

LAB: Logging in Kubernetes met EFK (Elasticsearch, Fluentd en Kibana)

In een vorige SIG hebben we gekeken naar het Kubernetes container platform.

Kubernetes managed nodes, waarop zogeheten PODs worden uitgevoerd. Een POD bevat één of meerdere containers, meestal gebaseerd op Docker.

Vanwege schaalbaarheid (en fail-over) kunnen in een systeem meerdere PODs met identieke containers draaien. Een collectie van identieke PODs wordt ontsloten door een zogenaamde service, lees: load balancer. Voor verschillende applicaties worden aparte services gedefinieerd.

De load balancer stuurt een HTTP request door naar één van de PODS waarin de betreffende applicatie wordt uitgevoerd.

Logging van een applicatie wordt geschreven naar het OS van de container, die dus in een willekeurige POD draait. Zoals gezegd worden PODs gemanaged door Kubernetes, en dus automatisch op- en afgeschaald. Hiermee verdwijnt tevens de logging!

Vanwege de grote aantallen PODs die in een systeem kunnen draaien, is het tevens ondoenlijk om handmatig alle logging van de PODs te openen en analyseren. Om dit probleem op te lossen, kan de logging van de verschillende containers automatisch worden afgevangen en doorgestuurd naar een decentrale bestemming.

De 2 meest gangbare oplossingen op dit gebied zijn momenteel de ELK en EFK stacks.

ELK is een afkorting voor Elasticsearch, Logstash en Kibana. De F in EFK staat voor Fluentd en vervangt Logstash. Fluentd is een zogenaamde Log aggregator, geschreven in Ruby.

In de setup van Fluentd kunnen meerdere input en output systemen, waaronder Kubernetes, Apache server en bestanden worden geconfigureerd. Tevens is het mogelijk om op inhoud van de data te filteren.

Voor Kubernetes wordt Fluentd uitgevoerd als een Daemon set, en worden de log regels verzameld die in de verschillende containers worden vastgelegd.

Een Daemon set zorgt er voor dat voor iedere (of sommige) nodes een specifieke Pod, in dit geval Fluentd) wordt uitgevoerd.

Naast Fluentd bestaat tevens Fluent-bit, een meer licht gewicht variant van Fluentd met weinig tot geen aggregatie mogelijkheden.

Zo, genoeg theorie. Laten we eens naar EFK gaan kijken.

Voorwaarden

Om dit lab uit te kunnen voeren heb je in ieder geval de volgende software geïnstalleerd:

- Docker
- Powershell
- Boxstarter
- Chocolatey
- Maak een folder met de naam **installefk** aan, en plaats daar de yaml files in die in Appendix B zijn opgenomen

Setup van het Kubernetes cluster

In de vorige SIG heb je, als het goed is, Minikube geïnstalleerd. Minikube draait een Kubernetes cluster op één node in een VM op jouw lokale computer, en is bedoeld als speel omgeving.

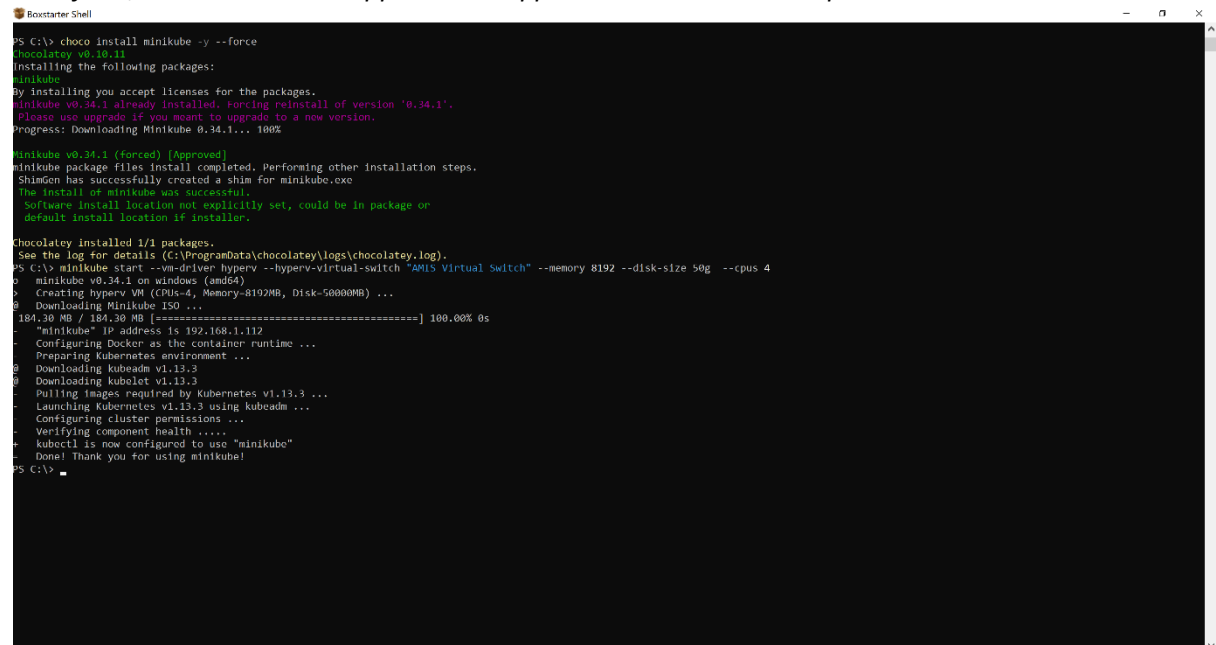
Minikube heeft als vervelende eigenschap dat het een dynamisch ip adres heeft, waardoor na een machine herstart het een ander ip adres kan gebruiken. Hierdoor zijn de certificaten die door Kubernetes worden gebruikt niet meer geldig en kan de tooling niet meer connecten naar het cluster.

Om deze reden ga je de minikube omgeving die eerder is geïnstalleerd eerst verwijderen. Een beschrijving van de stappen om dit te doen vind je in appendix A.

Nadat de oude minikube omgeving is opgeruimd, ga je een nieuwe aanmaken via de volgende stappen:

1. Open boxstarter shell
2. Type: **mkdir ~/.kube** (dit maakt een lege folder .kube aan in jouw user directory C:\users\[username])
3. **choco install minikube -y --force**
4. **minikube start --vm-driver hyperv --hyperv-virtual-switch "AMIS Virtual Switch" --memory 8192 --disk-size 50g --cpus 4** (--memory 4096 en --cpus 2 zou voldoende moeten zijn)

Zoals gezegd in de appendix is het minikube installatie proces niet bepaald stabiel. In het geval deze faalt, herhaal dan de stappen uit de appendix A. om minikube op te ruimen.



```
PS C:\> choco install minikube -y --force
Chocolatey v0.10.11
Installing the following packages:
minikube
By installing you accept licenses for the packages.
minikube v0.34.1 already installed. Forcing reinstall of version '0.34.1'.
Please use upgrade if you meant to upgrade to a new version.
Progress: Downloading Minikube 0.34.1... 100%

minikube v0.34.1 (forced) [Approved]
minikube package files install completed. Performing other installation steps.
ShimGen has successfully created a shim for minikube.exe
The install of minikube was successful.
Software install location not explicitly set, could be in package or
Default install location if installer.

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\> minikube start --vm-driver hyperv --hyperv-virtual-switch "Aq15 Virtual Switch" --memory 8192 --disk-size 50g --cpus 4
minikube v0.34.1 on windows (amd64)
- Creating hyperv VM (CPUs=4, Memory=8192MB, Disk=50000MB) ...
- Downloading Minikube ISO ...
184.30 MB / 184.30 MB [=====] 100.00% 0s
- "minikube" IP address is 192.168.1.112
- Configuring Docker as the container runtime ...
- Preparing Kubernetes environment ...
- Downloading kubeadm v1.13.3
- Downloading kubelet v1.13.3
- Pulling images required by Kubernetes v1.13.3 ...
- Launching Kubernetes v1.13.3 using kubeadm ...
- Configuring cluster permissions ...
- Verifying component health .....
- kubectll is now configured to use "minikube"
- Done! Thank you for using minikube!
PS C:\>
```

5. Vraag het minikube ip en de cluster info op nadat minikube succesvol is geïnstalleerd:

- type: **minikube ip**
- type: **kubectl config use-context minikube**
- type: **kubectl cluster-info**

6. Nu ga je kijken welke add ons er in minikube beschikbaar zijn:

- type: **minikube addons list**

```

PS C:\> minikube ip
192.168.1.112
PS C:\> kubectl config use-context minikube
Switched to context "minikube".
PS C:\> kubectl cluster-info
Kubernetes master is running at https://192.168.1.112:8443
KubeDNS is running at https://192.168.1.112:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\> minikube addons list
- addon-manager: enabled
- dashboard: disabled
- default-storageclass: enabled
- efk: disabled
- freshpod: disabled
- gvisor: disabled
- heapster: disabled
- ingress: disabled
- logviewer: disabled
- metrics-server: disabled
- nvidia-driver-installer: disabled
- nvidia-gpu-device-plugin: disabled
- registry: disabled
- registry-creds: disabled
- storage-provisioner: enabled
- storage-provisioner-gluster: disabled
PS C:\> minikube addons enable dashboard
- dashboard was successfully enabled
PS C:\>
PS C:\> minikube dashboard

```

Zoals je kunt zien is er een add-on EFK beschikbaar. In de volgende stappen ga je de EFK stack echter zelf installeren en maak je dus geen gebruik van de add-on.

!!Mocht je toch de add-on willen gebruiken, dan kun je deze inschakelen via het commando: minikube addons enable efk

type: minikube addons enable dashboard

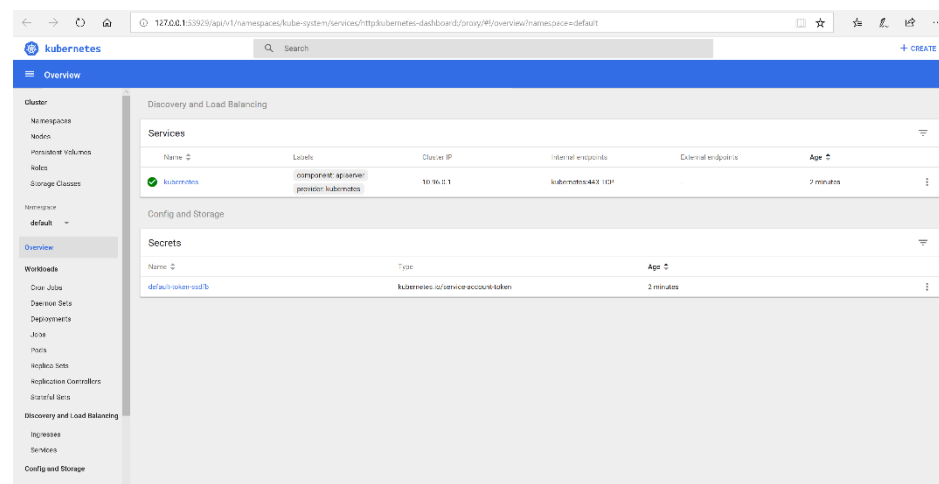
type: minikube dashboard

```

PS C:\> minikube dashboard
- Enabling dashboard ...
- Verifying dashboard health ...
- Launching proxy ...
- Verifying proxy health ...
- Opening http://127.0.0.1:53929/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/ in your default browser...

```

Houd het command window open om het dashboard actief te houden.



- Open een nieuwe boxstarter shell en installeer elasticsearch door middel van de volgende opdrachten:

kubectl create namespace logging

kubectl create -f C:/installefk/elastic.yaml -n logging

kubectl get pods -n logging

Als het goed is, zie je nu een naam die lijkt op: elasticsearch-7b4cc779b8-8sds7

Kijk ook naar de status van de pod. Wacht tot deze volledig opgestart is.

NAME	READY	STATUS	RESTARTS	AGE
elasticsearch-7b4cc779b8-572f4	0/1	ContainerCreating	0	15s

kubectl get service -n logging

Als het goed is, zie je informatie die lijkt op:

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
elasticsearch	NodePort	10.109.156.183	<none>	9200:31771/TCP	27s

Merk het poortnummer op via welke elasticsearch is blootgesteld, in dit geval 31771.

N.B. Aan een service wordt door Kubernetes willekeurig een poortnummer toegekend, op het moment dat deze wordt gecreëerd.

Probeer de pod te bereiken via deze service (herinner je het eerder opgevraagde ip van minikube, in het voorbeeld was dat: 192.168.1.112). Het commando is:

curl MINIKUBE_IP:ELASTICSEARCH_EXPOSED_PORT, in het geval van het voorbeeld is dat:

```
Boxstarter Shell
PS C:\> kubectl create namespace logging
namespace "logging" created
PS C:\> kubectl create -f C:\Users\jacco_c\Downloads\efk-kubernetes-master\efk-kubernetes-master\kubernetes\elastic.yaml -n logging
deployment.extensions "elasticsearch" created
service "elasticsearch" created
PS C:\> kubectl get pods -n logging
NAME                                READY    STATUS      RESTARTS   AGE
elasticsearch-7b4cc779b8-572f4      0/1      ContainerCreating   0          15s
PS C:\> kubectl get service -n logging
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
elasticsearch   NodePort    10.109.156.183 <none>         9200:31771/TCP   27s
PS C:\> curl 192.168.1.112:31771
{
  "statusCode": 200,
  "statusDescription": "OK",
  "content": {
    "name": "uXizXt3",
    "cluster_name": "docker-cluster",
    "cluster_uuid": "xvut86lwqrc-7JH6SNJPqg",
    "version": {
      "number": "6.5.4",
      "build_flavor": "default",
      "build_type": "tar..."
    }
  },
  "rawContent": "HTTP/1.1 200 OK\nContent-Length: 494\nContent-Type: application/json; charset=UTF-8\n\n{\n  \"name\": \"uXizXt3\",\\n  \"cluster_name\": \"docker-cluster\",\\n  \"cluster_uuid\": \"xvut86lwqrc-7JH6SNJPqg\",\\n  \"version\": {\\n    \"number\": \"6.5.4\",\\n    \"build_flavor\": \"default\",\\n    \"build_type\": \"tar...\",\\n  }\\n}\\n",
  "headers": {
    "Content-Length": 494,
    "Content-Type": "application/json; charset=UTF-8"
  },
  "inputFields": {},
  "links": {},
  "parsedhtml": mshtml.HTMLDocumentClass,
  "rawContentLength": 494
}
```

curl 192.168.1.112:31771

8. Open een nieuwe boxstarter shell en installeer Kibana door middel van de volgende opdrachten:

kubect create -f C:/installefk/kibana.yaml -n logging

kubectl get pods -n logging

NAME	READY	STATUS	RESTARTS	AGE
elasticsearch-7b4cc779b8-572f4	1/1	Running	0	4m
kibana-678774f4f6-zjl65	0/1	ContainerCreating	0	14s

kubectl get service -n logging

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
elasticsearch	NodePort	10.109.156.183	<none>	9200:31771/TCP	4m
kibana	NodePort	10.106.40.204	<none>	5601:30462/TCP	30s

minikube service list -namespace logging

NAMESPACE	NAME	URL
logging	elasticsearch	http://192.168.1.112:31771
logging	kibana	http://192.168.1.112:30462

Test Kibana in jouw browser op URL: **http://MINIKUBE_IP:KIBANA_EXPOSED_PORT.**

<http://192.168.1.112:30462>, in het geval van het voorbeeld.

```

PS C:\> kubectl create -f C:\Users\jacco_c\Downloads\efk-kubernetes-master\efk-kubernetes-master\kubernetes\kibana.yaml -n logging
deployment.extensions "kibana" created
service "kibana" created
PS C:\>
PS C:\> kubectl get pods -n logging
NAME                                READY   STATUS             RESTARTS   AGE
elasticsearch-7b4cc779b8-572f4      1/1     Running            0           4m
kibana-678774f4f6-zjl65             0/1     ContainerCreating  0           14s
PS C:\> kubectl get service -n logging
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
elasticsearch  NodePort    10.109.156.183 <none>        9200:31771/TCP   4m
kibana       NodePort    10.106.40.204 <none>        5601:30462/TCP   30s
PS C:\> kubectl get services
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes  ClusterIP   10.96.0.1     <none>        443/TCP           13m
PS C:\> kubectl get services -n logging
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
elasticsearch  NodePort    10.109.156.183 <none>        9200:31771/TCP   7m
kibana       NodePort    10.106.40.204 <none>        5601:30462/TCP   3m
PS C:\> minikube service list --namespace logging
-----
| NAMESPACE | NAME      | URL |
|-----|-----|-----|
| logging | elasticsearch | http://192.168.1.112:31771 |
| logging | kibana      | http://192.168.1.112:30462 |
|-----|-----|-----|

```

9. Installeer fluentd door middel van de onderstaande opdrachten. De namespace setting in de gebruikte yaml file zorgt er in dit voorbeeld voor dat de kube-system namespace wordt gebruikt. Dit is echter niet verplicht.

`kubectl create -f C:/installefk/fluentd-rbac.yaml`

Als het goed is zie je de volgende meldingen:

`serviceaccount "fluentd" created`

`clusterrole.rbac.authorization.k8s.io "fluentd" created`

`clusterrolebinding.rbac.authorization.k8s.io "fluentd" created`

`kubectl create -f C:/installefk/fluentd-daemonset.yaml`

`kubectl get pods -n kube-system`

```

NAME                                READY   STATUS    RESTARTS   AGE
coredns-86c58d9df4-dmgtp           1/1     Running   0           28m
coredns-86c58d9df4-j7z9m           1/1     Running   0           28m
etcd-minikube                       1/1     Running   0           27m
fluentd-5xg89                     1/1     Running   0         18s
.....
storage-provisioner                 1/1     Running   0           28m

```

`kube-system` is the namespace for objects created by the Kubernetes system.

12

Typically, this would contain pods like `kube-dns`, `kube-proxy`, `kubernetes-dashboard` and stuff like `fluentd`, `heapster`, `ingresses` and so on.

share improve this answer

answered May 15 '17 at 19:25



bartimar

1,742 ● 2 ● 19 ● 40


```
PS C:\> kubectl config set-context minikube --namespace=logging
Context "minikube" modified.
PS C:\> kubectl create -f C:\Users\jacco_c\Downloads\efk-kubernetes-master\efk-kubernetes-master\kubernetes\fluentd-rbac.yaml
serviceaccount "fluentd" created
clusterrole.rbac.authorization.k8s.io "fluentd" created
clusterrolebinding.rbac.authorization.k8s.io "fluentd" created
PS C:\> kubectl create -f C:\Users\jacco_c\Downloads\efk-kubernetes-master\efk-kubernetes-master\kubernetes\fluentd-daemonset.yaml
daemonset.extensions "fluentd" created
PS C:\> kubectl get pods -n kube-system
```

NAME	READY	STATUS	RESTARTS	AGE
coredns-86c58d9df4-dmgtp	1/1	Running	0	28m
coredns-86c58d9df4-j7z9m	1/1	Running	0	28m
etcd-minikube	1/1	Running	0	27m
fluentd-5xg89	1/1	Running	0	18s
kube-addon-manager-minikube	1/1	Running	0	27m
kube-apiserver-minikube	1/1	Running	0	27m
kube-controller-manager-minikube	1/1	Running	0	27m
kube-proxy-wglsr	1/1	Running	0	28m
kube-scheduler-minikube	1/1	Running	0	27m
kubernetes-dashboard-ccc79bfc9-tc8hf	1/1	Running	0	26m
storage-provisioner	1/1	Running	0	28m

Controleer of fluentd verbonden is met elasticsearch door de log van de betreffende fluentd pod op te vragen:

kubectl logs fluentd-5xg89 -n kube-system

Als het goed is zie je een soortgelijke melding als de volgende:

```
2019-03-01 09:20:01 +0000 [info]: #0 [out_es] Connection opened to Elasticsearch cluster =>
{:host=>"elasticsearch.logging", :port=>9200, :scheme=>"http", :path=>""}
```

```
2019-03-01 11:38:40 +0000 [info]: starting fluentd 1.3.2 pid=8 ruby="2.3.3"
2019-03-01 11:38:40 +0000 [info]: spawn command to main: cmdlines["/usr/bin/ruby2.3", "-Eascii-8bit:ascii-8bit", "/fluentd/vendor/bundle/ruby/2.3.0/bin/fluentd", "-c", "/fluentd/etc/fluent.conf", "-s", "/fluentd/plugins", "--gemfile", "/fluentd/Gemfile", "--under-supervisor"]
2019-03-01 11:38:40 +0000 [info]: gem 'fluent-plugin-elasticsearch' version '2.11.11'
2019-03-01 11:38:40 +0000 [info]: gem 'fluent-plugin-kubernetes_metadata_filter' version '2.1.6'
2019-03-01 11:38:40 +0000 [info]: gem 'fluent-plugin-rewrite-tag-filter' version '2.1.1'
2019-03-01 11:38:40 +0000 [info]: gem 'fluent-plugin-systemd' version '1.0.1'
2019-03-01 11:38:40 +0000 [info]: gem 'fluentd' version '1.3.2'
2019-03-01 11:38:40 +0000 [info]: adding match pattern="fluent.**" type="null"
2019-03-01 11:38:40 +0000 [info]: adding filter pattern="kubernetes.**" type="kubernetes_metadata"
2019-03-01 11:38:40 +0000 [info]: adding match pattern="**" type="elasticsearch"
2019-03-01 11:38:40 +0000 [info]: #0 [out_es] Connection opened to Elasticsearch cluster => {:host=>"elasticsearch.logging", :port=>9200, :scheme=>"http", :path=>""}
2019-03-01 11:38:40 +0000 [info]: #0 [out_es] Detected ES 6.x: ES 7.x will only accept `doc` in type name.
2019-03-01 11:38:40 +0000 [info]: adding source type="systemd"
2019-03-01 11:38:40 +0000 [info]: adding source type="systemd"
2019-03-01 11:38:40 +0000 [info]: adding source type="systemd"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: adding source type="tail"
2019-03-01 11:38:40 +0000 [info]: #0 starting fluentd worker pid=19 ppid=8 worker=0
```

10. Nu kun je de logging gaan testen.

Installeer/run hiervoor een pod/image met nginx:

```
kubectl run nginx --image=nginx -n logging
```

```
kubectl get pods -n logging
```

NAME	READY	STATUS	RESTARTS	AGE
elasticsearch-7b4cc779b8-572f4	1/1	Running	0	43m
kibana-678774f4f6-zjl65	1/1	Running	0	39m
nginx-7cbbd8cdc9-xgth6	1/1	Running	0	36s

Definieer een port forward, om de pod van buitenaf op poort 8081 aan te kunnen roepen:

```
kubectl port-forward nginx-7cbbd8cdc9-xgth6 8081:80 -n logging
```

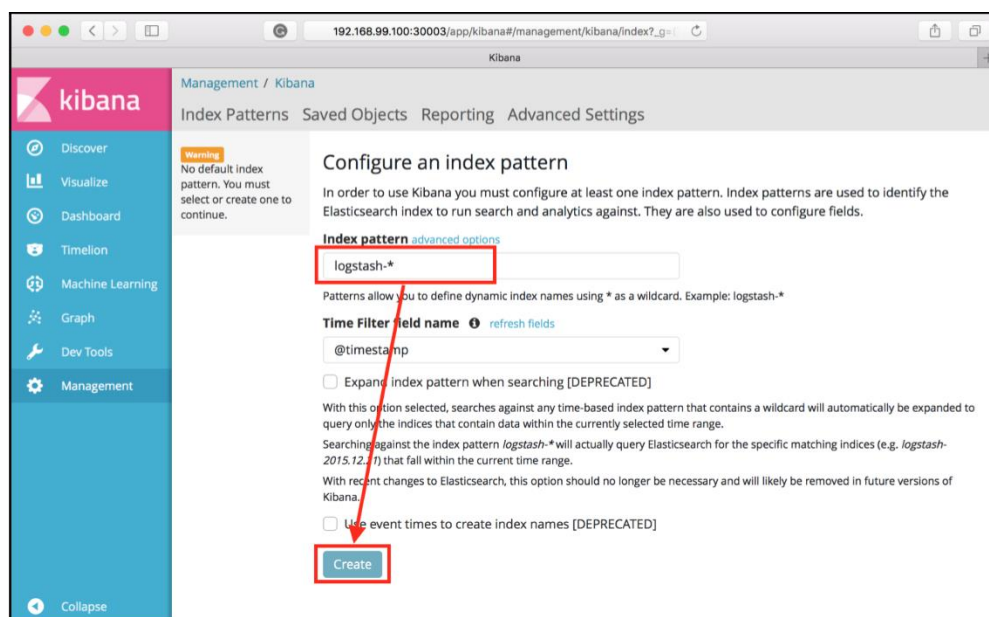
```
curl http://127.0.0.1:8081
```

- Open vervolgens het Kibana dashboard (zie stap 9, als dat niet meer geopend is).

Ga naar de tab management en kies voor index management.

Type **logstash*** in het index pattern en vervolgens **create**.

Selecteer vervolgens de optie discover en bekijk een aantal log meldingen.

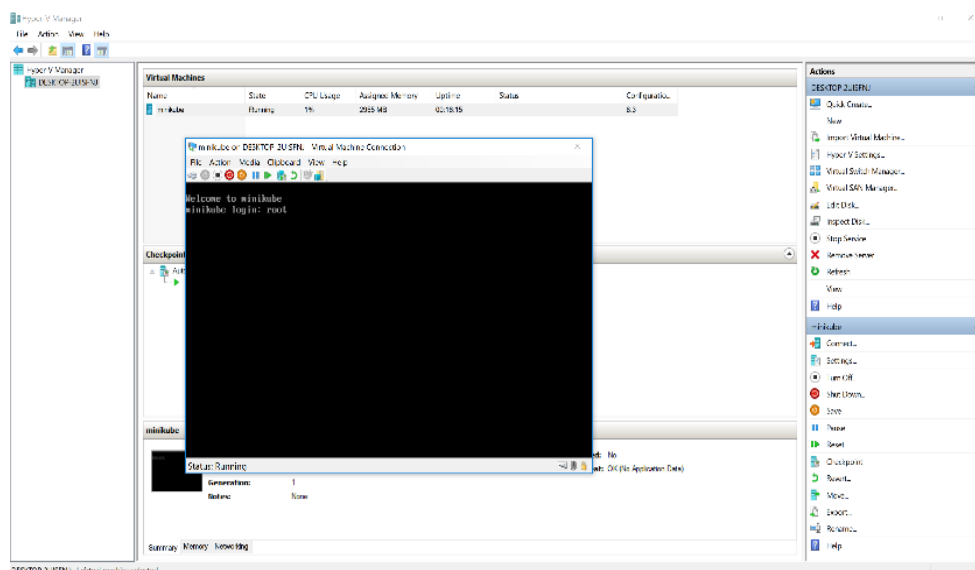


11. Nu de EFK stack up en running is, ga je de inmiddels bekende whiskeyshop service deployen op het minikube cluster.

- De whiskeyshop service die je gaat deployen staat in <https://github.com/AMIS-Services/sig-log-monitoring>. Download deze bestanden naar de computer waarop minikube draait.
- Mount de lokale bestanden in minikube, zodat deze daar gelezen kunnen worden. Het commando om dat te doen is: **minikube mount** [lokale dir]:[minikube dir] , bijvoorbeeld `minikube mount .\whiskeyshop\:/whiskeyshop`

[illegible]

- Open de minikube terminal (Hyper-V → dubbel klik op minikube → terminal window wordt geopend → log in als root)



- Navigeer in minikube naar de, via het mount statement, gekoppelde folder:

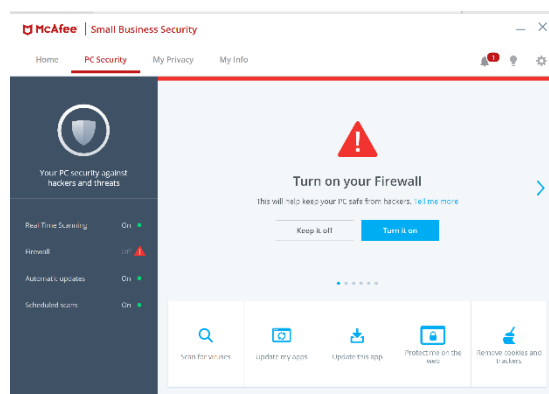
- Type **pwd** om te zien in welk pad je staat
- Type **ls -lt** om te zien welke dirs en bestanden er staan
- Type **cd ..** om 1 dir omhoog te navigeren

..

Vraag: zie je bestanden in de folder?

Als dit niet het geval is, dan is de oorzaak zeer waarschijnlijk een actieve firewall.

Zie je de bestanden wel als je de firewall hebt uitgeschakeld?



- Zorg dat je in de folder met de whiskeyshop staat en type:

docker build --tag=whiskeyshop:base --rm=true .

Als het goed is wordt er nu, op basis van de Dockerfile die in de whiskeyshop folder staat een image gemaakt en in de interne, lokale, registry geplaatst.

- Vraag het image op via: **docker images "w*"**

```
# docker images "w*"
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
whiskeyshop          base                9726b6a0b4ed       30 minutes ago
149MB
#
```

- Deploy het image als volgt naar een pod in minikube:

kubectl run whiskeyshop --image=whiskeyshop:base --port=8080 --image-pull-policy Never

- **kubectl get deployments**

- **kubectl expose deployment whiskeyshop --type=NodePort**

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	1h
whiskeyshop	NodePort	10.96.155.170	<none>	8080:32611/TCP	13s

- **minikube service list**

NAMESPACE	NAME	URL
default	kubernetes	No node port
default	whiskeyshop	http://192.168.1.112:32611

- Roep de whiskeyshop service aan op: [http://\[minikube-ip\]:\[exposed-port\]/whiskeys](http://[minikube-ip]:[exposed-port]/whiskeys),
bijvoorbeeld via: curl 192.168.1.112:32611/whiskeys

```

StatusCode      : 200
StatusDescription :
Content         : [{"id":"Jack","name":"Jack Daniels"}, {"id":"Glenn","name":"Glenfiddich"}, {"id":"Jameson","name":"Jameson"}]
RawContent      : HTTP/1.1 200
                  Transfer-Encoding: chunked
                  Content-Type: application/json; charset=UTF-8
                  Date: Fri, 01 Mar 2019 17:47:35 GMT

                  [{"id":"Jack","name":"Jack Daniels"}, {"id":"Glenn","name":"Glenfiddich"}]...
Forms           : {}
Headers         : [{"Transfer-Encoding", "chunked"}, [{"Content-Type", "application/json; charset=UTF-8"}, [{"Date", "Fri, 01 Mar 2019 17:47:35 GMT"}]}]
Images          : {}
InputFields     : {}
Links           : {}
ParsedHtml      : mshtml.HTMLDocumentClass
RawContentLength : 107

```

- Scroll door de logging in Kibana en kijk of je de aanroep van de whiskeyshop service terug kunt vinden.

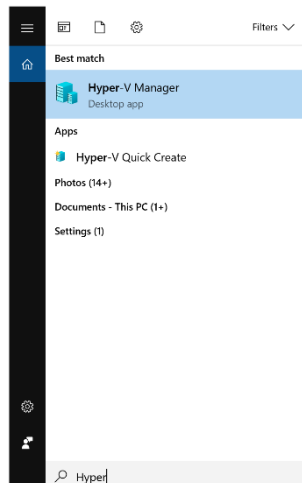
The screenshot shows the Kibana web interface. On the left is a sidebar with navigation options: Discover, Visualize, Dashboard, Timeline, Machine Learning, Graph, Dev Tools, and Management. The 'Discover' tab is selected. The main area displays a list of log entries. The selected entry is from March 1st, 2019, at 18:57:29.000. The log entry details are as follows:

Field	Value
_id	Awk6z2aaabccw5u7cn
_index	logstash-2019.03.01
_score	-
_type	Fluentd
docker.container_id	e0e8abdc2aba758f3d7818b90fc3cbbcc8e27d1720fcb89383bbeb3af53f26
kubernetes.container_name	whiskeyshop
kubernetes.host	minikube
kubernetes.labels.pod-template-hash	75d7f46b84
kubernetes.labels.run	whiskeyshop
kubernetes.master.url	https://10.96.0.1:443/api
kubernetes.namespace.name	default
kubernetes.pod_id	da55b19b-3c48-11e9-b8c7-00155d016c10
kubernetes.pod_name	whiskeyshop-75d7f46b84-xz55n
log	2019-03-01 17:57:29.785 INFO [whiskeyshop service, 9abbed30d813bc97, 9abbed30d813bc97, false] 1 --- [nio-8080-exec-9] c.a.whiskeyshop.service.WhiskeyService : Got whiskey Jameson
stream	stdout
tag	kubernetes.var.log.containers.whiskeyshop-75d7f46b84-xz55n_default_whiskeyshop-e0e8abdc2aba758f3d7818b90fc3cbbcc8e27d1720fcb89383bbeb3af53f26.log

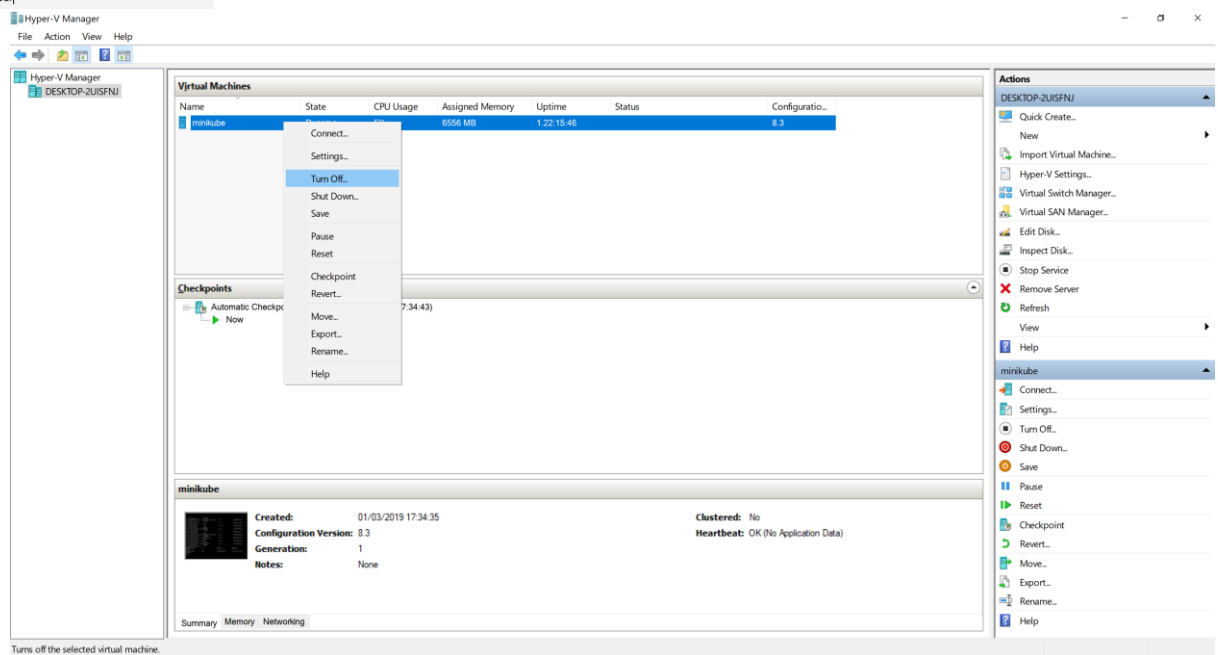
- Topper, je bent nu klaar met dit lab!

Appendix A: verwijderen Minikube

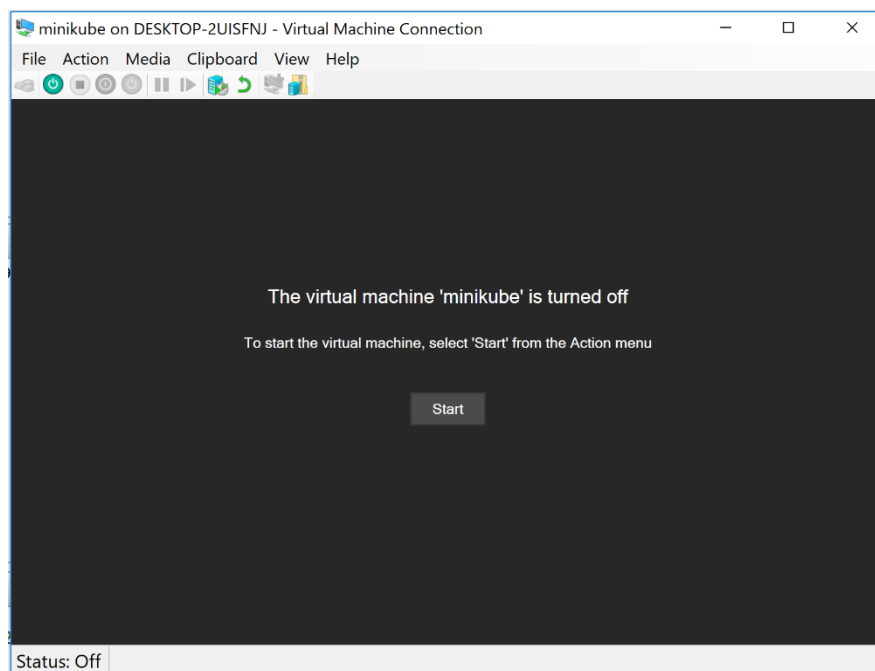
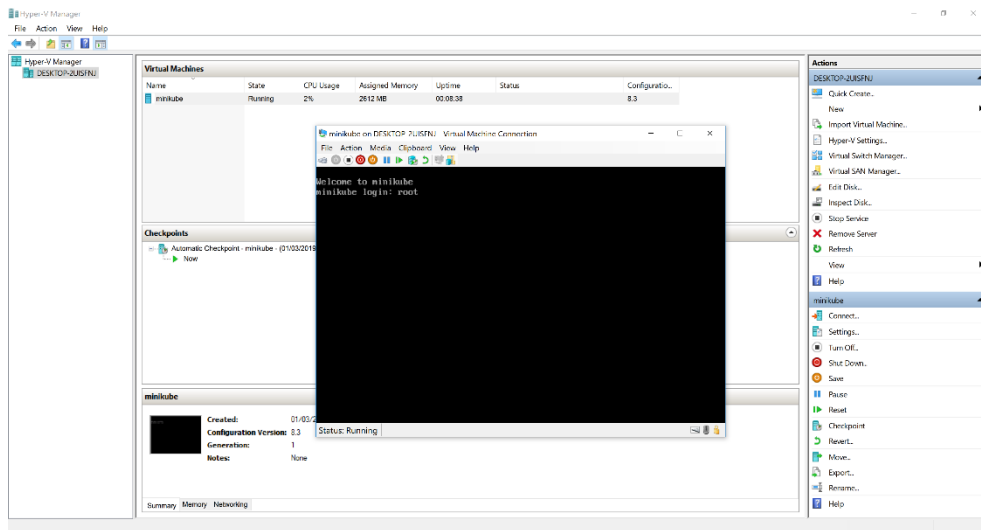
1. Open Hyper-V Manager



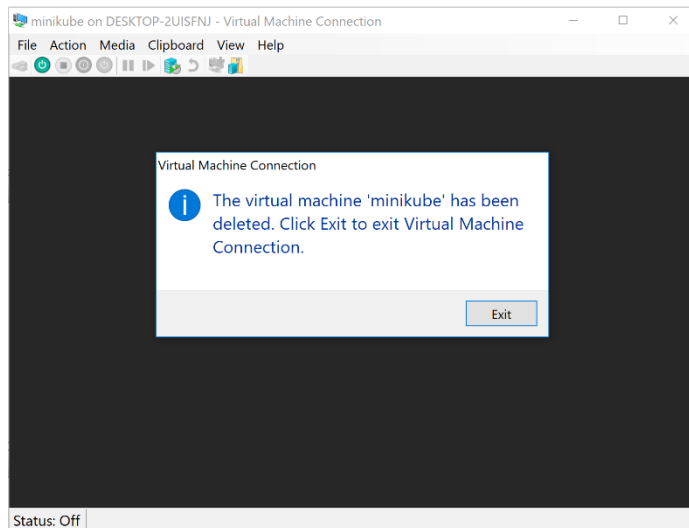
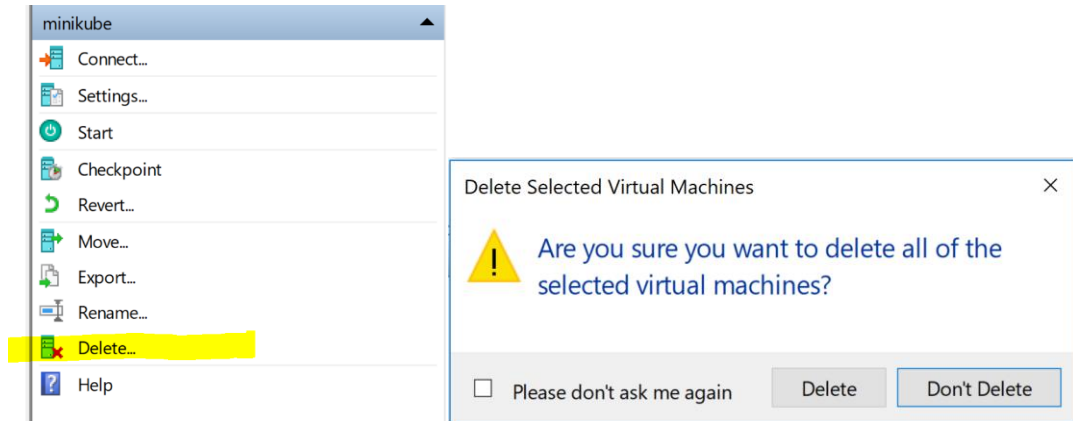
2. Open Hyper-V Manager, selecteer minikube → double klik muis



3. Een pop-up terminal window wordt geopend → login als: **root** en type: **sudo shutdown**



4. Wacht totdat minikube gestopt is (dit kan een tijdje duren). Kies daarna om minikube te verwijderen: selecteer de regel met de gestopte minikube en kies delete in het menu dat rechtsonder getoond wordt.



5. Open boxstarter of powershell in elevated mode (= met administrator rechten). Type de volgende 2 commando's om de folders .kube en .minikube te verwijderen:

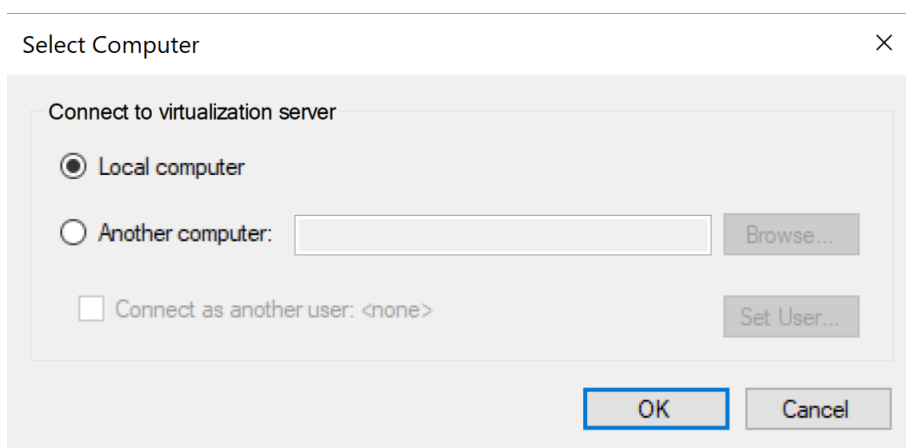
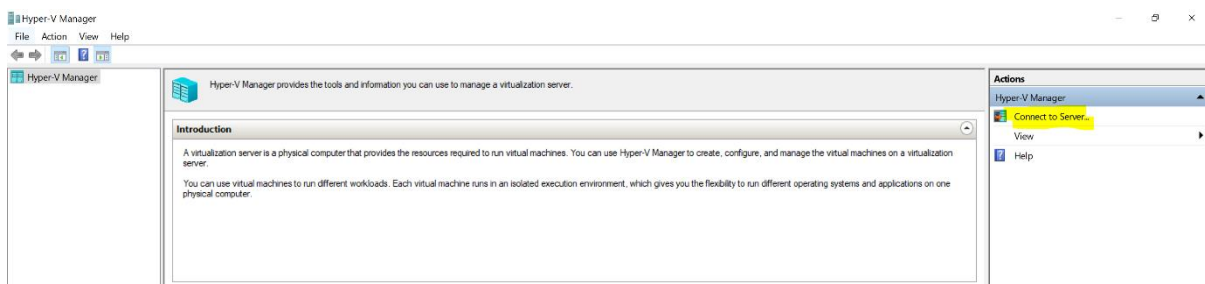
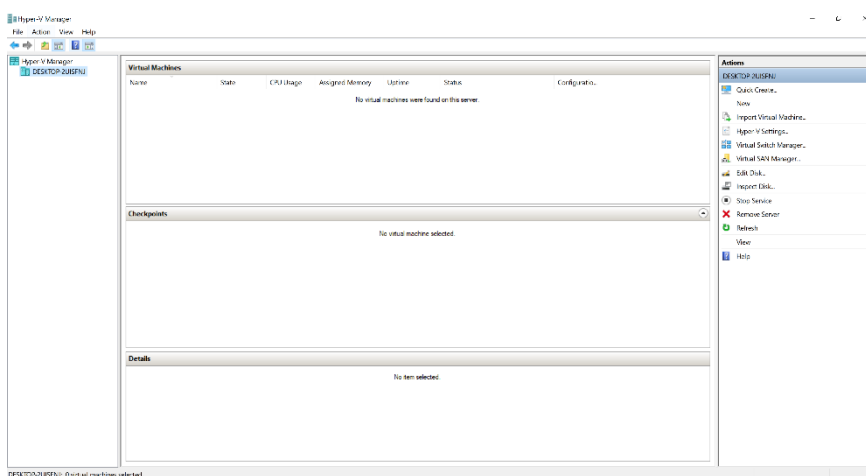
```
rd -r ~\.kube
```

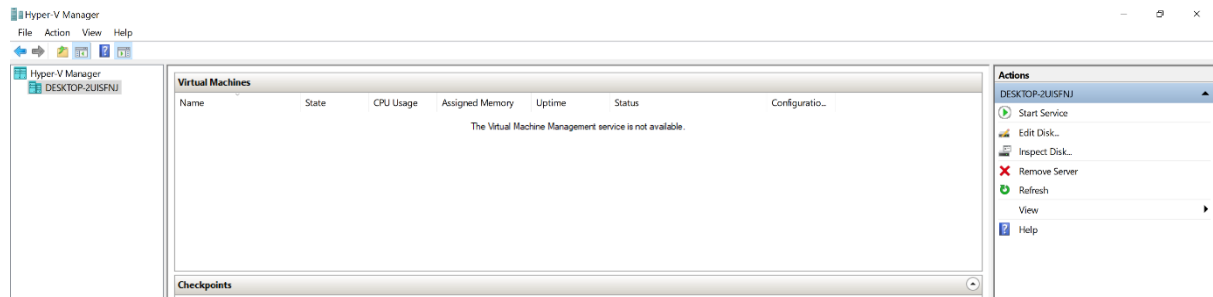
```
rd -r -force ~\.minikube
```

Als het goed is kan minikube nu opnieuw worden geïnstalleerd. Toch kan het zijn dat er tijdens de installatie van minikube (herhaaldelijk) een melding komt dat deze is gecrashed. In dat geval, herhaal de bovenstaande stappen en kies er tevens voor om de server te stoppen, deze te verwijderen en opnieuw aan te maken.

```
Boxstarter Shell
184.30 MB / 184.30 MB [=====] 100.00% 0s
! Unable to start VM: create: creating: exit status 1

* Sorry that minikube crashed. If this was unexpected, we would love to hear from you:
- https://github.com/kubernetes/minikube/issues/new
```





Appendix B: YAML files

Filenaam: **elastic.yaml**,

Wat doet het: maakt een deployment (installeert de image met de applicatie) en een service (exposed de applicatie) aan voor elasticsearch. Merk het poort nummer 9200 op, waarop met elasticsearch wordt gecommuniceerd.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: elasticsearch
spec:
  selector:
    matchLabels:
      component: elasticsearch
  template:
    metadata:
      labels:
        component: elasticsearch
    spec:
      containers:
        - name: elasticsearch
          image: docker.elastic.co/elasticsearch/elasticsearch:6.5.4
          env:
            - name: discovery.type
              value: single-node
          ports:
            - containerPort: 9200
              name: http
              protocol: TCP
      resources:
        limits:
          cpu: 500m
          memory: 4Gi
        requests:
          cpu: 500m
```

memory: 4Gi

apiVersion: v1

kind: Service

metadata:

name: elasticsearch

labels:

service: elasticsearch

spec:

type: NodePort

selector:

component: elasticsearch

ports:

- port: 9200

targetPort: 9200

Filenaam: **fluentd-daemonset.yaml**,

Wat doet het: maakt een daemon set aan (= per node één pod) met de fluentd log aggregator

```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  name: fluentd
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
    version: v1
    kubernetes.io/cluster-service: "true"
spec:
  template:
    metadata:
      labels:
        k8s-app: fluentd-logging
        version: v1
        kubernetes.io/cluster-service: "true"
    spec:
      serviceAccount: fluentd
      serviceAccountName: fluentd
      tolerations:
        - key: node-role.kubernetes.io/master
          effect: NoSchedule
      containers:
        - name: fluentd
          image: fluent/fluentd-kubernetes-daemonset:v1.3-debian-elasticsearch
          env:
            - name: FLUENT_ELASTICSEARCH_HOST
              value: "elasticsearch.logging"
            - name: FLUENT_ELASTICSEARCH_PORT
              value: "9200"
            - name: FLUENT_ELASTICSEARCH_SCHEME
              value: "http"
            - name: FLUENT_UID
```

```
    value: "0"

  - name: FLUENTD_SYSTEMD_CONF
    value: "DISABLE"

resources:
  limits:
    memory: 200Mi

  requests:
    cpu: 100m
    memory: 200Mi

volumeMounts:
  - name: varlog
    mountPath: /var/log

  - name: varlibdockercontainers
    mountPath: /var/lib/docker/containers
    readOnly: true

terminationGracePeriodSeconds: 30

volumes:
  - name: varlog
    hostPath:
      path: /var/log

  - name: varlibdockercontainers
    hostPath:
      path: /var/lib/docker/containers
```

Filenaam: **fluentd-rbac.yaml**,

Wat doet het: maakt een Role Based Access Controll aan. Deze zorgt er voor dat pods in andere namespaces (anders dan namespace kube-system) kunnen worden benaderd.

```
apiVersion: v1
```

```
kind: ServiceAccount
```

```
metadata:
```

```
  name: fluentd
```

```
  namespace: kube-system
```

```
---
```

```
apiVersion: rbac.authorization.k8s.io/v1beta1
```

```
kind: ClusterRole
```

```
metadata:
```

```
  name: fluentd
```

```
  namespace: kube-system
```

```
rules:
```

```
- apiGroups:
```

```
  - ""
```

```
resources:
```

```
- pods
```

```
- namespaces
```

```
verbs:
```

```
- get
```

```
- list
```

```
- watch
```

```
---
```

```
kind: ClusterRoleBinding
```

```
apiVersion: rbac.authorization.k8s.io/v1beta1
```

```
metadata:
```

```
  name: fluentd
```

```
roleRef:
```

```
  kind: ClusterRole
```

```
  name: fluentd
```


apiGroup: rbac.authorization.k8s.io

subjects:

- kind: ServiceAccount

name: fluentd

namespace: kube-system

Filenaam: **fluentd-service.yaml**,

Wat doet het: maakt een service aan voor fluentd (zodat deze extern benaderd kan worden).

apiVersion: v1

kind: Service

metadata:

name: fluentd

labels:

service: fluentd

spec:

type: NodePort

selector:

run: fluentd

ports:

- port: 24224

targetPort: 24224

Filenaam: **node-deployment.yaml**,

apiVersion: extensions/v1beta1

kind: Deployment

metadata:

name: node

labels:

name: node

spec:

replicas: 1

template:

metadata:

labels:

app: node

spec:

containers:

- name: node

image: fluentd-node-sample:latest

imagePullPolicy: Never

restartPolicy: Always

Filenaam: **kibana.yaml**,

Wat doet het: maakt een deployment en een service aan voor kibana. Merk de URL naar elastic search op.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: kibana
spec:
  selector:
    matchLabels:
      run: kibana
  template:
    metadata:
      labels:
        run: kibana
    spec:
      containers:
        - name: kibana
          image: docker.elastic.co/kibana/kibana:6.5.4
          env:
            - name: ELASTICSEARCH_URL
              value: http://elasticsearch:9200
            - name: XPACK_SECURITY_ENABLED
              value: "true"
          ports:
            - containerPort: 5601
              name: http
              protocol: TCP
```

```
apiVersion: v1
kind: Service
metadata:
  name: kibana
labels:
```

service: kibana

spec:

type: NodePort

selector:

run: kibana

ports:

- port: 5601

targetPort: 5601