



**HoGent**

Faculteit Bedrijf en Organisatie

Unikernels

Michiel De Wilde

Scriptie voorgedragen tot het bekomen van de graad van  
Bachelor in de toegepaste informatica

Promotor:  
Bert Van Vreckem  
Co-promotor:  
Jan Janssen

Instelling: Hogeschool Gent

Academiejaar: 2015-2016

2e examenperiode



Faculteit Bedrijf en Organisatie

Unikernels

Michiel De Wilde

Scriptie voorgedragen tot het bekomen van de graad van  
Bachelor in de toegepaste informatica

Promotor:  
Bert Van Vreckem  
Co-promotor:  
Jan Janssen

Instelling: Hogeschool Gent

Academiejaar: 2015-2016

2e examenperiode

## Samenvatting

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

# Voorwoord

Toen ik begon met programmeren had ik geen idee wat er gebeurde in de achtergrond van de computer. Ik starte met de simpele to-do-list applicaties om alles te leren over programmeren. Na een tijd kwam ik een black box tegen: het besturingssysteem. Vooral om servers sneller te laten werken en de techniek erachter te leren kennen begon ik aan een zoektocht. Linux was de startplek bij uitstek. Package managers en file systems waren de eerste concepten die mij met verstomming lieten staan. Toen ik meer en meer naar infrastructuur keek begon ik termen te leren en alle handige tips om ervoor te zorgen dat je server altijd beschikbaar is. Toen een paar jaar geleden Docker voor het eerst echt vaart maakte met containers was ik verbaasd. Ik dacht eerst dat dit nooit zou werken. Na een tijd heb ik wel het licht gezien en gebruikte ik containers meer en meer. Toen er gevraagd werd om een onderwerp voor mijn thesis dacht ik meteen en wat volgens mij de volgende stap is: unikernels.

# Inhoudsopgave

<b>1</b>	<b>Inleiding</b>	<b>4</b>
1.1	Probleemstelling en Onderzoeksvragen . . . . .	6
<b>2</b>	<b>Methodologie</b>	<b>7</b>
<b>3</b>	<b>Corpus</b>	<b>8</b>
<b>4</b>	<b>Conclusie</b>	<b>9</b>

# Hoofdstuk 1

## Inleiding

Vroeger was de tijd dat je een computer kon gebruiken beperkt. Vooral bij de eerste computers had men problemen om programma's en concepten uit te werken omdat de computer door meerdere mensen gebruikt werd. Een voorbeeld vanuit een situatie uit die tijd is het ontwikkelen van een programma. De source code van een programma werd manueel ingegeven en als een job in een queue gezet. Pas een paar dagen later kon men de resultaten bekijken. Als je een kleine schrijf- en/of logische denkfout maakte dan kon je opnieuw beginnen. De grootste bijdrage tot de ontwikkelsnelheid van programma's is de lengte van de feedbackcyclus: hoe snel kan je je programma testen laten werken. Dit leidt tot een verlies van tijd en dus geld.

Timesharing werd uitgevonden om het verlies van tijd te beperken. Bij timesharing konden de gebruikers inloggen op een console en zo computer gebruiken. Dit was een technische uitdaging. De computer zou van de ene context naar de andere moeten kunnen veranderen. Timesharing en verschillende gebruikers op 1 computer zou de basis vormen voor de moderne besturingsystemen. 1 Van de nadelen van timesharing is de isolatie van twee verschillende processen. Als er een kans bestaat dat ze elkaar kunnen beïnvloeden is dit een heel groot probleem.

Een besturingsysteem moet kunnen werken op verschillende soorten hardware. Om al deze hardware te ondersteunen moet er een standaard zijn waarop een besturings-systeem kan gebouwd worden. Dit zou betekenen dat veel van de complexiteit van het besturingssysteem naar de hardware zou verhuizen. Spijtig genoeg gebeurt dit nog altijd niet. Dit is 1 van de voornaamste redenen waarom de grootte van een OS tussen de 200MB en 1GB kan liggen. Een aantal besturingssystemen doen niet de moeite om de vele soorten hardware apparatuur te ondersteunen. Zij tonen de specificaties die een hardware apparaat moet hebben wanneer je het besturingssysteem wil gebruiken.

Doorheen de tijd werden computers meer krachtig en applicaties konden niet alle middelen van de computer ten volle benutten. Dit zorgde voor de creatie van virtual resources. Deze gingen de fysieke hardware simuleren om zo verschillende applicaties tegelijkertijd te laten werken op dezelfde virtuele machine. Zo worden de middelen van een fysieke machine optimaal gebruikt. De algemene term voor dit concept is virtualisatie. Bij het virtualiseren van een server gaat men een fysieke server opdelen in verschillende kleine en geïsoleerde delen. Deze delen kunnen dan gebruikt worden door verschillende gebruikers. De voordelen van het virtualiseren van een server zijn de volgende: financieel voordeel (men kan van 1 taak naar meerdere taken gaan op 1 server), het besparen van energie (minder servers gebruiken want ze worden beter benut), betere beschikbaarheid.

Er zijn verschillende vormen van virtualisatie. De meest gebruikte vormen bij servers is de virtualisatie van het besturingssysteem en virtualisatie van de hardware.

De virtuele machine is een toepassing van de virtualisatie van de hardware. Een virtuele machine bevindt zich op een fysieke machine door gebruik te maken van een hypervisor. De nadelen van een virtuele machine als de kleinste eenheid is de schaalbaarheid en de veiligheid. Wanneer je je eigen server opstelde dan moet je een besturingssysteem kiezen. Meestal gaan we kiezen voor een Linux distributie. Daarna moet je de software installeren die ervoor zorgt dat de applicatie kan werken. Daaropvolgend moet je je server beveiligen. Het veranderen van de SSH-poort, zorgen dat de root-user niet te bereiken is. IPtables en firewalls opstellen. Voor dit te vergemakkelijken kunnen we gebruikmaken van een configuratie management tool (Chef, Puppet, Ansible...).

Een hypervisor is een stuk software, firmware of hardware waarop een VM kan draaien. Een hypervisor zal zich bevinden op een host machine of op "baremetal". De host machine zorgt voor de middelen zoals CPU, RAM, ... Elke virtuele machine die zich bevindt op de host machine zal dan gebruik maken van deze middelen. We spreken van een hosted virtualization hypervisor of een bare-metal hypervisor. De eerste zal zich bevinden op een het besturingssysteem van de host machine en heeft geen directe toegang tot de hardware. Dit heeft als voordeel dat de hardware niet zo belangrijk is maar zorgt voor een extra laag tussen de hardware en de hypervisor. Een goede regel is: "hoe minder lagen we hebben, hoe beter de performantie." Een alternatief is een bare-metal hypervisor. Hierbij is er geen extra laag tussen de hardware en hypervisor. We hebben geen host operating system nodig omdat de hypervisor zich rechtstreeks bevindt op de interfaces van hardware. Dit zorgt voor betere performantie, schaalbaarheid en stabiliteit. Een hypervisor zorgt ervoor dat de virtuele machines het guest OS kan beheren en uitvoeren.

Meest bekende voorbeelden van bare-metal hypervisors zijn VMware ESXi, Micro-



soft Hyper-V en Xen.

Naast hardware virtualisatie kunnen we ook gebruikmaken van de virtualisatie van het besturingssysteem. Bij deze toepassing van virtualisatie gaan we de mogelijkheden van de kernel van het besturingssysteem gebruiken. De kernel van bepaalde besturingssystemen laat ons toe om meerdere geïsoleerde user spaces tegelijkertijd te laten werken. Dit zorgt ervoor dat er maar 1 besturingssysteem en een kernel zijn. De verschillende user spaces maken gebruik van CPU, geheugen en netwerk van de host server. Elke user space heeft zijn eigen configuratie omdat de user spaces op zichzelf staan en geïsoleerd zijn de andere user spaces. Tegenover hardware virtualisatie zal de besturingssysteem virtualisatie minder gebruik maken van het geheugen en de CPU omdat er maar 1 kernel is en niet meerdere besturingssystemen. Het opstarten neemt maar een fractie van de tijd in beslag tegenover virtuele machines. Deze user spaces worden ook wel containers genoemd.

Meest bekende voorbeelden zijn Docker en Linux Containers.

Er werd al aangehaald bij de virtuele machines dat veiligheid een probleem kan zijn. Dit probleem verdwijnt niet bij containers omdat ze 1 kernel delen. Wanneer de beveiliging van 1 container kan geschonden worden dan wordt het direct gemakkelijker om toegang te verschaffen tot de andere containers. Het gebruiken van 1 OS als de basis voor een aantal containers kan ervoor zorgen dat wanneer dit besturingssysteem niet meer opstaat al deze applicaties niet meer werken.

## **1.1 Probleemstelling en Onderzoeksvragen**

Unikernels zijn een nieuwe stroom binnen het landschap van besturingssystemen. We hebben al aangehaald dat containers de nieuwe werkwijze is wanneer men applicaties wilt ontwikkelen en schaalbaar wilt maken. Unikernels gaat nog een stap verder. De systeembeheerders moeten dus mee met containers en de veranderingen ondergaan. De vraag is welke veranderingen er zich zullen voordoen wanneer unikernels op de markt komen. Zullen de competenties van de systeembeheerder veranderen? Wordt het opzetten van applicaties meer en meer eenvoudiger of juist niet? We kunnen wel spreken over de opvolger van containers maar is deze al werkbaar in de toekomst? Wat is de impact op beveiliging, meer bepaald aspecten als beschikbaarheid, autorisatie, integriteit en vertrouwelijkheid van gegevens?

# **Hoofdstuk 2**

## **Methodologie**

## **Hoofdstuk 3**

### **Corpus**

## **Hoofdstuk 4**

### **Conclusie**

# **Bibliografie**

## **Lijst van figuren**

## **Lijst van tabellen**