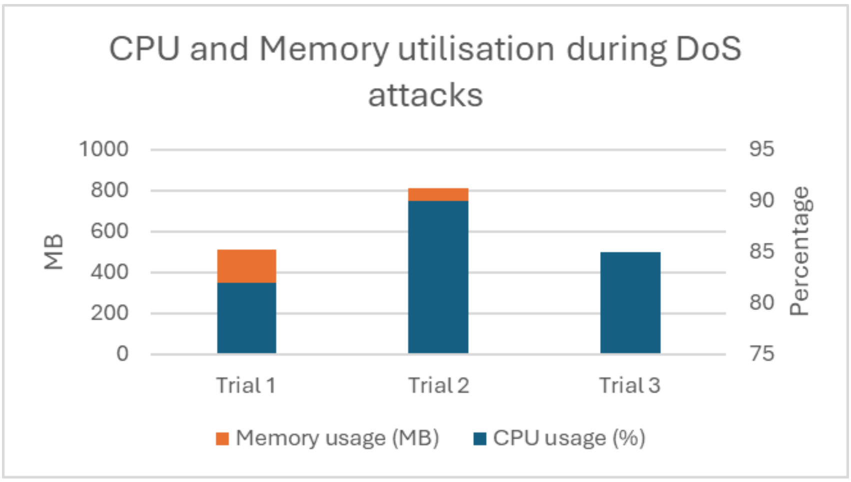


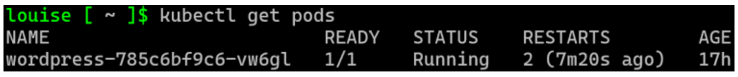
Figure 10: Resource usage during DoS attacks

It is interesting to mention that the number of attempts for both attacks were different: 500 for brute force attacks, and 3000 for DoS attacks. The same number of attempts could have shown comparable results. Once the alerts were configured, the detection time based on container runtime events and metadata was checked at an interval of 5 minutes. This duration time can be reduced, but it will also increase cloud costs. Indeed, shorter investigation times provide efficient detection and response mechanisms.

6.2 Impact of Container Metadata on Forensic Analysis

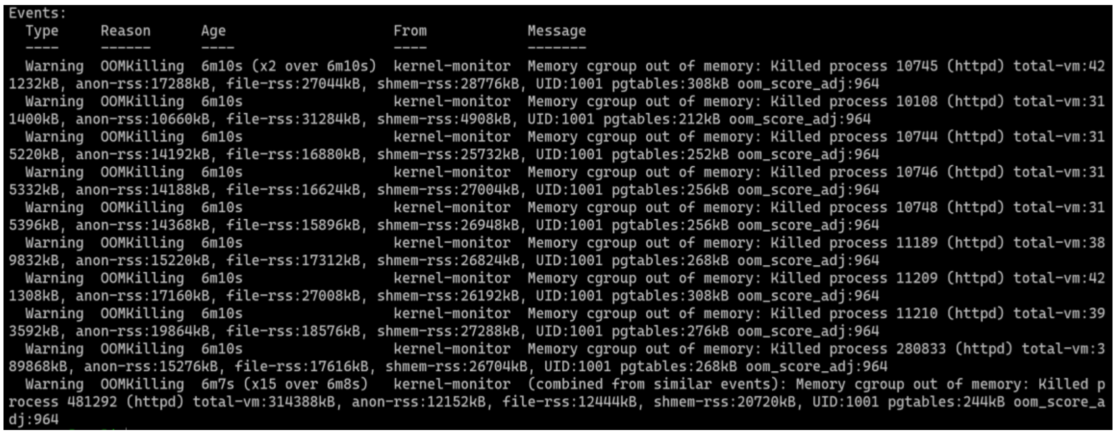
To understand the context of runtime events, metadata is an essential component. It provides static and dynamic information about the containers, the configuration, and the state of the container orchestration environment. Data was collected from Kubernetes resources (e.g., pods, deployments, services). Labels, annotations, resource specifications, but also status field are specified in this type of data. As mentioned in the previous subsection, node metadata also contributes to the forensic analysis. Elements such as node capacity and current resource usage can provide insights about the state of the cluster.

For instance, Figures 11 and 12 displays metadata related to pods and nodes. Following a DoS attack, it revealed that killed processes led to the to the restart of pods, highlighting the impact of the attack on the system. By identifying additional information on the container orchestration tool activity, metadata improves the accuracy of forensic investigations, leading to a better detection.



```
louise [ ~ ]$ kubectl get pods
NAME                         READY   STATUS    RESTARTS        AGE
wordpress-785c6bf9c6-vw6gl   1/1     Running   2 (7m20s ago)   17h
```

Figure 11: Pods metadata after a DoS attack



```
Events:
  Type     Reason      Age                      From             Message
  ----     ------      ----                     ----             -------
  Warning  OOMKilling  6m10s (x2 over 6m10s)   kernel-monitor  Memory cgroup out of memory: Killed process 10745 (httpd) total-vm:42
1232kB, anon-rss:17288kB, file-rss:27044kB, shmem-rss:28776kB, UID:1001 pgtables:308kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 10108 (httpd) total-vm:31
1400kB, anon-rss:10660kB, file-rss:31284kB, shmem-rss:4908kB, UID:1001 pgtables:212kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 10744 (httpd) total-vm:31
5220kB, anon-rss:14192kB, file-rss:16880kB, shmem-rss:25732kB, UID:1001 pgtables:252kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 10746 (httpd) total-vm:31
5332kB, anon-rss:14188kB, file-rss:16624kB, shmem-rss:27004kB, UID:1001 pgtables:256kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 10748 (httpd) total-vm:31
5396kB, anon-rss:14368kB, file-rss:15896kB, shmem-rss:26948kB, UID:1001 pgtables:256kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 11189 (httpd) total-vm:38
9832kB, anon-rss:15220kB, file-rss:17312kB, shmem-rss:26824kB, UID:1001 pgtables:268kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 11209 (httpd) total-vm:42
1308kB, anon-rss:17160kB, file-rss:27008kB, shmem-rss:26192kB, UID:1001 pgtables:308kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 11210 (httpd) total-vm:39
3592kB, anon-rss:19864kB, file-rss:18576kB, shmem-rss:27288kB, UID:1001 pgtables:276kB oom_score_adj:964
  Warning  OOMKilling  6m10s                    kernel-monitor  Memory cgroup out of memory: Killed process 280833 (httpd) total-vm:3
89868kB, anon-rss:15276kB, file-rss:17616kB, shmem-rss:26704kB, UID:1001 pgtables:268kB oom_score_adj:964
  Warning  OOMKilling  6m7s (x15 over 6m8s)    kernel-monitor  (combined from similar events): Memory cgroup out of memory: Killed p
rocess 481292 (httpd) total-vm:314388kB, anon-rss:12152kB, file-rss:12444kB, shmem-rss:20720kB, UID:1001 pgtables:244kB oom_score_a
dj:964
```

Figure 12: Node events on resource usage after a DoS attack

6.3 Comparison with Existing Tools

Microsoft Azure is not the only cloud service provider offering Containers as a Service (CaaS) environments. Amazon Elastic Kubernetes Service (Amazon EKS) from Amazon Web Services (AWS) is also widely used. Unlike Azure, AWS has a forensic tool named Amazon Automated Forensics Orchestrator for Amazon EC2. This solution provides a workflow to automate forensic processes and gathering collected data. Based on theoretical approach, it can capture data from an instance, and in the context of this research, it would be the instance where the container orchestration tool is deployed. As described in EKS documentation[5], this forensic tool can capture the container runtime events for the operating system memory. However, it is noted on the product page[6] that this tool is not suitable for every use cases.

Open-source runtime security tools like Falco is regularly mentioned to conduct forensics investigation on Kubernetes environments. This agnostic solution monitors container activity and can suspect suspicious behaviour using security rules. It is employed to analyse evidence, determine the root cause, and understand attacker's actions. As described by Bisson (2024), Falco can be installed on Azure and can be integrated with Azure Security Information and Event Management (SIEM) such as Sentinel and monitoring tools like Azure Monitor. According to the technical documentation written by Shankar (2021), security rules can be optimised to be more context-aware with container runtime events and metadata, reducing false positives and improving the accuracy of attack detection. For instance, knowing the expected resource usage of specific containers can help in detecting a potential Denial of Service (DoS) attack. In a blog posted on Sysdig website, the creator of Falco,

[5] https://aws.github.io/aws-eks-best-practices/security/docs/incidents/

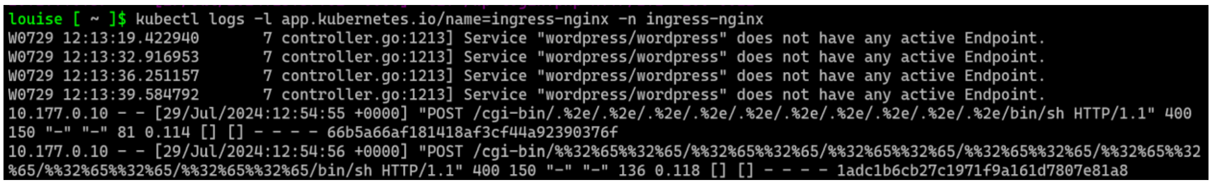
[6] https://docs.aws.amazon.com/solutions/latest/automated-forensics-orchestrator-for-amazon-ec2/welcome.html

Douglas (2022) highlights that container runtime events such as unwanted IP address can be used in Falco rules to mitigate DoS attacks. On a scientific approach, Bagheri *et al.* (2023) used Falco in the Kubernetes cluster to create a dataset of threat alerts as part of a non-disruptive proactive mitigation approach to cyber-attacks.

6.4 Challenges in Leveraging Containers Data in Forensics

When using container runtime events and metadata for forensics investigations in this experiment, some challenges were encountered. The multiple data sources and the large volume of information made it difficult to effectively trace the attack in a short response time. A tool to gather and analyse this data would have been beneficial. Unlike AWS, Azure do not have a dedicated tool for forensics investigations. While Azure Sentinel and Azure Monitor Investigator could have provided similar functionality, they were not accessible in this research environment. Falco could have been a potential solution, but it required additional time, learning, and practical experience on this open-source tool.

Some errors occurred during the testing process. Among them, the Prometheus alerts were not triggered during the DoS attack. While Grafana dashboards showed high CPU usage, node metadata indicated low CPU usage for the WordPress pods. It was due to the overall resources allocated to the node rather than the individual pods. Kubernetes is a complex infrastructure, even more when it is partially managed by the service cloud provider. Despite configuring the rate limiting on the ingress controller, as shown in Figure 13, the logs indicated that the WordPress service still went down.



```
louise [ ~ ]$ kubectl logs -l app.kubernetes.io/name=ingress-nginx -n ingress-nginx
W0729 12:13:19.422940       7 controller.go:1213] Service "wordpress/wordpress" does not have any active Endpoint.
W0729 12:13:32.916953       7 controller.go:1213] Service "wordpress/wordpress" does not have any active Endpoint.
W0729 12:13:36.251157       7 controller.go:1213] Service "wordpress/wordpress" does not have any active Endpoint.
W0729 12:13:39.584792       7 controller.go:1213] Service "wordpress/wordpress" does not have any active Endpoint.
10.177.0.10 - - [29/Jul/2024:12:54:55 +0000] "POST /cgi-bin/.%2e/.%2e/.%2e/.%2e/.%2e/.%2e/.%2e/.%2e/.%2e/.%2e/bin/sh HTTP/1.1" 400
150 "-" "-" 81 0.114 [] [] - - - - 66b5a66af181418af3cf44a92390376f
10.177.0.10 - - [29/Jul/2024:12:54:56 +0000] "POST /cgi-bin/%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32%65/%%32%65%%32
%65/%%32%65%%32%65/%%32%65%%32%65/bin/sh HTTP/1.1" 400 150 "-" "-" 136 0.118 [] [] - - - - 1adc1b6cb27c1971f9a161d7807e81a8
```

Figure 13: Ingress controller logs

Initially, accuracy was one of the efficiency parameters. However, due to the public cloud environment and its associated costs, the lab was only launched during the attack trials. Thus, it would have not been relevant to measure only these data collected during and after an attack, particularly for false positives/negatives.

6.5 Discussion

During brute force and DoS attacks, the forensic investigation identified key elements from the container runtime events and metadata highlighting attack evidence. Information such as the victim pod, the target file, or the attacker IP address were traced. Given the dynamic nature of containers and the managed infrastructure hosted on a public cloud, the attacks were successful, and data was collected both live on Azure and offline from the cluster terminal. This reverse engineering process went back to the event of the attacks, who committed it, when it occurred, whether it was successful, what access the attacker had, and what actions they took. Patterns from the container runtime events and metadata was determined: a high number of file access and multiple logins attempts during brute force attacks; spike in resource utilisation, increased packet rates and service unavailability during DoS attacks.

Three trials were conducted for each attack. However, this was insufficient to evaluate the accuracy of forensic investigations. Additionally, the web application remained vulnerable and implementing further security measures, such as the hardening of WordPress and adding an HTTPS certificate, would have been more pertinent for the analysis. The integration of a security tool like Falco could have improved the forensic investigations.

This research did not involve participants informed consent and use of dataset. As the context environment is a public cloud, the lab design was compliant with GDPR by ensuring the data was stored in the European Union. In alignment with the research objectives, this work contributed to the goal 9[7] of the United Nations Sustainability Goals. Indeed, this research aims for the development of a resilient infrastructure, prioritising on innovation. Containers as a Service (CaaS) environments represent a minimalist approach of infrastructure for hosting and managing applications. By optimising digital forensic investigations on public cloud-based containerised environments, the research improves digital infrastructure. While there has been research on local Kubernetes environments, no studies on digital forensics in CaaS environments have been published yet.

7 Conclusion and Future Work

The aim of this research was to validate whether container runtime events and metadata hosted on a public cloud environment can optimise digital forensics and to identify challenges involved in leveraging this type of data. This research proposes a lab setup of a containerised WordPress application hosted on Azure Kubernetes Service (AKS). Two types of attacks, namely brute force and Denial of Service (DoS), were performed on this exposed application with three trials each. Data was collected from container orchestration tool events and objects, WordPress container logs, and Grafana for performance metrics.

Results demonstrate that specific patterns from containers runtime events and metadata, such as repetitive login page access attempts in a brief period, indicate potential brute force attacks, and traces of IP addresses, a spike in resource utilisation captured the node metadata and Grafana, and pods restart leading to service unavailability are signs of DoS attacks. Metadata provides a better understanding of containers runtime events. These two types of information improve the detection response with appropriate security hardenings and the configuration of alert rules.

A comparison with existing tools revealed that forensics tools are available in other Containers as a Service (CaaS) environments such as Amazon Automated Forensics Orchestrator, and open-source and agnostic tools like Falco. However, a limitation of this study is the complexity of measure the accuracy to truly capture diverse real-world application. The absence of a forensic tool aggregates data from the various sources would have optimise the handling of a large volumes of information.

[7] https://sdgs.un.org/goals

This research can enhance digital forensics on containers, especially in public cloud environments. This work can be improved by complexifying the attacks. For instance, to differentiate brute force from the DoS attacks when there are launched simultaneously, to add other types of cyber-attacks, or to perform an attack from the target pod. Another target in the container orchestration tool could also provide additional insights in container forensics. In terms of a solution for any containerised environment (e.g., public, private, multi-cloud), more research must be carried out with the use of Falco.

References

Bagheri, S. *et al.* (2023) ‘Warping the Defence Timeline: Non-Disruptive Proactive Attack Mitigation for Kubernetes , in *ICC 2023 - IEEE International Conference on Communications*. Rome, Italy, 28 May - 01 June 2023, pp. 777–782, IEEE Xplore. Available at: https://doi.org/10.1109/ICC45041.2023.10278632 [Accessed 03 August 2024].

Bisson, S. (2024) ‘Securing Azure Kubernetes with Falco’, *InfoWorld*, 15 May. Available at: https://www.infoworld.com/article/2336407/securing-azure-kubernetes-with-falco.html [Accessed 27 August 2024].

Douglas, N. (2022) ‘Preventing DoS Kubernetes using Falco and Calico’, *Sysdig*, 10 October. Available at: https://sysdig.com/blog/denial-of-service-kubernetes-calico-falco/ [Accessed: 03 August 2024].

Edington Alex, M. and Kishore, R. (2017) ‘Forensics framework for cloud computing’,*Computers & Electrical Engineering*, 60, pp. 193–205, Elsevier. Available at: https://doi.org/10.1016/j.compeleceng.2017.02.006 [Accessed 07 August 2024].

Franco, J. *et al.* (2023) ‘Forensic Analysis of Cryptojacking in Host-Based Docker Containers Using Honeypots’, in *ICC 2023 - IEEE International Conference on Communications*. Rome, Italy, 28 May - 01 June 2023, pp. 4860–4865, IEEE Xplore. Available at: https://doi.org/10.1109/ICC45041.2023.10278764 [Accessed 13 February 2024].

Gharaibeh, T. *et al.* (2024) ‘Don’t, Stop, Drop, Pause: Forensics of CONtainer CheckPOINTs (ConPoint)’, in *The 19th International Conference on Availability,Reliability and Security (ARES 2024)*. Vienna, Austria, 30 July - 02 August 2024, pp. 1–11, ACM. Available at: https://doi.org/10.1145/3664476.3670895 [Accessed 26 August 2024].

Herman, M. *et al.* (2020) *NIST cloud computing forensic science challenges*. NIST IR8006. Gaithersburg, MD, USA: National Institute of Standards and Technology. Available at: https://doi.org/10.6028/NIST.IR.8006 [Accessed 16 February 2024].

Hyder, M.F. *et al.* (2023) ‘Towards Digital Forensics Investigation of WordPress Applications Running Over Kubernetes’, *IETE Journal of Research*, 70(4), pp. 3856–3871, Taylor and Francis Online. Available at:https://doi.org/10.1080/03772063.2023.2195837 [Accessed 13 February 2024].

Kyaw, A.K., Sioquim, F. and Joseph, J. (2015) ‘Dictionary attack on Wordpress: Security and forensic in. *2015 Second International Conference onInformation Security and Cyber*

Forensics (InfoSec), Cape Town, South Africa, 15-17November 2015, pp. 158–164, IEEE Xplore. Available at: https://doi.org/10.1109/InfoSec.2015.7435522 [Accessed 06 June 2024.

Mosca, P. *et al.* (2014) ‘Cloud Security: Services, Risks, and a Case Study on Amazon Cloud Services’, *International Journal of Communications, Network and SystemSciences*, 07(12), pp. 529–535. ResearchGate. Available at: https://doi.org/10.4236/ijcns.2014.712053 [Accessed 27 August 2024].

Shah, J.J. and Malik, L.G. (2014) ‘An approach towards digital forensic framework forcloud’, in *2014 IEEE International Advance Computing Conference (IACC)*, Gurgaon, India, 21-22 February 2014, pp. 798–801, IEEE Xplore. Available at: https://doi.org/10.1109/IAdCC.2014.6779425 [Accessed 26 August 2024].

Shankar, P. (2021) ‘Getting started with Kubernetes audit logs and Falco’, *Sysdig*, 9 February. Available at: https://sysdig.com/blog/kubernetes-audit-log-falco/ [Accessed 03 August 2024].

Singh, A. and Chatterjee, K. (2017) ‘Cloud security issues and challenges: A survey’,*Journal of Network and Computer Applications*, 79, pp. 88–115, Elsevier. Available at: https://doi.org/10.1016/j.jnca.2016.11.027 [Accessed 30 January 2024].

Souppaya, M., Morello, J. and Scarfone, K. (2017) *Application container security guide*. NIST SP 800-190. Gaithersburg, MD, USA: National Institute of Standards and Technology. Available at: https://doi.org/10.6028/NIST.SP.800-190 [Accessed 12 February 2024].

Sultan, S., Ahmad, I. and Dimitriou, T. (2019) ‘Container Security: Issues, Challenges,and the Road Ahead’, *IEEE Access*, 7, pp. 52976–52996, IEEE Xplore. Available at: https://doi.org/10.1109/ACCESS.2019.2911732 [Accessed 26 January 2024].

Tosatto, A., Ruiu, P. and Attanasio, A. (2015) ‘Container-Based Orchestration in Cloud: State of the Art and , in *2015 Ninth International Conference onComplex, Intelligent, and Software Intensive Systems*, Santa Catarina, Brazil, 08-10 July 2015 pp. 70–75, IEEE Xplore. Available at: https://doi.org/10.1109/CISIS.2015.35 [Accessed 26 August 2024].

Watts, T. *et al.* (2019) ‘Insight from a Docker Container Introspection’, in *Proceedingsof the 52nd Hawaii International Conference on System Sciences*, ResearchGate. Available at: https://doi.org/10.24251/hicss.2019.863 [Accessed 27 August 2024].