OULUN YLIOPISTO
UNIVERSITY of OULU

Security Challenges of Microservices

University of Oulu
Department of Information
Processing Science
Bachelor’s thesis
Samppa Jalonen
09.06.2022

Abstract

Security issues regarding microservice are well researched, however the different security issues and solutions have not been brought together as yet. This study searched through academic databases to find out what security issues and proposed solutions or mitigation methods can be found in existing literature. It found several security issues and methods in literature. Most security issues are raised regarding microservice that externally facing or in open environment. Majority of sources addressed security monitoring and authentication and authorization issues, fewer studies on implementation and bug-related issues such as container implementation and -bugs and some on networking related issues. This study found also that there is some amount of disconnect in literature when it comes to addressing security issues and their solutions and mitigation methods. The study offers a more detailed account of existing microservice security issues and solutions.

Keywords: Microservices, Security

Contents

1. Introduction

The purpose of this study is to find out what existing security issues have been identified by the academic literature and what sort of mitigation and or solution there are proposed against them in microservices.

Microservices are widely in use among large enterprises such as Netflix, as such security issues related to there are also seem to be certain amount of differences in existing literature when it comes to evaluating security issues such as issues with isolation and networking (Pereia-Vale, Anelis, Marquez, Astudillo, & Fernandez, 2019) that however aren't raised by other sources such as (Li, et al., 2020). Finding out what sort of issues and solutions there are in academic literature can be useful for understanding said issues and solutions.

There are previous studies such as a systematic mapping pilot study by (Pereia-Vale, Anelis, Marquez, Astudillo, & Fernandez, 2019) that did not address possible solutions in more detail and (Li, et al., 2020) while addressing a large quantity of existing issues and solutions in literature doesn't address all that have been raised by the former. This study seeks to find out what issues and solution / mitigation methods there are in literature.

This study uses systematic literature review and uses several academic databases for data gathering. As of now, there has not been studies that approaches these issues from systematic literature review (Kitchenham, 2004).

There is no clear majority for any particular security issue. Security issues overall consist of issues tied to preventing or stopping an adversary from gaining access or propagating control between microservices, implementation of authentication and authorization of microservices and users, implementation of isolation between microservices and mitigation of network -related issues. Some of these correlates with former studies such as popularity of authentication and authorization issues and the role of security monitoring, this study includes more details regarding issues such as network-related issues that are less developed and mostly absent in previous studies.

The content of the thesis is divided to issues and solutions that are relevant to stopping and monitoring adversaries from gaining access to microservices and propagating them, network related issues and solution / mitigation methods, isolation related issues, and prevention of an adversary gaining access over the microservice and propagation. Former consisting of methods preventing propagation such as perimeter defenses, diversification, security monitoring and authentication and authorization of services and users.

2. background

Necessary background consists of defining microservices and security issues and solution to those security issues.

2.1 microservices

There are several different ways to define microservices. Microservice function as isolated units for executing code, they are autonomous that is they can be deployed, destroyed, moved, and duplicate independently, and they are fine-grained that is microservice is responsible for its own task (Monteiro, et al., 2018). (Fowler & Lewis (2022) as cited by (Li et al., 2020)) defines microservice architecture (MSA) that is made up of small applications / services that are microservices. Microservices executed their own processes and communicate with another microservice via network. This allows a more flexible way to implement software, since it allows multiple different teams to work on their own part of the software without necessarily needing to worry about the progress of other teams. Separating different functionalities to a separate implementation means that functionality can be tested and maintained on its own (Pooyan, Claus, Nabor, Lewis, & Tilkov, 2018).

2.2 security challenges

There are several security issues that are raised by previous studies. These issues are tied to how microservice communicate with one and another and networking, isolation of microservices. Former can get rather complex since microservice require more complex communication model (Li, et al., 2020). They refer to different ways an adversary could use to prevent normal functioning of microservice or services and / or exploiting microservice system. An adversary can gain access to microservices through bug or vulnerabilities that can be exploited to gain control over the service and propagating it further. Security in microservice architecture is usually referred to as a system's ability to protect data and information from unauthorized use and providing access to authorized users (Li, et al., 2020). Microservices are also used more in open-environments that provide ways for adversaries to gain access to the microservice.

Solutions and mitigation methods follows this diversity, due to the nature of microservices and security. Former due to how modifiable microservices are, and later due to how hard it is to find any solution or mitigation method with any kind of finality to it. There are issues that are hard to solve such as Denial of Service (DOS) but they can be mitigated. It is also possible solutions themselves might not be good enough anymore due to changes in technology, or it is too difficult to implement from performance or management point of view. Solution and mitigation methods would refer here to methods that can be found in literature to address some issues.

This study has issues brought up by microservice being connected to internet that could allow adversaries launching denial of service attacks against service or services and vulnerabilities that virtualization implementation of microservice uses such as container.

3. Research Method

This study sets out to find what sort of security issues can be found in academic literature and what sort of solutions or mitigation methods they use or propose to use to address these issues. There are two research question.

What sort of security issues can be found in literature and what sort of solutions are proposed to address them.

Microservice security has been researched to certain extent. There is no previous research that employees systematic review process and among these searches the keywords microservice and security have been used by (Nuha, Nour, & Roger, 2016). This study recommends (”microservices” OR ”micro service” OR ”micro-services”) AND security” keywords. I selected English-only journals due to lack of proficiency in other languages and wide-availability of sources in English.

I searched academic databases, this study used academic databases Scopus, ACM, IEEE, ScienceDirect, Wiley Online library.

I followed the systematic literature review procedure by (Kitchenham, 2004). Document types are restricted to journals and conference proceedings to include current scientific literature that definite to be between years 2016 - 2021. I included documents if the source’s title, abstract, introduction and conclusion if they held information related to security issues of microservice and / or possible solution for those issues.

The query resulted into 1153 documents. I filtered through articles by name, type, abstract keyword and finally by introduction, abstract and conclusion. The name and document type-based selection resulted in inclusion of 500 documents. I included documents that are journals and conference proceedings, and documents that included security and / or microservice related string in title. Abstract based inclusion resulted in inclusion of 47 documents and introduction- and conclusion - based inclusion resulted in final of 26 documents. In these inclusions, the content of the text needed to include issues touch upon microservices and their security.

Finally, I snowballed several sources, in total of 39, that however might have some validity issues due to difference slightly different topics addressed such container security that have implications to microservice security but strictly speaking are their own topic. This is however also possible in the primary content selected. Inclusion can be justified due to close relationship between the topics that can be seen in the material. Still, they have implications to security issues of microservices and are cited by accepted sources.

4. Security Challenges and Solutions

Overall security issues in the literature concentrates on how to prevent adversaries gaining control over a microservice and propagating its control over the rest of the services or otherwise causing damage to the microservices. The former includes issues such as an adversary exploiting bugs to try to control microservices or underlying implementation. Later consists of traditional network attacks such as Denial of Service (DOS). There are several possible solution and mitigation methods in literature that offer different ways to deal with adversaries from gaining access to multiple microservices. These consists of solutions such as perimeter defenses, security monitoring, moving target defenses (MTD) and use of diversification on microservice and microservice-centric approaches such as authentication and authorization and isolation-based approaches.

Table 1 The Distribution of Tables

Topic	Number of documents
Perimeter Defense	7
Security Monitor	9
Diversification	5
Service Authentication and Authorization	9
Propagation methods	6
User authentication and authorization	6
Network - related issues	10
Isolation	11

Overall, there does not seem to be strong security issues in literature. Authentication and authorization issues overall are quite popular, as seen in table 1. The number of studies is still large in isolation and network-related issues, followed by service authentication and security monitoring. Both isolation and network-based issues are surprising since topics aren’t really addressed by (Li et al., 2020). It is noteworthy that this only count the number of studies that refer to issues. This study starts with security issues addressing microservices systems, authentication and authorization issues and stopping with isolation and network-related issues.

4.1 perimeter defense

Several sources raise the role of perimeter defense and the filtering mechanism for addressing external attacks. Jander, Braubach, & Pokahr (2018) deems perimeter defense to be the most important aspect of the network security. It is addressed in several ways, such as usage of intrusion detection system (IDS) and Firewalls.

There are some existing network and system admin tools such as Virtual Private Networks (VPN), Demilitarized Zone (DMZ) and Firewall that can be used to solve existing network security problems (Bánáti, Kail, Karóczkai, & Kozlovszky, 2018, p. 1). Out of these solutions, only firewall is addressed in the rest of the literature. Márquez & Astudillo (2019) note that microservice system protected by firewall are secure from attacks such as Man-In-The-Middle (MiTM) since outside tampering can be stopped. It however can't be relied on in open environment (p. 6).

IDS is addressed by (Nehme, Jesus, Mahbub & Abdallah, 2018 ; Sun, Nanda & Jaeger, 2015 ; Otterstad & Yagyrina, 2017) and addressed in more limited manner by (Li, Jin, Zou & Yuan, 2020). Sun et al.(2015) notes that IDS on the network edges can address ingoing and outgoing traffic but note that it is incapable of addressing internal threats. The capability of IDS is not shared by other sources. Nehme et al. (2018) note that there are some attacks such as SQL injections cannot be mitigated with firewalls available to microservices. The need or availability of such firewalls is not established in later or earlier studies. They also criticize it due to the need for adjust it for application and on the level of traffic (p. 5). It can also lead to problems with the prioritization since prioritization perimeter leads to a system in which it is hard gain access to but once it is gained it is relatively easy for an adversary to propagate their control or cause harm inside the system (Jander et al., 2018). Li, Ji et al. (2020, p. 1) note that detection methods based on packet signature or network behavior are not necessarily enough to mitigate outside interference such as low-rate dos attack where attacks are still possible if the rate of calls remains low enough not to be noticed by IDS. Perimeter defenses seem to be able to mitigate against outside adversaries, it is however good to point out that none of the sources really addressed how effective in different contexts.

4.2 security monitoring

Security monitoring is raised in literature raises for dealing with an adversary gaining access to a microservice and propagating its control to other microservices. Sun et al. (2015) articulates the need of microservice systems for security monitoring as they are often deployed in a cloud application where control over all services is not possible and there is no way to get a global view on the entire application (p. 2). They note that network complexity with large amount of services lead to difficulties to monitor entire system and single service being capable of taking down the entire system. Security monitoring proposed for dealing with the issues by introducing a service which purpose is to monitor actions inside the system. Outside finding out adversarial actions between microservice, security monitoring is also important for detection of downed services and ensuring availability of services (Nehme et al., 2018, p. 5). It can also be used to ensure quick defect-detection and fixing of issues by automated patching (Otterstad & Yagyrina, 2017, pp. 5 - 6).

The usual security monitor solution includes some separate service that is responsible for monitoring internal network and communication between services envisioned security monitoring for logging network events, protecting public facing microservices and protecting from attacks such as SQL attacks (Sun et al. 2015, p. 4). Their proposed solutions combine Software Defined Network (SDN) and Flow Tap API. Former for scanning network pacts and control forwarding and leveraging it for later that is used for separating where to place security monitor from network flows, allowing security monitor to be placed outside network edges that would mitigate monitoring and policy enforcement problems. Later, establishes a contract between application and infrastructure regarding how to deliver network events. Another security monitoring solution is provided by Torkura, Sukmana & Meinel (2017) who propose security monitoring with security gateway for continuous security assessment and monitoring of microservices. The solution consists of a security gateway that is responsible for enforcing security polices and is supported by dynamic document store and security health endpoints for storing and detecting vulnerabilities. Security gateway would allow definition and implementation of fine-grained security policies.

Several sources note how usage of SDN, virtual networks would security monitoring. SDN are heavily used in security monitoring solution in literature. SND allows control over authenticated of underlying containers, firewalls and access mechanism and offers forwarding mechanism (Ranjbar, Komu, Salmela & Aura, 2017, p. 3). SDN seems to be able to solve several security issues, but there are several issues found in it in literature that could have implications to microservices. Yu, Jin, Zhang, & Zheng (2018) points out some problems with SDN such as vulnerabilities with policy controller SDNs controller module that an adversary could gain control over controller module, thus gaining full information of network (p. 12). They note that testing and verification of 3rd party application like controller modules are difficult. SDN allows monitoring network flow inside, but it cannot guard microservices from all potential security problems, especially with large number services. They suggest a Framework consists of authentication-, authorization-, trust database-, policy database- and Monitoring and Evaluation modules. Policy module defines implementation of global network policy and Monitoring and Evaluation module review and evaluates existing relationships between modules and network application. Despite this, they do recommend SDN for network related issues, since it allows centralizing access control, their mechanisms, and firewalls. There is also some concern over how to define and implement security policies needed for monitoring.

Preuveneers & Joosen (2019) note that there is a need for unified microservice offering, delegation of access control and granting permissions. That has quite an important role in microservice security as size of the system increase in complexity in microservices such as large number of microservices added (p.12). They view SDN network policies as a possible solution but lacking in formalized verification or semantics. Their solution would help SDN control layer to authenticate and authorizes permissions to counter manipulation of network resource. Gerking & Schubert (2019) found information flow specification to be lacking and insecure, their suggestion is to improve better specification of security policies would be needed to ensure that secret information can't be deducted through public information in insecure network.

4.3 diversification

Literature notes that one potential way for adversaries to propagate control inside a microservice system is to exploit homogenous implementation of microservices. Microservice Architectures are vulnerable due to often homogenous implementation of microservices lead to vulnerabilities that can be taken advantages in multi-step attacks (Torkura, Sukmana, Anne, Cheng & Meinel, 2018). One possible vulnerability is the identical base images used by microservice instances make them vulnerable to multi-step attacks (p.1). The lateral movement between microservices is easy once collaborating microservice has been compromised, albeit this is due to lack of authentication and authorization in inter-instance communication (p.3).

There is also a security problem with code reuse. Shared Code and libraries can introduce lock-in problems that can escalate further with the need for propagation of patches (Márquez & Astudillo, 2019, p. 6).

Others note the possibility of using diversification and moving target defenses. The attacks mitigate by methods such as changing implementation languages that microservices use. Jin, Li, Zou, & Yuan (2019) proposes a framework which is responsible for automatic evaluation and optimization of MTD techniques, specifically IP-shuffling and live migration. They note some problems with existing defense mechanism that might lead to static defense, defenses are not necessarily capable of reflecting changes in the system leading to *effective drift problems* (p.1). They found that diversification could cause this and seek to mitigate it with their DSEOM framework that utilizes automated MTD techniques to solve effective drifting problem. Their solution is to use MTD alongside diversification engine and risk analysis. Diversification happens on runtime, leveraging both model and template based automatic code generation techniques. While they found their solution to eliminate problem with homogenous systems and decreasing vulnerabilities. They note that there are several limitations to it being applicability only to Open API compatible microservices and Swagger Codegen that is used for transformation holds only 30 programming languages/frameworks (p. 6).

The need for diversification is criticized by (Otterstad & Yagyrina, 2017, pp. 5 - 6). They note that microservice are usually diversified due to their functionalities. Despite this, they do admit that there might still be common vulnerabilities such as vulnerable libraries that could be exploited.

There is a unique solution that is not included in the rest of the literature. (Suneja, Kanso & Isci, 2019) propose a way how to perform security scanning of microservice by other microservices. This requires isolation between microservices to be broken to extend, but also allows security analysis of core-microservice by utility services. They note that it could be useful for monitoring microservices system.

4.4 service authentication and authorization

One common topic for security solution is how to introduce more effective authentication and authorization methods. Authentication and authorization do have a large role in securing microservices (Nehme, Jesus, Mahbud, & Abdallah, 2018, p. 3) authentication and authorization essential for securing service. Motivations for authentication and authorization of service are preventing propagation of adversaries control between services and enforcement of security policies. Propagation of adversaries' control inside system is made possible due to trust-based microservice communication and trust among inter-communications, assumption of common security trust domain can be used to propagate to other services ((Esposito, Castiglione & Choo, 2016) cited by Torkura et al., 2017, p. 3). Dynamic nature of microservices and ephemeral nature of resources such as dynamic deployment can be a source of problems due to changes in IP-address, port numbers and service endpoints (Torkura, Sukmana, & Meinel, 2017). Weak authentication and authorization can lead to services being maliciously influenced by adversaries (Yu, Jin, Zhang, & Zheng, 2018, p. 10) and weekly authenticated microservices are more vulnerable to DoS attacks (Li, Jin, Zou, & Yuan, 2020, p. 3). Netflix compromise in 2015 would be an example of such an attack (Nehme et al., 2018, p. 4 ; Sun et al., 2015, p. 1).

There are other motivations for authentication and authorization. Services need to be authenticated for reliable logging and as a basis for authorization and enforcement of security policies (Walsh & Manferdelli, 2017, p. 1). They note situation where authentication is not necessarily needed, such as communications between services happens with relatively static microservices on a trusted network.

Martin, Raponi, Combe & Pietro (2018) recommends stopping trust-based communication between containers / services (p. 6). This is something that rest of literature agree. Otterstad & Yagyrina (2017) note the role of external connection and input validation inside microservices to be important for dealing with adversaries gaining access inside the system, implying the need for further security (pp. 2 – 3).

However, there remains some conflict when it comes to picking implementation. Literature includes TLS and HTTP- based approaches for service authentication. Usage Mutual Transport Layer Security (MTLS) with self-hosted Public Key Infrastructure (PKI) for authentication is noted to be emerging practice by (Yarygina & Bagge, 2018, p. 16). Microservice gets deployed with certificates that are used for authentication of service and for controlling what microservice can call or be called by.

They give a generic model where services use MTLS for communication between services and TLS connection with Certificate Authority after it has generated a self-signed root and shared secret and provision them to new services after cryptographic hashing. They use Docker swarm and Netflix internal microservice network as promising solution to microservice authentication. They note MTLS does not address service authorization by itself but can be used to support fine-grained authorization by using PKI and typing based certificates for authorization. They recommend creating separate signing certificates for each microservice type.

HTTP digest-based authentication is recommended by (Yu, Jin, Zhang, & Zheng, 2018, p. 15) as an ideal solution for inter-service communication would consist of HTTP digest authentication that allows user definite encryption algorithm with the public key infrastructure (PKI) being responsible for authentication and authorization of services. They note alternative in using OAuth2 with Spring Cloud Security, later responsible for protecting user information.

Walsh & Manferdelli (2017) note several authentication methods based on certificates, noting advantages of mutual TLS over HTTP due to lack of need to for endpoints to store long-term secrets (pp. 1 – 2).

They find that using HTTPS to authenticate endpoint and HMAC with secret PKI keys other is ill-suited for microservices authentication due to vulnerabilities tied to PKI and possibly untrustworthy third parties and API key protocols due to variance of how tightly coupled with specific REST-oriented protocols are (p.1). PKI is criticized due to rate limits and API key management, long-lived secrets in durable storage service would require extra authentication mechanism. This criticism is not included in other sources, and most solution tend to use PKI based solutions.

PKI is however also criticized by (Jander, Braubach, & Pokahr, 2018, p. 2) alongside certificates for needing to use more resources for validation and checking than solution that they are looking for due to trust chains and related certificate authentication can get costly. They find that the most common TLS authentication is not satisfactory due to lack of authentication on client side, server side based on host name, and while customization is possible it does require application-level implementation (p. 2 – 3). Authentication extensions libraries could be used to address these issues, but they lack support. Possible solutions would be: Extending TLS with in-band authentication based on tokens, and microservices frameworks such as the Vert.x that offer event bus for communication. Both ultimately suffer from same the problem of TLS that is more complex solutions and require specialized knowledge and leads to added complexity. They offer a solution concept which implements protocol responsible to securing inter-service communication and authenticating services based on ephemeral keys that represent microservice roles (p.4 – 7).

4.5 propagation methods

Propagation methods are relevant due to the need for a secure way to transfer and support authentication and authorization information between services. Tokens is the most common method in studies, authentication and authorization studies either recommend its usage or uses them as a part of the solution.

In order to communicate with one another, microservices need some standard that support authentication and authorization. Token-based approach in that they allow flexible, single-time identification, this allows securing user information more effectively since microservices themselves do not gain access to user information (Bánáti, Kail, Karóczkai & Kozlovszky, 2018, p. 1). Yarygina & Bagge (2018) note microservices need to be able to know the state of user's authentication and authorization state, this can mean multiple service depending on the size of operation chain. Security tokens consisting of tokens generated by the server after successful user authentication by separate server and replaces user's authentication information for some limited time, such as HTTP cookies (pp. 6 - 7).

Interestingly, tokens seem somewhat worse in terms of security. Richter et al. (2018) compare different authentication and authorization methods from which they selected token-based authentication and authorization and find it preferable to certificates due to simplicity of implementation such as a lack of managing cryptographic keys while considering later to be a more secure option(p.8). Tokens allow more flexible implementation and as such are seen more favorably. JWT tokens would have several advantages, such as allowing delegation of access control and permission granting for proving unified microservice offering for microservice federation. (Preuveneers & Joosen, 2019, pp. 1 - 2). There is only one option to tokens presented in literature.

Bánáti et al. (2018) notes Security Assertion Markup Language (SAML) for exchanging authentication and authorization information as a possible candidate. However, they view SAML as suboptimal due to more resources heavy in sending and receiving, in size and complexity it adds, nor supports removing or adding information that tokens have(p.1).

Usage of tokens is tied to popularity of external services such as OpenID Connect. Yarygina & Bagge (2018) note that token-based authentication has gained popularity due to widespread adoption of OpenID connect (pp. 6 - 7). This is supported by (Nehme et al., 2018, p. 4 ; Bánáti et al., 2018, p. 1). The possible security issues brought up by separate server is not addressed in literature outside single criticism. The separate authentication server is criticized by (Jander, Braubach, & Pokahr, 2018, p. 1) who note that separate server does bring a problem with defense-in-depth due to separate and possibly centralized management of authorization.

Reverse security tokens based on (Doerfeld, 2015 cited by Yarygina & Bagge (2018)) for user to service based authentication. Tokens is generated after user authenticated and used only in internally and set with expiration time. Microservices responsible for resource validates tokens based on corresponding public key. Yarygina & Bagge (2018) note that this is a recent and academically undocumented type of token usage in the industry. They view this approach having potential since it can be integrated with tool support such as Oauth and OpenID connect. It does introduce extra management and increased vulnerability. Services need to have synchronized clocks, which they note can be handled using NTP -protocol.

The need to use TLS for communication between services to stop interception of messages and exploitation of token and service responsible for issuing tokens need to be able to safeguard their private keys from attackers (p.6).

4.6 user authentication & authorization

User authentication and authorization is addressed by few sources. Primarily topic seems to be where authentication and authorization of users takes place.

Most microservice systems communication happens between service and the external API and the user ((Newman. 2017) cited by (Jander, Braubach, & Pokahr, 2018, p. 1)). According to (Yarygina & Bagge, 2018, pp. 6 - 7) the common user authentication starts when the gateway validates user credentials and gives the user a token that is used by the resource service to authorize the user to use the service.

User authentication to be best left for a separate authorization server (Currie, 2017 ; Doerfeld, 2015). Usage of authentication / authorization server for user authentication and authorization (Yarygina & Bagge, 2018, p. 7 ; He & Yang, 2017, s. 8) is viewed as preferable implementation. (Doerfeld, 2015 ; Yu et al., 2018, pp. 10, 15) would use OpenID Connect and OAuth are responsible for authorization of users when user tokens are validated after authentication. Yarygina & Bagge (2018) note usage of Role-Based Access Control (RBAC) and Attribute Based Access Control (ABAC) for user authorization, it could also be incorporated into JWT tokens and later should be used for fine-grained resource authorization (p. 7).

Preuveneers & Joosen (2019) address user authorization with Oauth-based User-Managed Access (UMA) 2.0 access management protocol that allows management of access that they use to authorize user in federalized environment. They note that using dynamic granular access control solution on top of RedHat Keyclock 5.0 Identity and Access Management.

There is however some criticism towards centralized management of authorization that separate service would be by (Jander et al., 2018, p. 1 ; Bánáti et al., 2018, p. 1). Similarly, to service authentication and authorization and propagation usage of separate server is the usual way to implement user authentication and authorization.

4.7 network-related issues

Literature identified several traditional networking security issues such as Address Resolution Protocol (ARP) spoofing, Man in The Middle (MiTM) attacks and denial of service (DoS) are represented in the literature. ARP spoofing is addressed by (Bui, 2014 ; Martin et al., 2018 ; Chelladhurai et al., 2016) and is noted by (Yu, Jin, Zhang, & Zheng, 2018).

ARP poisoning attacks are noted to happen only between docker containers due to shared, and non-filtering network bridge that allows forging messaging between containers. There are several solutions offered for this such as (Bui, 2014, p. 6 ;Yu et al., 2018, p. 15) who note the possibility of adding manual filtering. Yu et al. (2018) note adoption of virtual network such as Software Defined Network(SDN) for monitoring network flows for preventing ARP spoofing as a possibility (p. 15). Martin et al. (2018) recommend forbidding network communication between containers (p.36). It is noteworthy that ARP spoofing is mentioned only in earlier literature pre-2017, and later studies references earlier works.

MitM is addressed by (Torkura K. A. et al., 2018 ; Ranjbar et al., 2017 ; Márquez & Astudillo, 2019 ; Yu et al., 2018).

Torkura et al. (2017) note that in inter-microservice trust-based communication has MiTM vulnerabilities that allow exploitation such as session and token hijacking that could lead to issues with inter-service communication and orchestration(p. 6). They claim that vulnerability exists in insecure networks such as open cloud-environment which have difficulties to recognize services due to dynamic deployment of microservices, dynamic IP-porting and -service endpoint introduction. Their proposed solution would be to automate implementation of security policies to find out and stop deployment of microservices with vulnerabilities. Márquez & Astudillo (2019) notes that microservices are vulnerable to MiTM attacks in open environments. It is possible to secure microservices by entering them into closed systems behind firewalls (p.6). They raise a possibility of using encryption-based approaches to mitigate MiTM attacks. It is noteworthy that possibility of using SDN is not really addressed directly, however it is usually used as a part of proposed solution when discussing security monitoring (pp. 1, 6) .

Dos attacks are addressed by (Márquez & Astudillo, 2019, p. 6 ; Bui, 2014, p. 5 ; Chelladhurai, Raj & Kumar, 2016, pp. 3-4 ; Ranjbar et al., 2017, p. 6 ; Li et al., 2020, p. 1). Márquez & Astudillo (2019) note that DoS to become a problem when number of service and external endpoints increase, making it possible to create expensive API calls across the system. Mitigation methods proposed in literature against dos attacks are based on controlling available resources concentrates on container implementations. Self-healing is addressed in limited manner by (Márquez & Astudillo, 2019, p. 6). They note that there is a lack of and a need to implement self-healing microservice- systems to ensure availability.

Bringing down allocated resources to containers is recommended by (Bui, 2014, p. 5 ; Chelladhurai et al., 2016, pp. 2 - 3). Later created a pragmatic method to address DoS attacks by limiting container resources. There are some issues with this solution. Limiting container resources or scaling targeted microservice instances can have a problem with low-rate attacks due to difficulty of detection of under threshold intrusions and scaling due to container-based services possibly having further problem with resources competition resulting from multiple microservices sharing the same host (Li, Jin et al., 2020, p. 1). Scaling can be unacceptable since a service might still be needed and some services are more depended upon by other microservice thus making system vulnerable (p.2). Their proposed method would solve this by dividing microservices requests under DoS attack to whitelisted- and unknown requests, with microservice promising whitelisted requests and uses rest of the resources for unknown requests. Their solution can be improved with filtering mechanism in complex situation where attacker uses different kind of attacks (p.9). They test the mechanism and find their solution is effective for whitelisted source but fail to meet their standards for unknown requests.

Yu et al. (2018) would employ already existing signature-, automata- or simulation-based methods (p. 15). There are criticisms of later methods in literature. Li et al. (2020) notes that exiting methods against low-rate DoS attacks such as isolation- and limitation mechanism, while successful in virtual machine (VM) environment, are not shown to be applicable in container-based environment (p. 1).

There is a solution that seeks to address MiTM and DoS which uses SynAPTIC and SDN to address MiTM and DoS attacks by authenticating underlying containers with HIP-based mechanism (Ranjbar, Komu, Salmela, & Aura, 2017, p. 6).

4.8 isolation

Isolation refers here to security issues brought by implementation of microservice, and its underlying implementation be it container or VM. In practice, security issues raised from bugs or vulnerabilities of containers implementations and resource separations underlying the microservices. Both can be used by malicious actor to attacks against services.

Literature notes several threats such as possible kernel exploits and escape from containers (Yu et al., 2018, pp. 5 – 6 ; Martin et al., 2018, p. 32 ; Casalicchio & Iannucci, 2020, p. 6). Escape attacks can be followed by privileges escalation or blocking rest of the containers on the host. Jian &

Chen (2017) found that it is possible to perform escape attack by changing namespace and by modifying shared memory(pp. 3 – 5). Gao, Gu, Pendarakis & Wang (2017) found information leakages based due to incomplete system resource definition that could be exploited in shared-environment, they propose a new namespace for finding out possible power-based attacks. It is unclear how relevant these are however since existing security features are seen as good enough for securing isolation (Martin et al., 2018 ; Manu, Patel, Akhtar, Agrawal & Murthy (2016). Literature seems to somewhat conflicted on this topic. Yu et al. (2018) note that some system resources can be difficult to partition effectively and not enough for security purposes (p. 14). Similarly, Li et al. (2020) notes poor performance isolation between containers leading to resources competition, implying that there might be some issues (p.2). Duarte & Antunes (2018) used static code analysis to find or avoid vulnerabilities in Docker and did not find it to be effective outside contributing to security by supporting software maintainability and quality (p. 9). They found main causes of docker vulnerabilities to be unprotected resources and incorrect permission management that are usually rooted to efficiency- and security trade-offs (p.9).

Fetzer (2016) concentrates confidentiality and integrity of data in microservice alongside correct execution of its processes and how to guard it from the host (pp. 2 – 3). His microservice – based approach employs Software Guard Extension (SGX) to provide protection from these threats. He notes that microservice is still vulnerable against network API attacks that could be to mitigated / solve them with compiler-based solution. Yarygina & Bagge (2018) note that SGX or Trusted Platform Module (TPM) could be used to protect authentication data such as long-time credentials, they note an example of this with protection of authentication and authorization data(p. 6). SGX solution has some issues since (Tian, Choi, Hernandez, Traynor, & Butler, 2019, p. 1) note SGX application security may hold vulnerabilities if application is placed within enclaves such as increased size of Trusted Computation Base (TCB), vulnerabilities with application and libraries such as Open SSL that are vulnerable cache-based side channel attack.

Some sources include the possibility of adding new layers to provide security. Martin et al. (2018) note that container has more vulnerability compared to VM. They note a possible future solution would be unikernel implementation that are promising for security purposes for solving kernel related problems. Yu et al. (2018) ideal solution is adding layers is for addressing isolation issues would run containers on top of VM, containers do not gain direct access having to go through VM and hypervisor. They do however note that security also requires implementation of monitoring, fault tolerance and other mechanism. They name SELinux as ideal for it due to access management of processes, permissions and file, available security policies alongside integration with containers. There is some support for this (Manu, Patel, Akhtar, Agrawal, & Murthy, 2016, p. 6) note that containers run on VM, and hypervisor are more secure.

5. discussion

The study found there are several security issues and solution to these issues in the literature. Overall authentication and authorization issues together would make it the largest topic with 21 documents addressing it, the studies addressing means to address security monitoring, perimeter defense and diversification totals to 15, and isolation- and network-related issues both total 9. This can be somewhat misleading, however, since a large amount of literature addressed multiple issues.

The main concern is how to prevent adversaries from gaining access to a microservice and propagating control to other microservices and there are a lot of commonalities between solutions in their respective topics, authentication and authorization issues seem to be a bit of an outlier in that there is some amount of disagreement on usage of PKI for service authentication and authorization in terms of performance and security.

The results of this study builds upon previous studies, while a lot of it is similar to previous studies. Security issues remain quite consistent when compared to previous studies, with the exceptions of isolation and networking issues when compared to (Li, et al., 2020). The exact reason for this is not quite clear. Partially this could be due to distinctions between different concepts of isolations. Isolation issues differ between more container / implementation centered view and isolations as an abstract feature of microservice. The issues in this study tied to isolation refer partially to both, as the issues raised would break isolation in later too. Networking issues are mostly left unaddressed, which is somewhat odd since issues such as MiTM and ARP spoofing are still considered to be a major issues in earlier studies. Still, it certainly could be a matter of change in how these issues are understood. This could explain why issues such as DOS aren't really addressed. It is possible that these issues are seen affecting microservice architectures as a whole, or there are adequate mitigation methods that are seen as more relevant. It is also possible that technologies such as SDN, alongside security monitoring, have made these sorts of issues effectively obsolete. Later seems realistic, however is not really addressed in the literature and could be a topic for further research. This could be a possible constraint to this study, as it is entirely based on existing literature that can be found in academic databases available and searchable with used tags. Some studies were snowballed into study which however were not found in initial search.

A topic that is not raised is container fusion addressed by (Suneja, Kanso, & Isci, 2019) and how it might have security implications to microservices and security monitoring. Having services that are responsible for security checking other service could prove useful.

The popularity of authentication and authorization is noted in previous study by (Nuha, Nour, & Roger, 2016) and security monitoring is noted to be a raising concern for microservice by (Li, et al., 2020). Network- and isolation-related issues are not raised to a similar role despite being popular topics in literature, and the most recent in sources. It is possible this is due to these issues consisting of topics that are already addressed somewhere else, such dos- attacks that are common in open-environments such as cloud environments and already have existing mitigation methods. Similarly, isolation issues could have been addressed in other ways. The latter seems somewhat unlikely, since vulnerabilities in implementations play a role in these issues. There are also issues that seem to disappear from later sources. An example of this would be ARP spoofing that seems to have disappeared from literature completely after 2016. Later sources that mention it refer to older sources. Reason for absence however is not addressed in literature. It is noteworthy that near similar issues such as MiTM are still addressed in later sources. This study for the most part found similar issues and solutions that are noted in previous studies, however it also does include some that aren't mentioned in others, such as isolation and networking-related issues. It also includes a more detailed approach to issues and solutions than previous have in areas such as authentication and authorization. This study does include all topic brought up by (Li, et al., 2020) except for isolation related issues.

6. conclusions

This study found there are several security issues and mitigation methods regarding microservices. Security issues concern with microservice is security monitoring and propagation of adversaries control between microservices, authentication and authorization of microservices, and isolation and network-related issues. There is similar diversity in solutions and / mitigation methods proposed. This is based on the number of previous studies that were found. Some issues are not noted in previous studies like this, such as isolation- and network-related issues and others are more detailed than in previous studies such as authentication and authorization issues. The primary security issues are related to how to prevent adversaries gaining access to microservices and propagating their control to other microservices.

References

Bánáti, A., Kail, E., Karóczkai, K., & Kozlovszky, M. (2018). Authentication and authorization orchestrator for microservice-based software architectures. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. pp. 1180-1184). Opatija, Croatia : IEEE. doi:10.23919/MIPRO.2018.8400214

Bui, T. (2014). Analysis of Docker Security. *Aalto University T-110.5291 Seminar on Network Security.*

Casalicchio, E., & Iannucci, S. (2020). The state-of-the-art in container technologies: Application, orchestration and security. *Concurrency and Computation: Practice and Experience Volume 32, Issue 17*.

Chelladhurai, J., Raj Chelliah, P., & Kumar, S. (2016). Securing Docker Containers from Denial of Service (DoS) Attacks. *2016 IEEE International Conference on Services Computing (SCC)* (pp. 856 - 859). San Francisco, CA, USA: IEEE.

Doerfeld, B. (2015, May 14). *Nordic APIS*. Retrieved from Nordic APIS: https://nordicapis.com/how-to-control-user-identity-within-microservices/

Duarte, A., & Antunes, N. (2018). An Empirical Study of Docker Vulnerabilities and of Static Code Analysis Applicability. *2018 Eighth Latin-American Symposium on Dependable Computing (LADC)* (pp. 27 - 36). Foz do Iguacu, Brazil: IEEE.

Esposito, C., Castiglione, A., & Choo, K.-K. R. (2016). Challenges in Delivering Software in the Cloud as Microservice. *IEEE Cloud Computing, 3*(5), 10 - 14. doi:10.1109/MCC.2016.105

Fetzer, C. (2016). Building Critical Applications Using Microservices. *IEEE Security & Privacy, vol. 14, no. 6*, 86-89.

Fowler, M., & Lewis, J. (2022, 05 20). *Microservices*. Retrieved from martinfowler.com: https://martinfowler.com/articles/microservices.html

Gao, X., Gu, Z., Kayaalp, M., Pendarakis, D., & Wang, H. (2017). ContainerLeaks: Emerging Security Threats of Information Leakages in Container Clouds. *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (pp. 237-248). Denver, CO, USA: IEEE.

Gerking, C., & Schubert, D. (2019). Component-Based Refinement and Verification of Information-Flow Security Policies for Cyber-Physical Microservice

Architectures. *2019 IEEE International Conference on Software Architecture* (pp. 61 - 70). Hamburg: IEEE.

He, X., & Yang, X. (2017). Authentication and Authorization of End User in Microservice Architecture. *Journal of Physics: Conference Series, Volume 910,*. Guilin, China: IOP Publishing Ltd.

Jander, K., Braubach, L., & Pokahr, A. (2018). Defense-in-depth and Role Authentication for Microservice Systems. *Procedia Computer Science, 130*(The 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops), 456 - 463. doi:10.1016/j.procs.2018.04.047

Jian, Z., & Chen, L. (2017). A Defense Method against Docker Escape Attack. *ICCSP '17: Proceedings of the 2017 International Conference on Cryptography, Security and Privacy* (pp. 142-146). New York NY United States: Association for Computing Machinery.

Jin, H., Li, Z., Zou, D., & Yuan, B. (2019). Cloud, DSEOM: A Framework for Dynamic Security Evaluation and Optimization of MTD in Container-Based. *IEEE Transactions on Dependable and Secure Computing (Volume: 18, Issue: 3, May-June 1 2021)* (pp. 1125 - 1136). Hai Jin; Zhi Li; Deqing Zou; Bin Yuan: IEEE.

Kitchenham, B. (2004). *Procedures for Performing.* Eversleigh: Keele University Technical Report; Empirical Software Engineering.

Li, S., Zhang, H., Jia, Z., Zhong, C., Zhang, C., & Shan, Z. (2020). Understanding and addressing quality attributes of microservices Achitecture: A Systematic literature review. *Information and Software Technology, Information and Software Technology 131*(Elsevier B.V). doi:10.1016/j.infsof.2020.106449

Li, Z., Jin, H., Zou, D., & Yuan, B. (2020). Exploring New Opportunities to Defeat Low-Rate DDoS Attack in Container-Based Cloud Environment. *IEEE Transactions on Parallel and Distributed Systems (Volume: 31, Issue: 3, March 1 2020)*, 695 - 706.

Mannino, J. (n.d.). *Security In Microservice World.*

Manu, A., Patel, J., Akhtar, S., Agrawal, V. K., & Murthy, K. (2016). Docker container security via heuristics-based multilateral security-conceptual and pragmatic study. *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)* (pp. 1-14). Nagercoil, India: IEEE.

Márquez, G., & Astudillo, H. (2019). Identifying availability tactics to support security architectural design of microservice-based systems. *Proceedings of the 13th European Conference on Software Architecture - ECSA '19 - volume 2* (p. 2019). New York, New York, USA: ACM Press.

Martin, A., Raponi, S., Combe, T., & Pietro, R. D. (2018). Docker ecosystem – Vulnerability Analysis. *Computer Communications 122*, 30-43.

Monteiro, L., Almeida, W., Hazin, R. R., Lima, A., Gomes e Silva, S., & Ferraz, F. (2018). A Survey on Microservice Security–Trends in Architecture, Privacy and

Standardization on Cloud Computing Environments. *International Journal on Advances in Security, 11*(3 & 4), 201 - 213.

Nehme, A., Jesus, V., Mahbud, K., & Abdallah, A. (2018). Securing Microservices. *IEEE Computer Society*, 42 - 49.

Newman, S. (2017, July 7). *https://www.container-solutions.com/about-us*. Retrieved from https://blog.container-solutions.com/microservice-insecurity: https://blog.container-solutions.com/microservice-insecurity

Nuha, A., Nour, A., & Roger, E. (2016). A Systematic Mapping Study in Microservice Achitecture. *2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA).* Macau, China: IEEE. doi:10.1109/SOCA.2016.15

Otterstad, C., & Yagyrina, T. (2017). Low-Level Exploitation Mitigation by Diverse Microservices. *European Conference on Service-Oriented and Cloud Computing* (pp. 49-56). Springer, Cham.

Pereia-Vale, Anelis, Marquez, G., Astudillo, H., & Fernandez, E. (2019). Security Mechanism Used in Microservices-based systems: A Systematic Mapping. *45th Latin American Computing Conference* (pp. 45 - 55). Panama City: Institute of Electrical and Electronics Engineers Inc.

Pooyan, J., Claus, P., Nabor, M., Lewis, J., & Tilkov, S. (2018). Microservices The Journey So Far and Challenges Ahead. *IEEE Software, 35*(May / June 2018), 24-35. doi:10.1109/MS.2018.2141039

Preuveneers, D., & Joosen, W. (2019). Towards Multi-party Policy-based Access Control in Federations of Cloud and Edge Microservices. *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (pp. 29-38). Stockholm, Sweden: IEEE.

Ranjbar, A., Komu, M., Salmela, P., & Aura, T. (2017). SynAPTIC: Secure and Persistent Connectivity for Containers. *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)* (pp. 262-267). Madrid Spain: IEEE Press. doi:10.1109/CCGRID.2017.62

Richter, D., Neumann, T., & Polze, A. (2018). Security considerations for microservice architectures. *CLOSER 2018 - Proceedings of the 8th International Conference on Cloud Computing and Services Science* (pp. 608-615). Funchal, Madeira: SCITEPRESS.

Sun, Y., Nanda, S., & Jaeger, T. (2015). Security-as-a-Service for Microservices-Based Cloud Applications. *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)* (pp. 50-57). Vancouver, BC, Canada: IEEE.

Suneja, S., Kanso, A., & Isci, C. (2019). Can Container Fusion Be Securely Achieved? *Proceedings of the 5th International Workshop on Container Technologies and Container Clouds - WOC '19* (pp. 31-36). Davis CA USA: Association for Computing Machinery. doi:10.1145/3366615.3368356

Tian, D. (., Choi, J. I., Hernandez, G., Traynor, P., & Butler, K. R. (2019). A Practical Intel SGX Setting for Linux Containers in the Cloud. *Proceedings of the Ninth*

ACM Conference on Data and Application Security and Privacy (pp. 255–266). New York, NY, USA: Association for Computing Machinery.

Torkura, K. A., Sukmana, M. I., & Meinel, C. (2017). Integrating Continuous Security Assessments in Microservices and Cloud Native Applications. *UCC '17: Proceedings of the10th International Conference on Utility and Cloud Computing* (pp. 171-180). Austin Texas USA: Association for Computing Machinery.

Torkura, K. A., Sukmana, M. I., Kayem, A. V., Cheng, F., & Meinel, C. (2018). A Cyber Risk Based Moving Target Defense Mechanism for Microservice Architectures. *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)* (pp. 932-939). Melbourne, VIC, Australia: IEEE. doi:10.1109/BDCloud.2018.00137

Walsh, K., & Manferdelli, J. (2017). Mechanisms for Mutual Attested Microservice Communication. *UCC '17 Companion: Companion Proceedings of the10th International Conference on Utility and Cloud Computing* (pp. 59-64). Austin Texas USA: Association for Computing Machinery.

Wikipedia. (2022, 04 12). Retrieved from https://en.wikipedia.org/wiki/Microservices

Yarygina, T., & Bagge, A. H. (2018). Overcoming Security Challenges in Microservice Architectures. *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)* (pp. 11-20). Bamberg, Germany: IEEE. doi:10.1109/SOSE.2018.00011

Yu, D., Jin, Y., Zhang, Y., & Zheng, X. (2018). A survey on security issues in services communication of Microservices-enabled fog applications. *Concurrency Computat Pract Exper. 2018*.