Exploiting and Securing Docker Containers and Kubernetes Pods from a MitM attack: A Systematic Review

An Academic research paper for possible submission to an IEEE conference

Henry Kabuye

Submitted in partial requirements for the degree of Msc Cybersecurity

School of Computing, Engineering and Digital Technologies
United Kingdom
January 11, 2022

Supervisor: Ismail Khalid Kazmi
2nd Reader: Chunyan Mu

Exploiting and Securing Docker containers and Kubernetes pods from a MitM attack: A systematic review

Henry Kabuye
School of Computing,
Engineering & Digital Technologies
Teesside University UK
henrinnes@yahoo.co.uk

Ismail Khalid Kazmi
School of Computing,
Engineering & Digital Technologies
Teesside University UK
i.kazmi@tees.ac.uk

Chunyan Mu
School of Computing,
Engineering & Digital Technologies
Teesside University UK
c.mu@tees.ac.uk

Paolo Modesti
School of Computing,
Engineering & Digital Technologies
Teesside University UK
p.modesti@tees.ac.uk

***Abstract*—*PURPOSE* - Workloads in containers, such as Docker containers and Kubernetes pods, are just as vulnerable as workloads in non-container environments. Hackers may use many of their older techniques, like the phishing exploits, application exploits and network intrusions on these containers. This article is a systematic review research project based on design and creation that explores several novel techniques for effectively securing containerised-based operating systems from "Man in the Middle" (MitM) attack. The use of a conceptual model associated with the level of abstraction utilised for the top-level representation of communication and cryptographic primitives is a unique feature of the proposed framework in this article. The security mechanisms also include the use of a security library named "AnBxJ java library" and container firewalls based on a layer 7 of the OSI (Open System Interconnect) model. Answers to the research question; *"How can containerised-based operating systems be effectively secured from Man-in-the-Middle attack?"*, is found in this article. This article aims to assist practitioners in protecting their containerised pod installations by systematising data about Docker security practices and providing a 'zero trust' architecture on such containerised-based operating systems.**

***METHODOLOGY* - The research methodology used in this analysis is a Systematic Review (SR) composed of the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA). PRISMA relates to a compilation of results of evidence-based reviews that are in line with the same researched topic leading to evidence-based solutions.**

***FINDINGS* - Success factors were identified, and a security mechanism was implemented successfully on a containerised-based operating system scenario.**

***VALUE* - The discovered, according to the research question, may be beneficial for practitioners in protecting their Kubernetes and Docker container installations by systematising information about Kubernetes security practices and providing a 'zero trust' architecture on their containerised-based operating systems.**

***Index Terms*—Kubernetes, Docker containers, Secure mechanism, AnBxJ java Library, Orchestration.**

Teesside University, Middlesbrough, TS1 3BA, UK

I. Introduction

Containers, such as Docker, have influenced the development and deployment of many software systems [1], [2]. Containers are virtualization abstractions at the operating system level for executing separated systems on a host with a single kernel [3]. A container image is a small, standalone executable package that contains all the necessary components to execute cloud applications, such as system libraries, code, settings and system tools [3]. To safeguard digital assets and minimise the attack surface accessible to the adversary, it is critical to design resilient and secure systems [4]–[6]. The author proposed a security mechanism in conjunction with a security library "AnbxJ java library" to provide cryptographic primitives and an API for communication between the containerised-based platforms. The proposed mechanism was based on the AnB (Alice and Bob) notation to provide verification of abstract and concrete models as well as automatic generation of Java implementation [4]. Through the proposed mechanisms of creating a secure architecture that was deployed on a containerised-based platform scenario, cyber attacks such as "Man-in-the-middle (MitM)" attacks were mitigated.

Several commercial cloud service providers, such as Amazon, Google, IBM, and Microsoft, are investing increasingly on micro-service deployment utilising containerized environments to enable cloud service flexibility and quick response [7]. OpenVZ, Docker and Kubernetes are all well-known containerization technologies. Kubernetes, often known as *"K8s"*, is an open-source system used in this article for automatic deployment, scaling, and administration of applications in containers [8]. The automated coordination, management, and configuration of computer systems, applications, and services is known as *orchestration* [8]. According to Corodescu *et al* [9], complicated processes and workflows in Information technology (IT) are managed more easily with orchestration.

A zero-trust mechanism is deployed as the cybersecurity paradigm based on resource protection and the assumption that implicit trust should not be granted and then often evaluated [4], [10], [11]. Studies [4], [6], have shown that cryptography is critical for secure authentication on containerised-based enable communication systems. Rather than filtering traffic based on IP addresses, the author proposed a layer 7 firewall based on a java security library to provide an API with cryptographic primitives that can examine the contents of data packets to see whether they include malware or other cyber threats [4]. The proposed architecture is set up on a virtualised laboratory platform called the vSphere VMware platform provided by Teesside University. Virtual laboratories can play a critical role in addressing the aforesaid difficulties in containerised-based settings, particularly in the context of this project and the ethical considerations, where combining conceptual knowledge with technological abilities gains strategic relevance [5]. A virtual laboratory was setup to provide compelling options to solve the concerns mentioned above.

In the remaining sections of this article: Section II is the review of literature; section III are the methods used in the research findings; section IV are the results of the research findings; section V is the discussion of the research findings; section VI demonstrates the threats model and sections VII-IX, demonstrates a proposed security mechanism, implementation and evaluation of securing containerised-based operating systems.

Research Question (RQ) ”How can containerised-based operating systems be effectively secured from Man-in-the-Middle exploitation?”

A. *Rationale of the Study*

Hackers are more likely to use many of their older techniques, like the phishing exploits, application exploits and network intrusions on containerised-based operating systems. According to the Common Vulnerabilities and Exposures (CVE) [12], a security vulnerability *”CVE-2021-25737”* in Kubernetes was uncovered, allowing a user to re-route pod traffic to private networks on a Node. Endpoint internet protocols (IPs) in the localhost or link-local range are already prevented by Kubernetes, however the same check was not done on Endpoint Slice IPs. As a result, revealing an external IP address to access an application in a cluster is more likely to result in a problem of a ”Man-in-the-Middle (MitM)” attack. Thus, the importance of this study to effectively address this vulnerability.

B. *Technical Objectives*

The main objective was to create a secure and working architecture for the Docker containers and Kubernetes pods from Man-in-the-Middle exploitation. The traffic flow between the database and the web applications has to be secured, and only authorised applications and users can access the databases.

C. *Financial Objectives*

The task of securing the Docker containers and Kubernetes, strongly welcomes the costs factor since these platforms are based on cloud environments that have most security features chargeable. The ”high” cost of cloud services is the source of most of the present annoyance. In this article, possible solutions to these issues were addressed.

D. *Other objectives*

-To determine whether the security of Docker containers and Kubernetes can be exploited through a Man-in-the-middle attack.
-To identify the security mechanisms that can be deployed to protect Docker containers and Kubernetes pods from exploitation.
-To develop a mechanism for protecting containerised based operating systems from man-in-the-middle exploitation.
-To test the effectiveness of the mechanism in protecting containerised operating systems against man-in-the-middle attacks.

II. Review of literature

The associated work in this section does not only give an overview of secondary research but also evaluates the primary studies on the issue. This in return explains how they connect to the contributions in this article to integrate the evidence.

The necessity to build collaboration between those who develop software (Dev) and the operations team who manage production systems (Ops) created the notion of DevOps (Ops) [13], [14]. Continuous Delivery (CD) and Continuous Integration (CI) pipelines are simple to manage and deploy using Docker containers and Kubernetes pods. Studies [3], [15] show that these Docker containers and Kubernetes pods enables DevOps teams to manage container resources based on their requirements, and they even supports zero-downtime updates, rolling updates that allow incremental container updates. Studies [7], [16] demonstrate that many typical Software-defined Networking (SDN) and Transfer Layer Security (TLS) misconfigurations are caused by using the improper cypher suites. According to TrendMicro [17] report of 2020, one of the top security concerns for development and operations teams in companies has been vulnerabilities in container components deployed with cloud architecture.

Studies [18]–[20] show that Linux is the ideal operating system for the cloud orchestration mechanisms due to its durability, broad ecosystem, good hardware support, outstanding performance, and low cost. According to Biederman and Networx [21], Linux was initially considered in 2006 and the kernel namespaces functionality required to support containers in Linux matured. This puts Linux in the best position as the baseline operating system used in the artefact during this study. Docker containers are simple to set up on cloud platforms and on-premise infrastructures [3], [7], [8], [19], [20]. However, a study [22] demonstrated that the inherent packages and libraries in the containerised images of the host kernel are the source of the majority of the vulnerabilities.

III. Methods

Following Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA), the research methodology used in this study is a Systematic Review (SR) relating to a compilation of results of evidence-based reviews that are in line with the same researched topic leading to evidence-based solutions [6], [23], [24]. The activities of the study are conducted systematically and logically interconnected, thus none of them is a random activity or value [6]. The author independently screened out abstracts and papers for inclusion as well as carrying out data extraction and bias assessment.

A. Eligibility Criteria

The language considered is English and the years considered are between 2018 to 2022.

B. Information Sources

The digital database information sources considered are IEEE, ACM (Association for Computing Machinery) and Scopus database.

C. Search Strategy

A systematic search of literature concerning containerised based operating system and security mechanisms was performed. According to Jakobovic et al [25] , Boolean functions are employed in a broad range of industries, including telecommunications, coding theory, and cryptography. The Boolean operators AND, OR, and NOT were used to limit or expand the search results. This triggered a search term; ***"docker" OR "kubernetes" OR contain* AND secur****

D. Selection Process

The selection process was drawn on to the **"key concepts"** of the study while answering a set of questions drafted by the author to guide the research project to the RQ. These key concepts include:

1) `Docker` containers.
2) `Kubernetes` pods.
3) `Security` in containerised-based operating systems.
4) "`Man-in-the-middle (MitM)`" attack.

In order to maximise proof recovery in the databases while avoiding needless searches, Khadjesari *et al* [26] recommends using the **"PIO (Problem Intervention Outcome) element"** in the selection process and creation of a study question. The search strategy was made easier by populating the Boolean operators into each PIO element as illustrated in table I.

A set of four (4) questions drafted by the author guiding the research project to the RQ include:

1) What has been published on `Containerised - based` operating systems?
2) What are the current `peer reviewed` work published on containerised-based operating systems?
3) How do the authors handle `security` on containerised-based operating systems following the principles of security (Confidentiality, Integrity and Availability)?
4) How can containerised-based operating systems be effectively secured from `Man-in-the-Middle` exploitation?

TABLE I
PIO Element

PIO Element		
Component	**Description**	**Use of Boolean operators**
P - Problem	MiTM attack	"Man in the middle" OR "MitM" OR "Man-in-the-Middle"
I - Interventions	Exploiting and Securing Docker containers and Kubernetes pods	"docker" OR "kubernetes" OR contain*
O - Outcome	Secure containerised-based operating systems	AND secur*

E. Data Collection process

This data collection process was guided by a set of four (4) questions as mentioned in the earlier sub-section and populated in tables II, III and IV through a detailed search on ACM, IEEE and Scopus digital libraries respectively.

F. Data Items

Foroughi and Luksch [27] defined four (4) general phases to completing the data items of a cybersecurity project. The **first phase** is to define the problem by posing a security challenge. The security challenge in this matter was a "Man-in-the-Middle" attack and related vulnerabilities on containerized-based operating systems.

In the **second phase**, gather required information in line with the problem statement and relevant formula. In this case, the information gathered is through the ACM, IEEE and Scopus databases that are proven to give relevant information about the issue and in line with computing subjects.

The obtained data should be used in the **third phase**, during the analysis process, to offer appropriate data that may be used to anticipate or solve the identified problem.

The **final phase** is a production step, which involves deploying required modules as well as a system to conduct the entire process automatically and on a regular basis when needed. And this final stage is demonstrated in a scenario artefact deployment of a java security library to automatically detect and provide cryptographic primitives for security within the containerized based operating systems.

IV. Results

A. Study Selection related to Inclusion and Exclusion criteria

ACM, IEEE and Scopus databases recovered an initial total of 98,007; 11,844 and 197,607 articles respectively without any filters inserted. This was meant to answer the author's question *"What has been published on containerised-based operating systems?"* Considering the eligibility criteria of the author, the English language and the years between 2018 to 2022 were filtered and hence narrowing down to a total of 3,659; 4,432 and 19,757 articles for ACM, IEEE and Scopus database respectively. Including the peer reviewed articles at

TABLE II
ACM DATABASE

ACM Digital Database			
QUESTION	**KEYWORDS**	**FILTERS**	**No. of Articles**
What has been published on Containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur*	None	98,007
What are the current peer reviewed work published on containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur* AND [Publication Date: (01/01/2018 TO 31/12/2021)]	Year: 2018 to 2022 Included: Peer reviewed. Excluded: Newspapers, Book reviews. Under: Computer Science	3,659
How do the authors handle security on containerised-based operating systems following the principles of security (Confidentiality, Integrity and Availability)?	[[Title: ”docker”] OR [Title: ”kubernetes”] OR [[Title: contain*] AND [Title: secur*]]] AND [[Abstract: ”docker”] OR [Abstract: ”kubernetes”] OR [[Abstract: contain*] AND [Abstract: secur*]]] AND [[Keywords: ”docker”] OR [Keywords: ”kubernetes”] OR [[Keywords: contain*] AND [Keywords: secur*]]] AND [Publication Date: (01/01/2018 TO 31/12/2021)]	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Newspapers, Book reviews. Limit to: Title, Abstract and Author keywords. Discipline: Computer Science	56
How can containerised-based operating systems be effectively secured from Man-in-the-Middle exploitation?	[[Title: ”docker”] OR [Title: ”kubernetes”] OR [[Title: contain*] AND [Title: secur*]]] AND [[Abstract: ”docker”] OR [Abstract: ”kubernetes”] OR [[Abstract: contain*] AND [Abstract: secur*]]] AND [[Keywords: ”docker”] OR [Keywords: ”kubernetes”] OR [[Keywords: contain*] AND [Keywords: secur*]]] AND [[All: ”man in the middle”] OR [All: ”mitm”] OR [All: ”man-in-the-middle”]] AND [Publication Date: (01/01/2018 TO 31/12/2021)]	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Book reviews, Newspapers Limit to: Title, Abstract and Author keywords. Discipline: Computer Science	1

TABLE III
IEEE DATABASE

IEEE Digital Database			
QUESTION	**KEYWORDS**	**FILTERS**	**No. of Articles**
What has been published on Containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur*	None	11,844
What are the current peer reviewed work published on containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur* AND [Publication Date: (01/01/2018 TO 31/12/2021)]	Year: 2018 to 2022 Included: Peer reviewed. Excluded: Newspapers, Book reviews. Under: Computer Science	4,432
How do the authors handle security on containerised-based operating systems following the principles of security (Confidentiality, Integrity and Availability)?	(”Abstract”:”docker” OR ”Abstract”:”kubernetes” OR ”Abstract”:contain* AND ”Abstract”:secur*) AND (”Document Title”:”docker” OR ”Document Title”:”kubernetes” OR ”Document Title”:contain* AND ”Document Title”:secur*) AND (”Author Keywords”:”docker” OR ”Author Keywords”:”kubernetes” OR ”Author Keywords”:contain* AND ”Author Keywords”:secur*)	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Newspapers, Book reviews, TUP publisher. Limited to: Title, Abstract and Author keywords. Discipline: Computer Science	50
How can containerised-based operating systems be effectively secured from Man-in-the-Middle exploitation?	(”Document Title”:”docker” OR ”Document Title”:”kubernetes” OR ”Document Title”:contain* AND ”Document Title”:secur*) AND (”Abstract”:”docker” OR ”Abstract”:”kubernetes” OR ”Abstract”:contain* AND ”Abstract”:secur*) AND (”Author Keywords”:”docker” OR ”Author Keywords”:”kubernetes” OR ”Author Keywords”:contain* AND ”Author Keywords”:secur*) AND (”Full Text & Metadata”:”Man in the middle” OR ”Full Text & Metadata”:”MitM” OR ”Full Text & Metadata”:”Man-in-the-Middle”)	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Book reviews, Newspapers Limit to: Title, Abstract and Author keywords. Discipline: Computer Science	5

this stage, answers the author's second question; *”What are the current peer reviewed work published on containerised-based operating systems?”* Furthermore, book reviews and newspapers were excluded from the search as well as limiting it to the *”Title”, ”Abstract” and ”Author keywords”*, which resulted into a total of 56; 50 and 74 for ACM,IEEE and Scopus database respectively. To narrow down the search results to enable them point towards the intended RQ the *AND ”Man in the middle” OR ”MitM” OR ”Man-in-the-Middle”* filter was implemented into the search strategy. And finally, a total of seven (7) articles were found relevant to the research question after inserting in the necessary filters into the search strategy with a ratio 1:5:1 for ACM, IEEE and Scopus respectively.

B. Results of Syntheses

The synthesis of the PIO element of the included papers serves as a foundation for interpreting the review's conclusions and are a valuable review output in and of themselves found in tables VI, VII, VIII, IX, X, XI, XII. The illustrated results in tables VI, VII, VIII, IX, X, XI, XII are tabulated by screening out six (6) itemised fields below.

1) `Publication Author` of the study selection.
2) `Framework tools` used in the study.
3) `Assessment tools` used in the study.
4) `Baseline Operating System` used in the study.
5) `Related Vulnerabilities tested` during the study.
6) `Mitigation Techniques` suggested or used in the study.

TABLE IV
SCOPUS DATABASE

Scopus Digital Database			
QUESTION	**KEYWORDS**	**FILTERS**	**No. of Articles**
What has been published on Containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur*	None	197,607
What are the current peer reviewed work published on containerised-based operating systems?	”docker” OR ”kubernetes” OR contain* AND secur* AND [Publication Date: (01/01/2018 TO 31/12/2021)]	Year: 2018 to 2022 Included: Peer reviewed. Excluded: Newspapers, Book reviews. Under: Computer Science	19,757
How do the authors handle security on containerised-based operating systems following the principles of security (Confidentiality, Integrity and Availability)?	(TITLE (”docker” OR ”kubernetes” OR contain* AND secur*) AND ABS (”docker” OR ”kubernetes” OR contain* AND secur*) AND KEY (”docker” OR ”kubernetes” OR contain* AND secur*)) AND PUBYEAR ¿ 2017 AND PUBYEAR ¡ 2022 AND (LIMIT-TO (SUBJAREA , ”COMP”)) AND (EXCLUDE (DOCTYPE , ”ch”) OR EXCLUDE (DOCTYPE , ”re”))	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Newspapers, Book reviews. Limited to: Title, Abstract and Author keywords. Discipline: Computer Science	74
How can containerised-based operating systems be effectively secured from Man-in-the-Middle exploitation?	(TITLE (”docker” OR ”kubernetes” OR contain* AND secur*) AND ABS (”docker” OR ”kubernetes” OR contain* AND secur*) AND KEY (”docker” OR ”kubernetes” OR contain* AND secur*)) AND PUBYEAR ¿ 2017 AND PUBYEAR ¡ 2022 AND ALL (”Man in the middle” OR ”MitM” OR ”Man-in-the-Middle”) AND (LIMIT-TO (SUBJAREA , ”COMP”)) AND (EXCLUDE (DOCTYPE , ”ch”) OR EXCLUDE (DOCTYPE , ”re”))	Year: 2018 to 2022 Included: Peer Reviewed Excluded: Book reviews, Newspapers Limit to: Title, Abstract and Author keywords. Discipline: Computer Science	1

TABLE V
STUDY SELECTION

Included	
PUBLICATION	**PUBLISHER**
A. Mailewa, S. Mengel, L. Gittner and H. Khan, ”Vulnerability Prioritization, Root Cause Analysis, and Mitigation of Secure Data Analytic Framework Implemented with MongoDB on Singularity Linux Containers.,” ACM, 2020. [16]	ACM
W. S. S. Ahamed, P. Zavarsky and B. Swar, ”Security Audit of Docker Container Images in Cloud Architecture,” in 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), Jalandhar, 2021. [22]	IEEE
J. Wenhao and L. Zheng, ”Vulnerability Analysis and Security Research of Docker Container,” in 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, 2020. [28]	IEEE
S. Sultan, I. Ahmad and T. Dimitriou, ”Container Security: Issues, Challenges, and the Road Ahead,” IEEE Access, vol. 7, pp. 52976 - 52996, 2019. [29]	IEEE
B. Pearson and D. Plante, ”Secure Deployment of Containerized IoT Systems,” in 2020 SoutheastCon, Raleigh, 2020. [30]	IEEE
A. Tomar , D. Jeena, P. Mishra and R. Bisht, ”Docker Security: A Threat Model, Attack Taxonomy and Real-Time Attack Scenario of DoS,” in 2020 , Noida, 2020. [31]	IEEE
W. Zhang, K. Li, T. Li, S. Niu and Z. Gao, ”CM-droid: Secure container for android password misuse vulnerability,” Computers, Materials and Continua, vol. 59, no. 1, pp. 181-198, 2019. [32]	Scopus

TABLE VI
RESULT 1 OF SYNTHESES

Result 1	
Publication Author	Mailewa et al [16]
Framework Tools	MongoDB
Assessment Tools	OpenVas, Dagda, PortSpider, MongoAudit, Nmap, Metasploit-Framework, Nessus
Baseline Operating System	Linux
Related Vulnerabilities Tested	TCP timestamp issues, Log files issues, MongoDB version exposed, ICMP timestamp issues, Network traffic restriction issues, SSL/TLS cipher issues, MongoDB port issues, OpenSSH issues, Postfix service issues, Blacklist and blacklist issues, DoS issues, GRUB bootloader issues, TLS/SSL encryption issues, User authentication issues, /tmp directory issues.
Mitigation Techniques	Vulnerability Prioritization, Root causes analysis, Prevention Techniques [31]

TABLE VII
RESULT 2 OF SYNTHESES

Result 2	
Publication Author	Ahamed, Zavarsky and Swar [22]
Framework Tools	Google Docker Images
Assessment Tools	Google Container Registry
Baseline Operating System	Google Cloud (PaaS)
Related Vulnerabilities Tested	Package version issues, Default user Privilege issues, Configuration issues
Mitigation Techniques	Security Audit

V. DISCUSSION

This document presents a systematic review of the literature basing on seven (7) articles that point out different security mechanisms to secure containerised based operating systems from MitM attacks. The next sub-sections discuss the evaluation of the collected data-sets and the analysis of the work.

A. Evaluation of data-sets

Verification of the studied techniques and results are compared with other techniques inline with securing containerised based operating systems. The evaluation of the proposed approaches in the study is done using ”real” data as explained below:

1) `"Theoretical approach"` for studies that implement a theoretical mitigation approach in their specified case scenario.

TABLE VIII
RESULT 3 OF SYNTHESES

Result 3	
Publication Author	Wenhao and Zheng [28]
Framework Tools	Docker Schema
Assessment Tools	File system isolation, Remote isolation and communication isolation, host resource constraints and Device management, image transmission and Network isolation
Baseline Operating System	SELinux
Related Vulnerabilities Tested	Dos issues, Root privilege issues, MAC flooding issues, Man in the middle issues.
Mitigation Techniques	Linux Security Modules(LSM), AppArmor, Privileges capability mechanism, Secure Computation Mode(Seccomp), mandatory access control, Netfilter, Linux kernel Integrity protection technology, Log audit, security threat detection, Streamline operation system

TABLE IX
RESULT 4 OF SYNTHESES

Result 4	
Publication Author	Sultan, Ahmad and Dimitriou [29]
Framework Tools	Docker
Assessment Tools	Docker Image Vulnerability Analysis Framework (DIVA) [33]
Baseline Operating System	Linux
Related Vulnerabilities Tested	Meltdown and spectra attacks
Mitigation Techniques	CGroup mechanisms, namespace mechanisms, capability mechanism, Secure Computation Mode(Seccomp), Linux Security Modules(LSM)

TABLE X
RESULT 5 OF SYNTHESES

Result 5	
Publication Author	Pearson and Plante [30]
Framework Tools	Docker
Assessment Tools	N/A
Baseline Operating System	Linux
Related Vulnerabilities Tested	man in the middle attack
Mitigation Techniques	N/A

TABLE XI
RESULT 6 OF SYNTHESES

Result 6	
Publication Author	Tomar *et al* [31]
Framework Tools	Docker
Assessment Tools	NMap
Baseline Operating System	Linux
Related Vulnerabilities Tested	Malware, Dos attack, privilege escalation, Escape attack, ARP spoofing, MAC flooding, MitM, Unauthorised access, Poisoned images, out-dated software, Attack using CCAT, Malicious code injection, crypto-jacking, Tampering, Kernel exploit
Mitigation Techniques	N/A

TABLE XII
RESULT 7 OF SYNTHESES

Result 7	
Publication Author	Zhang *et al* [32]
Framework Tools	Android
Assessment Tools	CM-Droid
Baseline Operating System	Linux
Related Vulnerabilities Tested	Password issues
Mitigation Techniques	CM-Droid secure container

2) `"Practical approach"` for studies that implement a practical approach in their specified case scenario.
3) `"Machine Learning (ML) / Artificial Intelligence (AI) approach"` for studies that implement a Machine learning approach in their specified case scenario.

Table XIII shows the study approaches used as per specified publication author.

TABLE XIII
REAL DATA APPROACHES

Real data			
Publication Author	**Theoretical approach**	**Practical approach**	**Machine Learning approach**
Mailewa et al [16]	✓		
Ahamed, Zavarsky and Swar [22]	✓		
Wenhao and Zheng [28]	✓		
Sultan, Ahmad and Dimitriou [29]	✓		
Pearson and Plante [30]	✓	✓	
Tomar *et al* [31]	✓	✓	
Zhang *et al* [32]	✓	✓	

B. Analysis of the data-sets

The analysis of the data-sets is done using the SWOT *(”Strengths”, ”Weaknesses”, ”Opportunities”, ”Threats”)* analysis tool to disseminate the content to detail [34].
Strengths: - The vulnerability assessment tools in the searched data presented quality results for the specified cases. For instance, Mailewa et al [16] presented the use of OpenVAS, Dagda, Portfider, MongoAudit, nmap and Nessus vulnerability assessment tools to come to conclusive results.
Weaknesses: - The searched data lacked concurrency security design solutions on containerised based operating systems [35]. For instance, they lacked security designs related to programs that are in a collection of independent processes that eventually run in parallel and with the same outcome. Moreover, they lacked solutions embedded with machine learning algorithms that could create at least a training set for detected and classified vulnerabilities. This could eliminate known and future vulnerabilities having the same features in their running processes
Opportunites: - The security mechanisms in the searched data-sets demonstrate detection of different cyber-attacks while providing confidentiality and integrity of applications

in the containers. But they all ignore the security of the underlying container's image. In this case, the attacker may replace the original container image with an infected image to run undetected. The 2019 McAfee [36] report indicated that ”Dockerhub” found several of its images used maliciously by unauthorised agents. This creates an opportunity of building a security mechanism that could detect anomalies in containers. However, Karn et al [37] proposed a strategy of used Machine Learning (ML) algorithms for detection and classification Kubernetes pods to determine whether the data flow contains running processes or not.

Threats:- Security on containerised based operating systems is on a rise as one of the major concerns of the researchers. According to the searched data, the mitigation techniques are focused on the outside of the container application and are ignoring the inside of the container. This increases the attack surface of the containers to more likely attacks such as the MitM attacks.

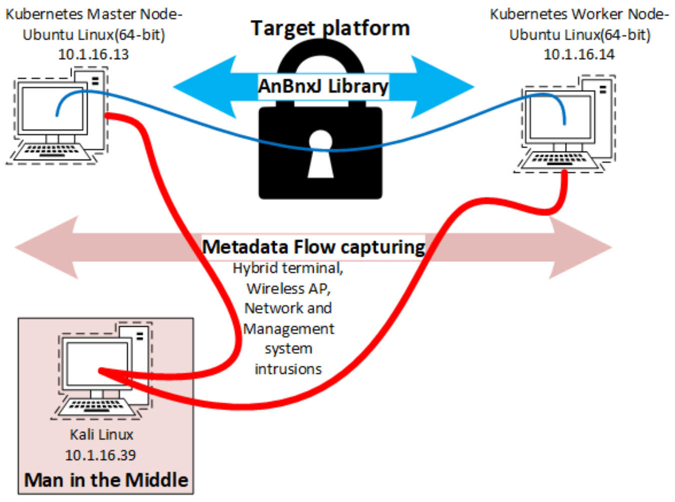


Fig. 1. *The Attack Tree Scenario*

VI. The Threat Model

A. *The ”Man-in-The-Middle (MiTM)” attack.*

Haselsteiner and Breitfuß [38] derived an issue that the adversary intercepts and modifies the communication between the receiver and the sender, who believe they are conversing securely. An attacker can alter the communications so that they reach her or his intended target.

The recently discovered Kubernetes security vulnerability *”CVE-2021-25737* [12],” which allows a user to re-route pod traffic to private networks on a Node, highlights the need for developers to double-check the security of Endpoint Slice IPs. Kubernetes, on the other hand, already blocks endpoint internet protocols (IPs) in the localhost or link-local range. Basically, disclosing an external IP address to access a cluster application is more likely to result in a ”Man-in-the-Middle (MitM)” attack.

Figure 1 illustrates the attack tree scenario of the MitM attack and the proposed AnBxJ java security library to provide an API and cryptographic primitives between the master node (10.1.16.13) and the worker node (10.1.16.14). The master-node and worker node are configured in separate containers and on separate linux based operating systems.

B. *Vulnerability Assessment*

In order to assess effectively the discussed vulnerability, it is very important to understand the architecture of Kubernetes as illustrated in figure 2. The master node is in charge of controlling and managing the cluster. The *etcd, kube-scheduler, kube-apiserver* and *kube-controller* are the four essential components [8]. The *etcd* is a datastore that is used to store the cluster's configuration and status. The Kubernetes control plane's front end is the *kube-apiserver*. To connect with the Kubernetes cluster, the user or management request must communicate with the *kube-apiserver*. The *kube-scheduler* keeps an eye on unscheduled pods and allocates them to a node based on a variety of parameters, including resource constraints, affinity, and anti-affinity policies. And to keep the cluster in the proper condition, the *kube-controller* keeps an eye on it all the time.

Mailewa *et al* [16] asserts that vulnerability assessment and management is a very important part in the information technology risk management. According to Mailewa *et al* [16], vulnerability assessment must be prioritised in a likelihood manner (”Low”, ”Medium”, and ”High”) through an initial analysis of the root cause of the vulnerability in the system's current state. However in this research report, the author considers an initial analysis on the presence of the *”CVE-2021-25737* [12] vulnerability on Kubernetes pods. Moreover, an open-source OpenVAS vulnerability assessment scanner was used to determine the severity of *”CVE-2021-25737* vulnerability [39].

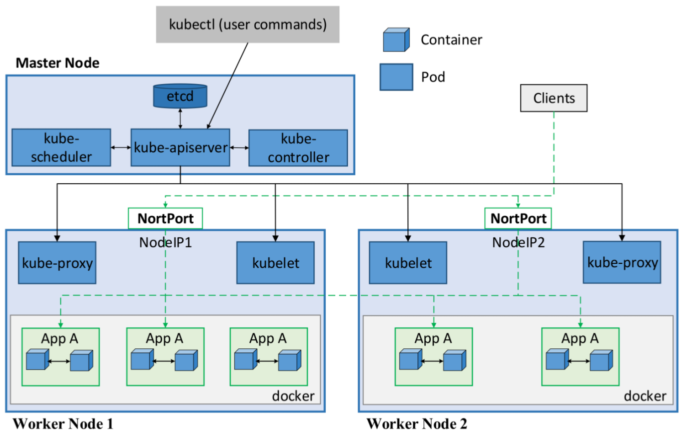


Fig. 2. *Kubernetes Architecture [8]*

VII. Proposed Security Model and Design

Security protocols and mechanisms are essential for building secure and stable distributed applications such as Kubernetes, however, their implementation is quite challenging and error prone. Ahamed, Zavarsky and Swar [22] presented a security audit mechanism on Docker containers as well as a cloud architecture embedded with java codes. The author proposed a security mechanism based on a simple language AnB to provide an abstract model and concrete model as shown in figure 3 for formal verification and automatic Java implementation [4], [40]. The proposed architecture is a basis of concurrency java script programs embedded with containerised based programs that are running parallel independent processes, collectively to arrive at the same outcome [35], [40]. Figure 5 illustrates the proposed secure docker architecture embedded with the AnBxJ security library.

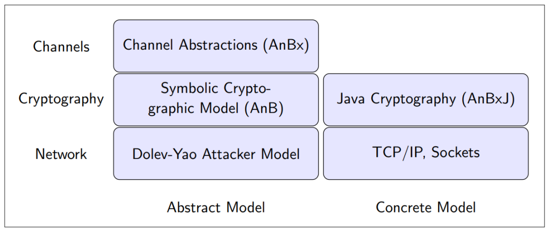


Fig. 3. *Abstract and Concrete model [4].*

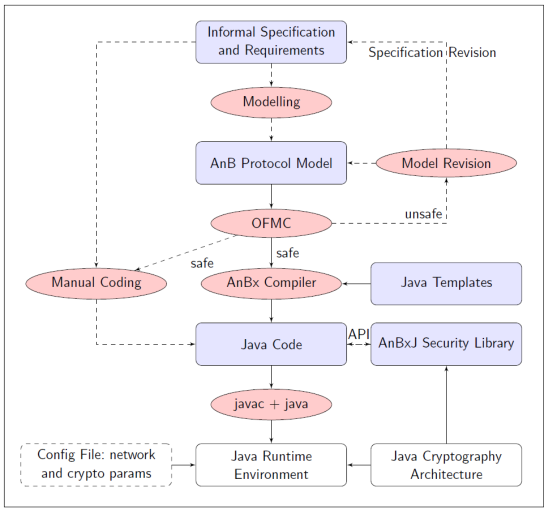


Fig. 4. *Framework: Workflow and Toolkit [4]*

VIII. Implementation and Evaluation

A sample scenario artefact was designed to illustrate the researched effective mechanisms of securing docker containers and Kubernetes pods. Three virtual machines were installed on a vSphere VMware platform. Two of these were Ubuntu 18.04.6 versions and one was Kali Linux. One Ubuntu machine was to serve as the master node and the other Ubuntu machine to serve as the worker node. The master node machine (10.1.16.13) is set to have 2core CPUs and 4GB RAM as well as the worker node (10.1.16.14). The ubuntu machines must go into a series of installation commands for the Kubernetes

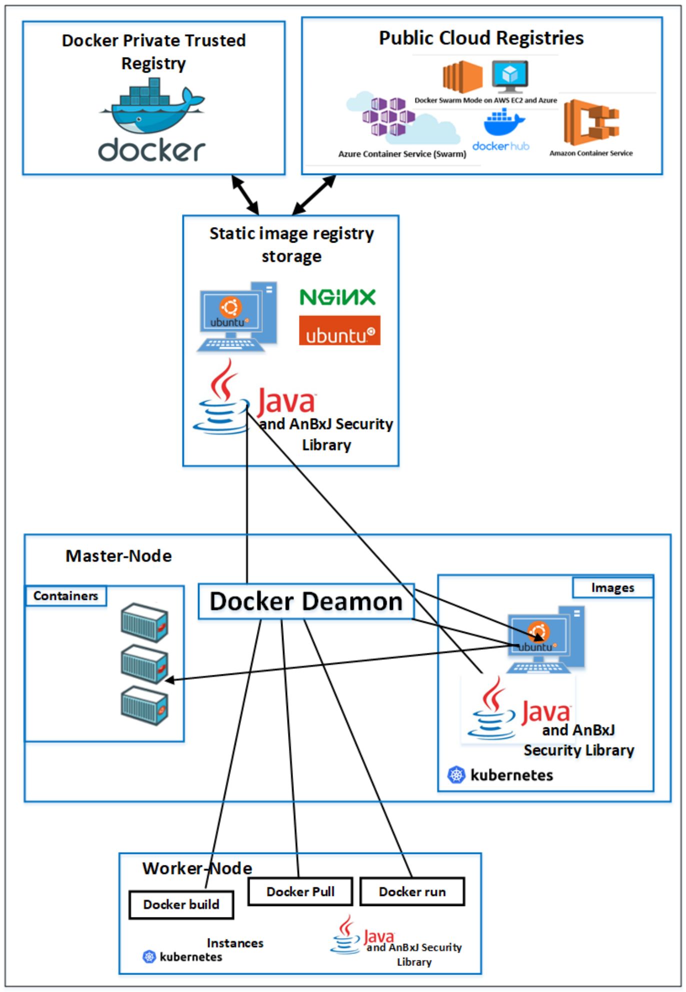


Fig. 5. *Proposed Secure Docker architecture with AnBxJ security Library.*

pods to work effectively [41]. For instance the *swap* has to be disabled on both machines with a command *"swapoff -a"*. The docker 20.10.7 version is installed on both Ubuntu machines. By using the *"sudo systemctl enable docker"* command to enable the installed docker on both machines, activates the docker into the running mode. For Kubernetes to run effectively, it requires the installation of tools *"kubeadm, kubelet and kubectl"* on both the worker node machine and the master node machine [3], [8].

The AnBxJ library safeguarded communication between the master-node and the worker-node, therefore the application required development methods and testing procedures for recorded meta-data. Facts in modelling security attributes generated during the compilation chain were required for the study of the AnB security goals [4], [42].

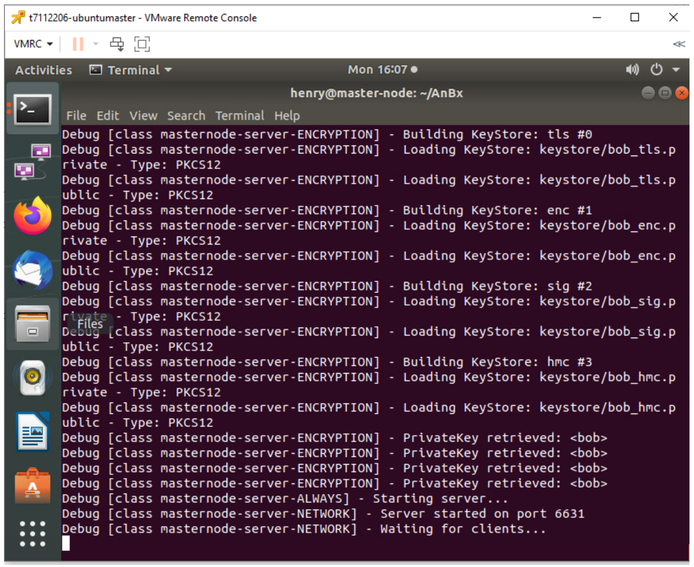
A. *Connection-Queue*

The programmed Java code is designed to accept connections during authentication processes, then close connections in cases of unauthorised access and at the end of each session.

Both the worker-node and master-node received private keys as a result of the encryption operations.

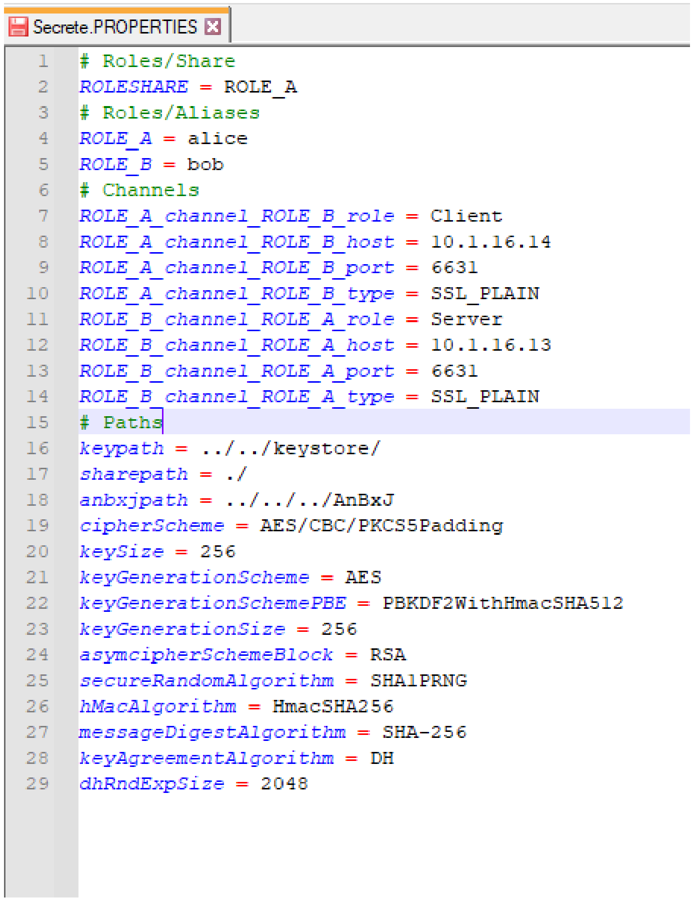
B. Multi-listener

A set of compiled java codes activated a multi-listener during the session on the master-node server. The master-node server had the capacity of listening to only connections configured in the ***".PROPERTIES" file***. Figure 6 illustrates the master-node server waiting for connections as configured in the ***".PROPERTIES" file*** to come via its safe and resilient channel (In this case; Port 6631).


t7112206-ubuntumaster - VMware Remote Console
VMRC
Activities
Terminal
Mon 16:07
henry@master-node: ~/AnBx
File Edit View Search Terminal Help
Files

```
Debug [class masternode-server-ENCRYPTION] - Building KeyStore: tls #0
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_tls.p
rivate - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_tls.p
ublic - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Building KeyStore: enc #1
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_enc.p
rivate - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_enc.p
ublic - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Building KeyStore: sig #2
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_sig.p
rivate - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_sig.p
ublic - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Building KeyStore: hmc #3
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_hmc.p
rivate - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - Loading KeyStore: keystore/bob_hmc.p
ublic - Type: PKCS12
Debug [class masternode-server-ENCRYPTION] - PrivateKey retrieved: <bob>
Debug [class masternode-server-ENCRYPTION] - PrivateKey retrieved: <bob>
Debug [class masternode-server-ENCRYPTION] - PrivateKey retrieved: <bob>
Debug [class masternode-server-ENCRYPTION] - PrivateKey retrieved: <bob>
Debug [class masternode-server-ALWAYS] - Starting server...
Debug [class masternode-server-NETWORK] - Server started on port 6631
Debug [class masternode-server-NETWORK] - Waiting for clients...
```

Fig. 6. *Master-node waiting for clients.*



```
# Roles/Share
ROLESHARE = ROLE_A
# Roles/Aliases
ROLE_A = alice
ROLE_B = bob
# Channels
ROLE_A_channel_ROLE_B_role = Client
ROLE_A_channel_ROLE_B_host = 10.1.16.14
ROLE_A_channel_ROLE_B_port = 6631
ROLE_A_channel_ROLE_B_type = SSL_PLAIN
ROLE_B_channel_ROLE_A_role = Server
ROLE_B_channel_ROLE_A_host = 10.1.16.13
ROLE_B_channel_ROLE_A_port = 6631
ROLE_B_channel_ROLE_A_type = SSL_PLAIN
# Paths
keypath = ../../keystore/
sharepath = ./
anbxjpath = ../../../AnBxJ
cipherScheme = AES/CBC/PKCS5Padding
keySize = 256
keyGenerationScheme = AES
keyGenerationSchemePBE = PBKDF2WithHmacSHA512
keyGenerationSize = 256
asymcipherSchemeBlock = RSA
secureRandomAlgorithm = SHA1PRNG
hMacAlgorithm = HmacSHA256
messageDigestAlgorithm = SHA-256
keyAgreementAlgorithm = DH
dhRndExpSize = 2048
```

Fig. 7. *sample properties file*

C. Detection

A critical examination was done on the metadata flow for endpoint addresses under the list of EndpointSlices to see if the *"CVE-2021-25737* [12] vulnerability was exploited. Using the openVAS vulnerability assessment scanner, the severity of MitM attacks was shown as high on the list of EndpointSlices [39].

D. Mitigation

A Study [43] shows a mitigation mechanism through construction of a validating admission webhook that prohibits EndpointSlices with endpoint addresses from exploiting this vulnerability (*"CVE-2021-25737*) without having to upgrade kube-apiserver. However, a policy establishment is required to impose this limitation in case of an existing admission policy mechanism as well as the requirement of upgrading the kube-episever.

In this regard, the author proposed a security mechanism that could eliminate all the hardships of timely upgrading of the kube-apiserver and easily configured by software developers. The security mechanism was designed basing on the Java programming language and cryptographic primitives offered by the Java Cryptography Architecture (JCA) [40]. Moreover, the multi-listener and connection-queue feature properties in the proposed mechanism, give an added advantage to eliminate malicious connections proactively. Thus, eliminating the EndpointSlices with endpoint addresses from exploiting the *"CVE-2021-25737* vulnerability. The author used the AnBxJ library which is a sub section of an open-source tool called the AnBx Compiler and Code Generator [42]. The ***".PROPERTIES" file*** as shown in figure 7 and several Java script codes to be found in the Github [44] platform. These codes and the AnBxJ Java library were compiled by the AnBx Compiler and Code Generator [42]. Figure 8 illustrates the AnBxJ Java library architecture.

IX. Future work and Conclusion

Java as a programming language has been ignored by many security developers through bias and the sensitivity of cases when writing their scripts for debugging. The researched security mechanisms in this work shows that the power of security in API applications remains in java cryptographic architectures (JCA). Moreover, this paper presents a systematic review of reviewed studies on securing containerised-based operating systems and demonstrates a java security mechanism. The java security mechanism proposed in this review, is based on a simple language AnB (Alice and Bob) notation to provide an

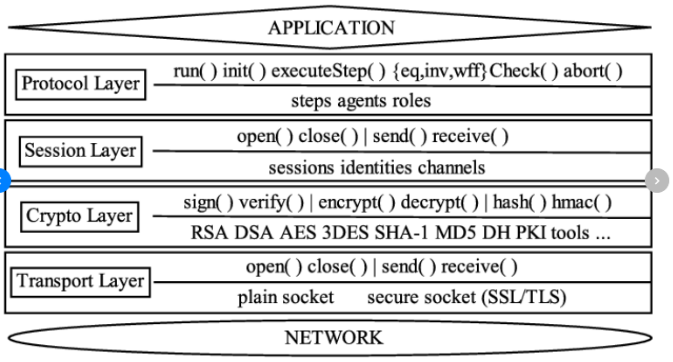

APPLICATION
Protocol Layer
run( ) init( ) executeStep( ) {eq,inv,wff}Check( ) abort( )
steps agents roles
Session Layer
open( ) close( ) | send( ) receive( )
sessions identities channels
Crypto Layer
sign( ) verify( ) | encrypt( ) decrypt( ) | hash( ) hmac( )
RSA DSA AES 3DES SHA-1 MD5 DH PKI tools ...
Transport Layer
open( ) close( ) | send( ) receive( )
plain socket secure socket (SSL/TLS)
NETWORK

Fig. 8. *AnBxJ Java library architecture [4]*

abstract and concrete model. Further studies may focus on java security firewalls on concurrency designs using the AnB novel notation and machine learning (ML) algorithms to detect and classify anomalies in containerised based operating systems.

X. Ethical statement

The project is a research-based investigation that used Teesside University resources to study various digital platforms online while adhering to Teesside University's ethics and behaviour. The artefact requires no additional human interaction and was created using the University's vSphere VMware virtualised platform, which Teesside University has fully protected to prevent any harm to any organisation or individual.

References

[1] D. Reis, B. Piedade, F. F. Correia, J. P. Dias and A. Aguiar, "Developing Docker and Docker-Compose Specifications: A Developers' Survey," IEEE Access, vol. 4, pp. 1-1, 2021.
[2] A. Bhardwaj and C. R. Krishna, "Virtualization in Cloud Computing: Moving from Hypervisor to Containerization—A Survey," Arabian Journal for Science and Engineering, vol. 46, p. 8585–8601, 2021.
[3] Casalicchio E. (2019) Container Orchestration: A Survey. In: Puliafito A., Trivedi K. (eds) Systems Modeling: Methodologies and Tools. pp. 221-235, EAI/Springer Innovations in Communication and Computing. Springer, Cham. https://doi.org/10.1007/978-3-319-92378-9-14
[4] P. Modesti, "Integrating Formal Methods for Security in Software Security Education," Informatics in Education, vol. 19, no. 1648-5831, pp. 425-454, 2020.
[5] R. Laschi and A. Riccioni, "Design and implementation of a virtual lab for supporting students in modeling, evaluating and programming secure systems," in Proc. of the 13th International Conference on Interactive Computer-Aided Learning (ICL), Villach, 2008.
[6] B. J. Oates, Researching information systems and computing, SAGE PUBLICATIONS, 2005.
[7] N. G. Bachiega, P. S. L. Souza, S. M. Bruschi and S. d. R. S. d. Souza, "Container-Based Performance Evaluation: A Survey and Challenges," in 2018 IEEE International Conference on Cloud Engineering (IC2E), Orlando, 2018.
[8] Kubernetes, "Production-Grade Container Orchestration," 2021. [Online]. Available: https://kubernetes.io/. [Accessed 1 11 2021].
[9] A.-A. Corodescu, N. Nikolov, A. Q. Khan, A. Soylu, M. Matskin, A. H. Payberah and D. Roman, "Locality-Aware Workflow Orchestration for Big Data," in Proceedings of the 13th International Conference on Management of Digital EcoSystems, Tunisia, 2021.
[10] Scott Rose, Oliver Borchert, Stu Mitchell and Sean Connelly, Zero Trust Architecture, NIST, [online] Available: https://doi.org/10.6028/NIST.SP.800-207.
[11] Andrew Goodman, What Is Zero Trust?. Accessed on: October 26, 2021. Available: https://dzone.com/articles/what-is-zero-trust
[12] CVE, "CVE list," CVE, 21 01 2021. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-25737. [Accessed 11 10 2021].
[13] M. Loukides, What is DevOps?, O'Reilly Media, Inc., 2012.
[14] R. Jabbari, N. b. Ali, K. Petersen and B. Tanveer, "What is DevOps? A Systematic Mapping Study on Definitions and Practices," in In Proceedings of the Scientific Workshop Proceedings of XP2016 (XP '16 Workshops), New York, 2016.
[15] P. Heidari, Y. Lemieux and A. Shami, "QoS Assurance with Light Virtualization - A Survey," in 2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Luxembourg, 2016.
[16] A. Mailewa, S. Mengel, L. Gittner and H. Khan, "Vulnerability Prioritization, Root Cause Analysis, and Mitigation of Secure Data Analytic Framework Implemented with MongoDB on Singularity Linux Containers.," ACM, 2020.
[17] TrendMicro, "The New Norm: TrendMicro Security Predictions for 2020," TrendMicro, 2020. [Online]. Available: https://www.trendmicro.com/vinfo/us/security.
[18] A. Celesti, D. Mulfari, M. Fazio, M. Villari and A. Puliafito, "Exploring Container Virtualization in IoT Clouds," St. Louis, 2016.
[19] Tuomas Autio, "Securing a Kubernetes Cluster on Google Cloud Platform" 2021.
[20] Zhiheng Zhong and Rajkumar Buyya. 2020. A Cost-Efficient Container Orchestration Strategy in Kubernetes-Based Cloud Computing Infrastructures with Heterogeneous Resources. ACM Trans. Internet Technol. 20, 2, Article 15 (May 2020), 24 pages. DOI:https://doi.org/10.1145/3378447.
[21] E. W. Biederman and L. Networx, "Multiple instances of the global linux namespaces", Proceedings of the Linux Symposium. Citeseer, 2006.
[22] W. S. S. Ahamed, P. Zavarsky and B. Swar, "Security Audit of Docker Container Images in Cloud Architecture," in 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), Jalandhar, 2021.
[23] R. Linger and L. Goldrich, Agile research for cybersecurity, Institute for Information Infrastructure Protection, 2014.
[24] L. Cohen, L. Manion and K. Morrison, Research Methods in Education, 8 Edition ed., London Newyork: Taylor Francis Group, 2017.
[25] D. Jakobovic, S. Picek, M. S. R. Martins and M. Wagner, "Toward more efficient heuristic construction of Boolean functions," Applied Soft Computing, vol. 107, pp. 1568-4946, 2021
[26] Z. Khadjesari, E. Murray, C. Hewitt, S. Hartley and C. Godfrey, "Can stand-alone computer-based interventions reduce alcohol consumption? A systematic review," Society for the study of addiction, vol. 106, no. 2, pp. 267-282, 2010.
[27] F. Foroughi and P. Luksch, "Data science methodology for cybersecurity projects," arXiv preprint arXiv:1803.04219, 2018.
[28] J. Wenhao and L. Zheng, "Vulnerability Analysis and Security Research of Docker Container," in 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, 2020.
[29] S. Sultan, I. Ahmad and T. Dimitriou, "Container Security: Issues, Challenges, and the Road Ahead," IEEE Access, vol. 7, pp. 52976 - 52996, 2019.
[30] B. Pearson and D. Plante, "Secure Deployment of Containerized IoT Systems," in 2020 SoutheastCon, Raleigh, 2020.
[31] A. Tomar, D. Jeena, P. Mishra and R. Bisht, "Docker Security: A Threat Model, Attack Taxonomy and Real-Time Attack Scenario of DoS," in 2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence), Noida, 2020.
[32] W. Zhang, K. Li, T. Li, S. Niu and Z. Gao, "CM-droid: Secure container for android password misuse vulnerability," Computers, Materials and Continua, vol. 59, no. 1, pp. 181-198, 2019.
[33] R. Shu, X. Gu, and W. Enck, "A study of security vulnerabilities on docker hub," in Proc. 7th ACM Conf. Data Appl. Secur. Privacy, Mar. 2017, pp. 269-280.
[34] G. Sepehr, A. Mansoureh, and A.-M. Mandana, "Swot methodology: A state-of-the-art review for the past, a framework for the future," Journal of Business Economics and Management, vol. 12(1), p. 24–48, 2011.
[35] B. Pervan and J. Knezović, "A Survey on Parallel Architectures and Programming Models," in 2020 43rd International Convention on Infor-

mation, Communication and Electronic Technology (MIPRO), Opatija, 2020.
[36] McAfee, ”rp-quarterly-threats-aug-2019.pdf,” 2019. [Online]. Available: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-aug-2019.pdf. [Accessed 26 10 2021].
[37] R. R. Karn, P. Kudva, H. Huang, S. Suneja and I. M. Elfadel, ”Cryptomining Detection in Container Clouds Using System Calls and Explainable Machine Learning,” IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 3, pp. 674-691, 2021.
[38] E. Haselsteiner and K. Breitfuß, ”Security in near field communication (NFC),” In Workshop on RFID security , pp. 12-14, 2006.
[39] Greenbone, ”OpenVAS – Open Vulnerability Assessment Scanner,” Greenbone, 2021. [Online]. Available: https://www.openvas.org/. [Accessed 30 09 2021].
[40] ORACLE, ”Java Cryptography Architecture (JCA) Reference Guide,” Oracle, 2021. [Online]. Available: https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html. [Accessed 30 09 2021].
[41] Giridharaprasad, ”Setup Kubernetes cluster using kubeadm in vSphere virtual machines.,” Medium, 27 01 2020. [Online]. Available: https://gprasath.medium.com/setup-kubernetes-cluster-using-kubeadm-in-vsphere-virtual-machines-985372ee5b97. [Accessed 13 10 2021].
[42] P. Modesti, AnBx: Automatic Generation and Verification of Security Protocols Implementations International Symposium on Foundations Practice of Security (FPS 2015), Clermont-Ferrand, France, 2015
[43] A. Terzolo, ”Enabling Multi-Tenancy and Fine-grained Security in a Multi-Cluster Architecture,” Politecnico di Torino, 2021.
[44] H. Kabuye, ”Exploiting-securing-Docker-and-kubernetes,” Github, 10 09 2021. [Online]. Available: https://github.com/Henrinnes/exploiting-securing-Docker-and-kubernetes. [Accessed 2 10 2021].