HO
GENT

Departement IT en Digitale Innovatie

Container orkestratie security: indentificeren van problemen en onderzoek naar de impact van security tools

Nick Heymans

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Wim De Bruyn
Co-promotor:
Steven Trescinski

Instelling: —

Academiejaar: 2020-2021

Tweede examenperiode

Departement IT en Digitale Innovatie

Container orkestratie security: indentificeren van problemen en onderzoek naar de impact van security tools

Nick Heymans

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Wim De Bruyn
Co-promotor:
Steven Trescinski

Instelling: —

Academiejaar: 2020-2021

Tweede examenperiode

Woord vooraf

Samenvatting

In deze bachelorproef worden de moderne veiligheidsrisico's geanalyseerd die gepaard gaan met container virtualisatie en container orkestratie. Hierbij zal specifiek gekeken worden naar de grootste veiligheidsrisico's, welke effecten deze kunnen hebben op een productieomgeving en hoe deze vermeden kunnen worden. Tevens zal onderzocht worden welke 'security tools' (zoals 'Project Calico' en 'Kube-hunter') er bestaan en hoe deze ingezet kunnen worden bij het beveiligen van een container cluster. De laatste jaren is het gebruik van containers in de IT infrastructuur fors gestegen en dit zal zo blijven evolueren in de komende jaren. Aan alle technologieen zijn nu eenmaal veiligheidsrisico's verbonden, container orkestratie vormt hier geen uitzondering op de regel. In voorgaand onderzoek werd er reeds gefocust op de grootste veiligheidsrisico's desondanks is er zeer weinig ingegaan op de effecten bij het toepassen van 'best practices'. Met deze paper tracht ik het gebruik en de daaraan verbonden risico's van container virtualisatie en container orkestratie te onderzoeken. Daarnaast zal er ook gekeken worden naar hoe de verschillende 'security tools' kunnen helpen bij het beveiligen van containers. Ten slotte wordt er onderzocht hoe deze risico's vermeden of opgelost kunnen worden en welk effecten ze hebben op de relevante criteria. Dit laatste zal via een 'proof-of-concept' opstelling gebeuren.

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

1. Inleiding

Tijdens de ontwikkeling van traditionele applicaties wordt de applicatie ontwikkeld in een specifiek testomgeving. Vervolgens wordt de applicatie overgezet naar de productieomgeving wat vaak voor problemen zorgt (bijvoorbeeld van een Linux testomgeving naar een Windows productieomgeving). Deze problemen kunnen vermeden worden door gebruik te maken van containers. Deze vergemakkelijken het uitrollen en schalen van applicaties.

Een container is een pakket waar één enkele applicatie in zit, samen met alle nodige afhankelijkheden(Education, 2019). Dit zorgt ervoor dat deze gemakkelijk en snel van de ene omgeving naar de andere kan overgezet worden.

Naarmate het gebruik van containers steeg, steeg ook de nood naar op manier om deze vanuit één centrale locatie te beheren. Om aan deze vraag te voldoen werden container orkestratie tools, zoals Kubernetes [1], ontwikkeld. Deze tools helpen bij het opzetten, uitbreiden en verbinden van een grote hoeveelheid containers.

In deze bachelorproef worden de moderne veiligheidsrisico's geanalyseerd die gepaard gaan met container virtualisatie en container orkestratie. Hierbij zal specifiek gekeken worden naar de grootste veiligheidsrisico's, welke effecten deze kunnen hebben op een productieomgeving en hoe deze vermeden kunnen worden.

Aan alle technologieën zijn nu eenmaal veiligheidsrisico's verbonden, container virtualisatie en container orkestratie vormen hier geen uitzondering op de regel. Een groot onderdeel van container security zijn de zogenaamde 'security tools' (zoals 'Project Calico' en 'Kube-hunter').

[1] https://kubernetes.io/

In voorgaand onderzoek werd er reeds gefocust op de grootste veiligheidsrisico's desondanks is er zeer weinig ingegaan op de effecten bij het toepassen van 'best practices' en het gebruik van 'security tools. Met deze paper tracht ik het gebruik, en de daaraan verbonden risico's, van container virtualisatie en container orkestratie te onderzoeken. Daarnaast zal er ook gekeken worden naar hoe de verschillende 'security tools' kunnen helpen bij het beveiligen van containers.

Ten slotte wordt er onderzocht hoe deze risico's vermeden of opgelost kunnen worden en welk effecten ze hebben op de relevante criteria. Dit laatste zal via een 'proof-of-concept' opstelling gebeuren.

1.1 Probleemstelling

Het concrete probleem bestaat er dus uit dat veel bedrijven gebruik beginnen maken van containers en container orkestratie tools zonder al te veel aandacht te besteden aan de beveiliging hiervan. Er dient gekeken te worden naar wat voor effecten de beveiliging van een container omgeving met zich mee brengt en hoe men best te werk gaat.

1.2 Onderzoeksvraag

Wat zijn de belangrijkste security risicos? Welke security tools zijn er en hoe werken ze? Welke impact hebben best practices en security tools op criteria?

1.3 Onderzoeksdoelstelling

Het doel van deze bachelorproef is hoofdzakelijk om verslag met aanbevelingen op stellen voor het beveiligen van een container cluster. Deze aanbevelingen zullen gestaafd worden door enkele scenarios en hun effect op enkele criteria.

1.4 Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:

In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.

In Hoofdstuk 3 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen.

In Hoofdstuk 4, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op

de onderzoeksvragen. Daarbij wordt ook een aanzet gegeven voor toekomstig onderzoek binnen dit domein.

2. Stand van zaken

De stand van zaken of *State of the art* geeft een beeld van de technologieën die worden overwogen voor dit onderzoek en op welke manieren ze kunnen toegepast worden om een antwoord te vinden op de onderzoeksvragen.

2.1 Containers

In dit hoofdstuk zal uitgelegd worden wat containers zijn, hoe ze werken en waarvoor ze worden gebruikt.

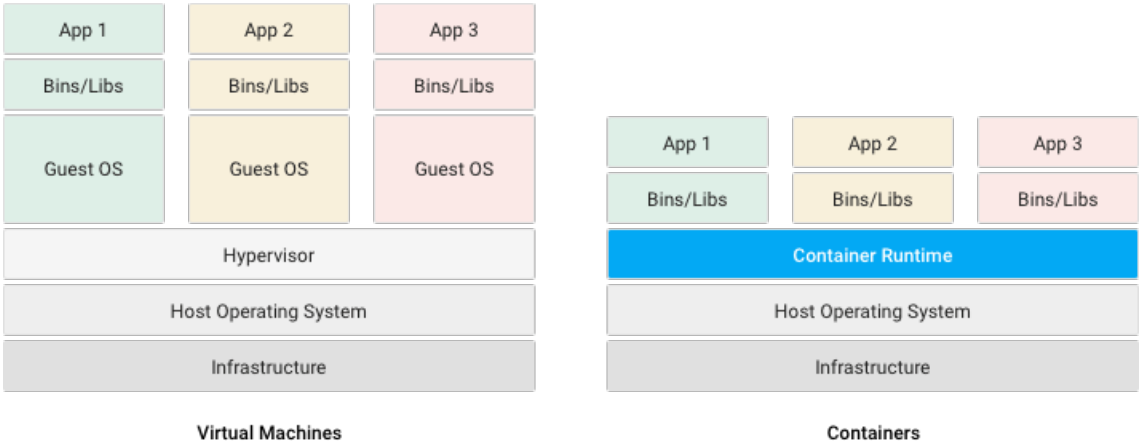
Containers bieden veel voordelen vergeleken met normale virtuele machines (vm's). Ze kunnen snel opgezet worden en zijn gemakkelijk om te configureren terwijl virtuele machines vaak groot en traag zijn. Containers zijn pakketten waarin een applicatie verpakt zit, samen met al zijn benodigde bibliotheken en afhankelijkheden. Hierdoor kunnen ze vlot van de ene omgeving naar de andere worden overgezet zonder dat er extra configuratie nodig is (Education, 2019). Containers maken gebruik van een vorm van besturingssysteem virtualisatie om processen te isoleren van het host besturingssysteem en zo ook het CPU gebruik en hoeveelheid RAM geheugen van die processen te controleren.(Docker, 2018)

Containers hebben geen eigen besturingssysteem nodig, alle containers delen 1 gezamenlijke 'runtime engine'. Een runtime engine is de laag die verantwoordelijk is voor de communicatie tussen het besturingssysteem van de host machine en de containers zelf. De meeste gebruikte runtime engine is de 'Docker Engine'[1].

[1] https://docs.docker.com/engine/

2.1.1 Container vs. virtuele machine

Bij traditionele virtuele machines virtualiseert een 'hypervisor' de fysieke hardware. De hypervisor regelt het resource gebruik tussen de verschillende vm's en zorgt ervoor dat de hardware van de 'host' (de fysieke hardware waar de hypervisor op geinstaleed is) eerlijk verdeelt wordt. Elke vm heeft zijn eigen volwaardig besturingssysteem, wat ervoor zorgt dat ze zeer veel resources gebruiken en vaak traag zijn. In plaats van de onderliggende hardware te visualiseren containers het besturingssysteem (meestal is dit Linux), zodat elke individuele container enkel de applicatie en de bijhorende bibliotheken en afhankelijkheden bevat (Education, 2020).



Figuur 2.1: Container vs. virtuale machine (Google, 2016)

2.1.2 Waarvoor worden containers gebruikt

Containers zijn zeer veelzijdig en kunnen in veel verschillende omstandigheden gebruikt worden. Docker[2] geeft enkele use cases waarvoor containers zeer geschikt zijn (**Docker2021**) :

- *Microservices*: Containers zijn klein en licht, waardoor ze goed passen bij microservice-architecturen waarin applicaties zijn opgebouwd uit vele, losjes gekoppelde en onafhankelijk services.
- *Modernisering en migratie van applicaties*: Een van de meest voorkomende benaderingen voor het moderniseren van applicaties begint met het containeriseren ervan, zodat ze naar de cloud kunnen worden gemigreerd.
- *Nieuwe ontwikkelaars snel inwerken*: Door gebruik te maken van containers verloopt het opzetten van een nieuwe lokale ontwikkelingsomgeving snel en vlot, waardoor de ontwikkelaars direct aan de slag kunnen

[2]https://www.docker.com/

2.2 Docker

Hier zal uitgelegd worden wat docker is en hoe het ontstaan en gegroeid is.

2.2.1 Hoe werkt docker

Hier zal de werking van Docker uitgelegd worden, ook de verschillende technische termen die eigen zijn aan Docker zullen uitgelegd worden.

2.3 Container orkestratie

Hier zal uitgelegd worden wat container orkestratie is en waarvoor het gebruikt kan worden.

2.3.1 Hoe werkt container orkestratie

Hier zal de innerlijke werking van container orkestratie uitgelegd worden.

2.4 Kubernetes

Hier zal uitgelegd worden wat k8s is en hoe het werkt, ook de verschillende technische termen die eigen zijn aan k8s zullen uitgelegd worden.

2.5 Security

Hier word het security aspect van containers en container orkestratie besproken. (redelijk high level want kan ongeloofelijk complex worden)

2.5.1 Meest voorkomende security problemen

Hier worden de meest voorkomende security problemen opgelijst en worden ze kort besproken (Techinsche uitleg over hoe deze problemen onstaan)

2.5.2 Hoe een container cluster beveiligen

Uitleg over hoe een container cluster beveiligt kan worden (Best practices en security tools) + uitleg over ingebouwde security features.

2.6 Security tools

Hier worden enkele security tools opgeslijst en hun werking kort besproken.

3. Methodologie

4. Conclusie

A. Onderzoeksvoorstel

Het onderwerp van deze bachelorproef is gebaseerd op een onderzoeksvoorstel dat vooraf werd beoordeeld door de promotor. Dat voorstel is opgenomen in deze bijlage.

A.1 Inleiding en State-of-the-art

A.1.1 Wat zijn containers?

Het uitrollen en schalen van applicaties wordt steeds vaker gedaan met behulp van containers. Tijdens de ontwikkeling van traditionele applicaties wordt de applicatie ontwikkeld in een specifiek testomgeving. Vervolgens wordt de applicatie overgezet naar de productieomgeving wat vaak voor problemen zorgt (bijvoorbeeld van een linux testomgeving naar een Windows productieomgeving). Een container is een pakket waar één enkele applicatie in zit, samen met alle nodige afhankelijkheden(Education, 2019). Dit zorgt ervoor dat deze gemakkelijk en snel van de ene omgeving naar de andere kan overgezet worden. De containers maken gebruik van een 'runtime engine', dit is een laag die verantwoordelijk is voor de communicatie tussen het operating system van de host machine en de containers zelf. De meeste gebruikte 'runtime engine' is de 'Docker Engine'[1]. Deze is al sinds 2013 de industriestandaard als het gaat over container software(McCarty, 2018). Naarmate het gebruik van containers steeg, steeg ook de nood naar op manier om deze vanuit één centrale locatie te beheren. Om aan deze vraag te voldoen werden container orkestratie tools, zoals Kubernetes [2], ontwikkeld. Deze tools helpen bij het opzetten, uitbreiden en verbinden van een grote hoeveelheid containers.

[1] https://docs.docker.com/engine/
[2] https://kubernetes.io/

A.1.2 Waarom container applicaties?

Container applicaties hebben enkele voordelen tegenover normale applicaties, ze draaien namelijk geïsoleerd van de rest van het systeem. Ze kunnen dus perfect werken zonder afhankelijk te zijn van andere containers. Dit garandeerd dat als er één container aangetast is, de rest zonder interruptie kan verderwerken. De containers delen wel verschillende resources van het host systeem, wat de deur opent voor veiligheidsinbreuken tussen containers.

A.1.3 Context voor dit onderzoek

Gartner (Pettey, 2019) voorspeld dat tegen 2022 maar liefst 75% van alle internationale organisaties gecontaineriseerde applicaties zullen gebruiken in hun productieomgeving. Dit zowel in lokale datacenters alsook in online cloud omgevingen. Uit een raport van Tripwire (2019) blijkt dat 94% van bevraagden bezorgd zijn over de veiligheid van hun containers. Uit hetzelfde raport blijkt ook dat 47% weet dat ze kwetsbare containers gebruiken in hun productieomgeving. Spijtig genoeg werd bij voorgaande onderzoeken, zoals StackRox (2020), het effect van 'security best practices en tools' op opzetsnelheid, benodigde resources en stabiliteit steeds onderbelicht.

A.1.4 Verloop van het onderzoek

In deze paper zal ik onderzoeken wat de belangrijkste bronnen van veiligheidsinbreuken zijn en hoe deze vermeden kunnen worden. Tegelijkertijd krijg ik via dit onderzoek de opportuniteit om er achter te komen of er vooral technische problemen of menselijke fouten aan de basis liggen van de veiligheidsrisico's. In de volgende paragraaf staat er beschreven hoe ik te werk zal gaan.

A.2 Methodologie

Voor dit onderzoek zullen er drie scenario's opgezet worden. Elk scenario zal verschillende keren worden uitgevoerd en voor elk criteria zal het genomen worden (eventueel reking houdende met uitschieters). Bij elke scenario zullen er verschillende 'security best practices en tools' gebruikt worden. Deze zullen getest worden op basis van de volgende criteria:

- Deployment snelheid
- Benodigde resources
- Stabiliteit

Voorbeelden van scenario's:

- S(0): Er wordt een container applicatie opgezet in een Kubernetes cluster zonder extra security configuratie.
- S(1): Er wordt een container applicatie opgezet in een Kubernetes cluster en enkele 'best practices' worden toegepast.
- S(2): Er wordt een container applicatie opgezet in een Kubernetes cluster waar er gebruik word gemaakt van enkele 'security tools' zoals 'Project Calico' en 'Kube-hunter'.

Door gebruik te maken van deze scenario's en criteria hopen we vast te stellen dat het toepassen van 'security best practices en tools' een positieve invloed heeft op het gebruik van containers en orkestratie tools.

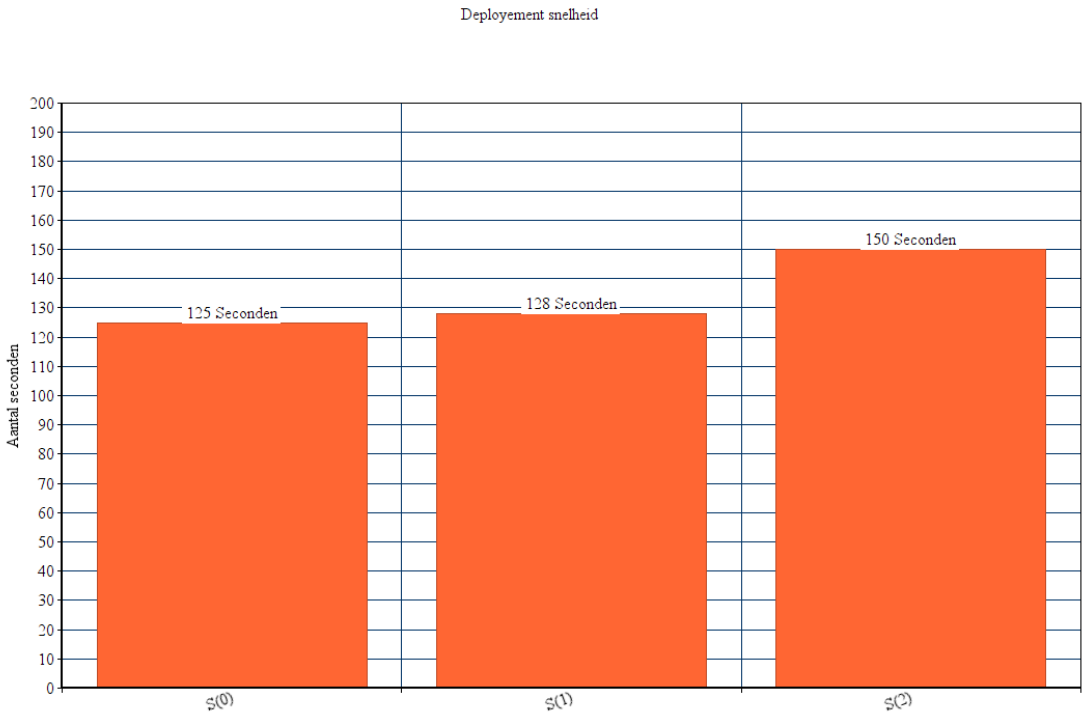
A.3 Verwachte resultaten

Op basis van de criteria wordt er verwacht dat Scenario 0 en 1 even snel op opgezet kunnen worden en evenveel resources gebruiken. Ze zullen beide kwetsbaarder zijn aangezien de 'best practices' vooral bestaan uit het correct gebruik van wachtwoorden en gebruiker privileges. Scenario 2 daarintegen zal iets meer tijd nodig hebben om opgezet te worden(zie Figuur1) en zal daarbij meer resources gebruiken(zie Figuur2). Dit zou te wijten zijn aan de gebruikte 'security tools' die extra tijd en resource nodig hebben.

A.4 Verwachte conclusies

Uit dit onderzoek willen we concluderen dat het toepassen van 'best practices' en het correct gebruik van security tools een positief effect teweeg brengt bij het gebruik van container orkestratie tools. We trachten daarnaast ook aan te duiden dat het omzeilen van security risico's een belangrijk aspect is bij het ontwikkelen van container applicaties. Tot slot kan er geconcludeerd worden dat het beveiligen van container clusters steeds belangrijker wordt. Daarnaast is het tevens van belang dat de persoon die een cluster opzet daarbij de onderliggende werkwijze goed kent en zich bewust is van de mogelijke valkuilen.

A.5 Bijlagen



Figuur A.1: Verwachte opstart tijd

Benodigde resources	CPU %	Memory %
S(0)	45%	30%
S(1)	46%	32%
S(2)	55%	45%

Figuur A.2: Verwacht resource gebruik

Bibliografie

Docker. (2018). *What is a Container?* https://www.docker.com/resources/what-container

Education, I. C. (2019, mei 25). *Containerization*. IBM. https://www.ibm.com/cloud/learn/containerization#toc-what-is-co-r25Smlqq

Education, I. C. (2020, september 2). *Containers vs. VMs: What's the Difference?* IBM Cloud Education. Verkregen 8 maart 2021, van https://www.ibm.com/cloud/blog/containers-vs-vms

Google. (2016). *Containers at Google: A better way to develop and deploy applications*. Google. https://cloud.google.com/containers

McCarty, S. (2018, februari 22). *A Practical Introduction to Container Terminology*. https://developers.redhat.com/blog/2018/02/22/container-terminology-practical-introduction/#h.6yt1ex5wfo3l

Pettey, C. (2019, april 23). *6 Best Practices for Creating a Container Platform Strategy*. https://www.gartner.com/smarterwithgartner/6-best-practices-for-creating-a-container-platform-strategy/

StackRox. (2020, februari 19). *State of Container and Kubernetes Security* (onderzoeksrap.). StackRox. https://www.stackrox.com/kubernetes-adoption-security-and-market-share-for-containers/

Tripwire. (2019, januari 1). *Tripwire State of Container Security Report* (onderzoeksrap.). Tripwire. Verkregen 30 november 2020, van https://3b6xlt3iddqmuq5vy2w0s5d3-wpengine.netdna-ssl.com/state-of-security/wp-content/uploads/sites/3/Tripwire-Dimensional-Research-State-of-Container-Security-Report.pdf