KUBESAFE

THOMAS MORE
2025

Design RAPPORT

BPU2

Alexander Conserva
Bent Van den Eynde
Chris van Uitert
Jaron Janssens
Maximus Muzika
Milan Dils

KUBESAFE

Table of CONTENTS

Inleiding

Veel bedrijven en ontwikkelaars hebben een werkende PHP-applicatie, maar missen de technische kennis om deze veilig en efficiënt te hosten. Het vinden van een betrouwbare hosting oplossing kan dan een uitdaging zijn.
Wij bieden een self-hosted cluster in het datacenter van Thomas More, speciaal ontworpen voor gebruikers met beperkte hosting ervaring. Onze oplossing combineert gebruiksvriendelijkheid met geavanceerde beveiliging, redundantie en schaalbaarheid, zodat applicaties klaar zijn voor de toekomst.

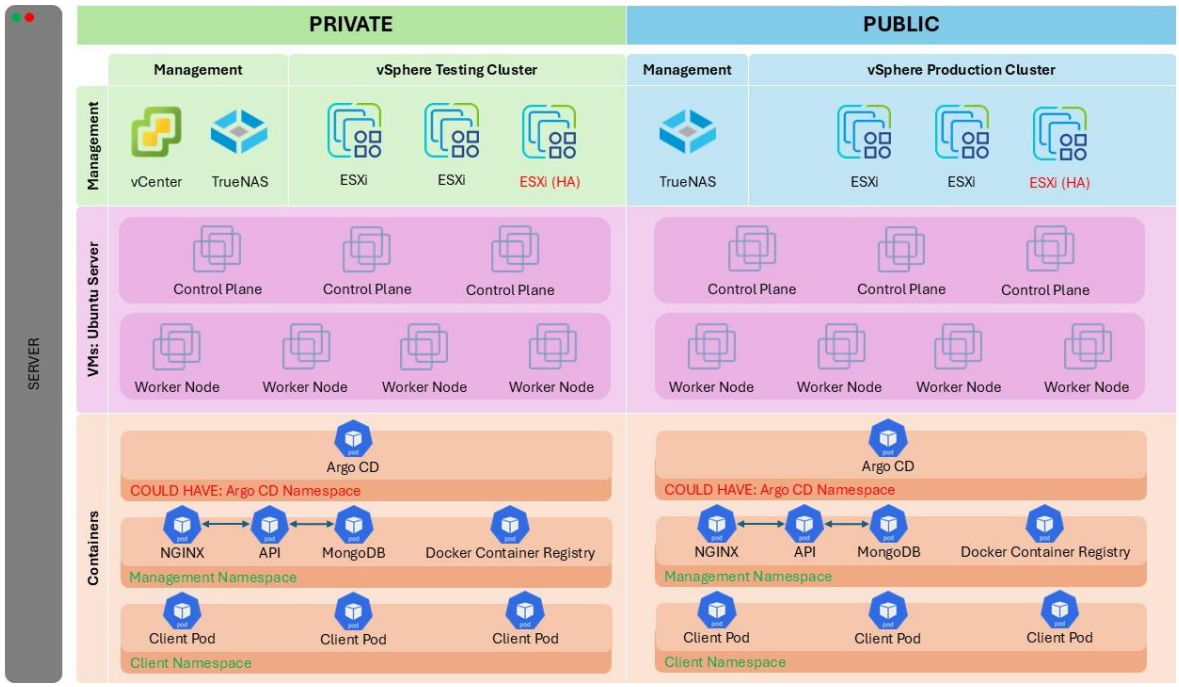
Globaal Schema

Onze oplossing is opgedeeld in twee omgevingen: Private en Public. De Private omgeving fungeert als testomgeving waarin we applicaties verifiëren voordat ze publiek beschikbaar worden. Hier beheren we ook de vSphere- en Kubernetes-clusters. De Public omgeving bevat het product cluster, waar applicaties toegankelijk zijn voor klanten.

De infrastructuur is opgebouwd uit drie lagen:

1. **Managementlaag** – Bevat de vSphere clusters en opslagoplossingen.
2. **Virtuele machine laag** – Vormt de backbone van onze Kubernetes-cluster.
3. **Container-/servicelaag** – Bevat alle containers en services die nodig zijn voor de applicaties.

In dit document bespreken we deze drie lagen verder in detail.



Functionele Eisen

Must-Have:

- Hosting Platform voor shared hosting
- PHP applicatie kunnen hosten

Should-Have:

- Support voor diverse relationele databanken
- Beheerders dashboard

Could-Have:

- GIT upload implementatie
- Procedurebeschrijving

Won’t-Have:

- Geen aanvullende eisen op dit moment

Niet-Functionele Eisen

Must-Have:

- Beveiliging via security framework
- Schaalbaarheid (d.m.v. Kubernetes)

Should-Have:

- Automatisatie
- Redundantie/backups
- Gebruiksvriendelijkheid

Could-Have:

- Hoge uptime
- Applicatie performantie

Won’t-Have:

- Geen aanvullende eisen op dit moment

Laag 1 design beschrijft onze virtualisatie laag. Ons gehele project draait op één enkele server in het datacenter van Thomas More Geel. Op deze server draait VMware ESXi 8. Deze opzet vormt voor ons een fysieke beperking. Idealiter zouden we drie fysieke servers hebben, waarbij elke server zijn eigen ESXi-hypervisor draait. Om deze beperking te omzeilen, simuleren we de situatie door de bovenliggende ESXi buiten ons project te beschouwen. Het project zelf omvat de volgende onderdelen:

Private

Het private gedeelte van ons platform bevat het management en de testing omgeving. Deze omgeving is alleen voor ons team en mag niet toegankelijk zijn voor derden. Ook plaatsen we hier een TrueNAS voor network storage voor het hele project. Hier zijn 3 ESXi-hypervisors in 1 private cluster zodat we gebruik kunnen maken en kunnen experimenteren met VMware High Availability. Samengevat zit er in het private gedeelte het volgende:

1x VMware vCenter 8 (management)
3x VMware ESXi 8 (testing cluster)
1x TrueNAS Server (private network storage)

Public

Het publieke gedeelte is bedoeld voor de klant. Hier draaien wij ons platform, waarmee de klant interactie heeft en waar het publiek toegang heeft tot de applicaties op ons platform. Om redundantie te waarborgen, maken we hier 1 public cluster van drie ESXi-hypervisors. Hier draait ook een TrueNAS server voor de publieke storage zodat deze gescheiden blijft van de private cloud. Samengevat maken we het volgende voor de productie omgeving:

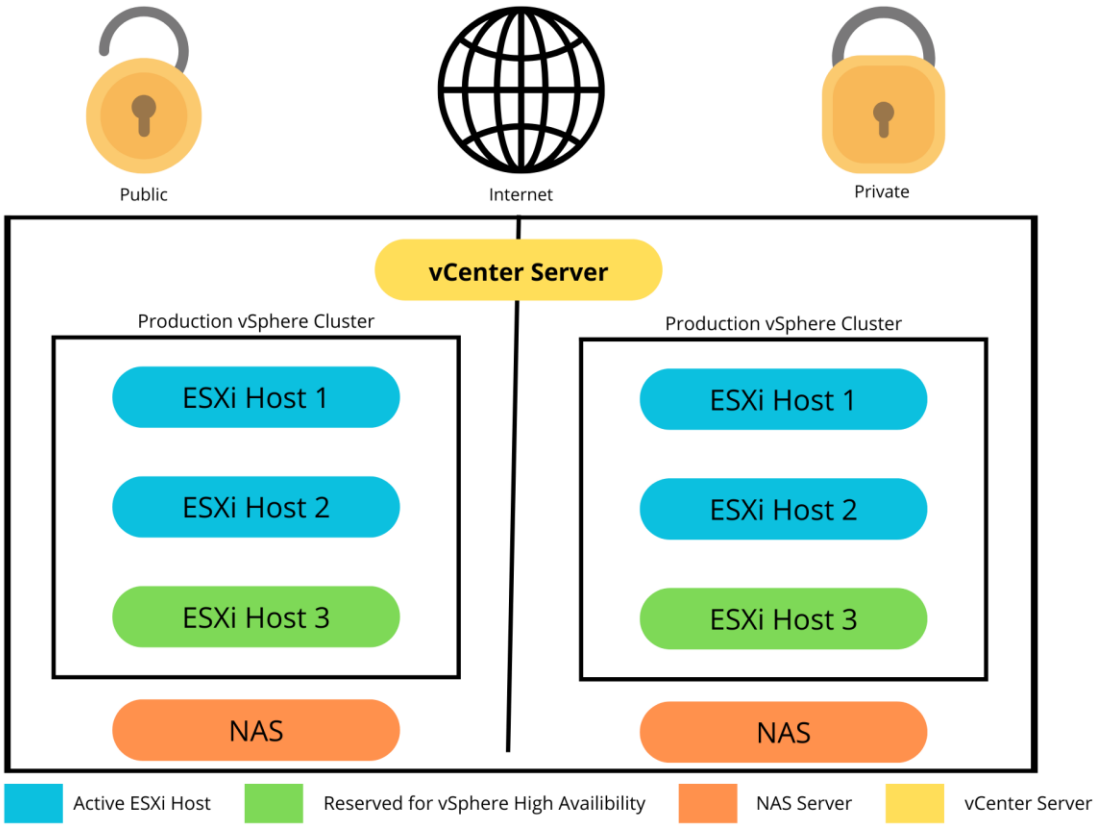
3x VMware ESXi 8 (productie)
1x TrueNAS Server (public network storage)

Redundantie

In deze setup hebben we 6 ESXi-hypervisors en 2 clusters, waardoor we gebruik kunnen maken van VMware High Availability en andere VMware services. We voorzien een NAS storage zodat onze ESXi's aan shared storage kunnen deelnemen. Wanneer een host faalt, kunnen de andere ESXi's aan de network storage.

Security

We scheiden de private en de public laag. Dit doen we om ervoor te zorgen dat klanten of derden geen toegang krijgen tot de private cluster. Uiteraard kunnen de clusters wel met elkaar communiceren zodat wij logging en management kunnen uitvoeren.


Public
Internet
Private
vCenter Server
Production vSphere Cluster
ESXi Host 1
ESXi Host 2
ESXi Host 3
NAS
Production vSphere Cluster
ESXi Host 1
ESXi Host 2
ESXi Host 3
NAS
Active ESXi Host
Reserved for vSphere High Availibility
NAS Server
vCenter Server

Laag 2 beschrijft de virtuele machine laag. Hier leggen we uit welke workloads op de hypervisors draaien en hoe we onze vSphere-clusters met virtuele machines inrichten. Daarnaast beschrijven we ook de opzet van onze Kubernetes-clusters.

Ubuntu Server 24.04.2 LTS

De volgende stap is het opzetten van de Virtuele Machines. Dit gaan we doen op basis van een template zodat we maar één keer deze moeten configureren en aan hardening moeten doen. Op deze VM's draaien we ons Kubernetes cluster. 1/3 van de ESXi-servers wordt gereserveerd voor High Availability (HA). Daarom moeten we deze infrastructuur op twee ESXi-servers per vSphere cluster implementeren, en laten we ESXi 3 leeg:

ESXi 1:

1x Ubuntu Server VM: Kubernetes Control Plane Node + Argo CD
1x Ubuntu Server VM: Kubernetes Control Plane Node
2x Ubuntu Server VM: Kubernetes Worker Node

ESXi 2:

1x Ubuntu Server VM: Kubernetes Control Plane Node
2x Ubuntu Server VM: Kubernetes Worker Node

ESXi 3 (High Availability)

Kubernetes Kubeadm Cluster

Voor de implementatie van ons cluster moeten we een geschikte Kubernetes-oplossing kiezen. Verschillende opties zijn beschikbaar, elk met hun eigen voordelen en toepassingen. We hebben de volgende oplossingen vergeleken:

Minikube: Gebruikt voor lokale ontwikkeling en testen. Het is eenvoudig, snel en ideaal voor het opzetten van een lokale Kubernetes-testomgeving. Voordelen: snelle installatie, lage resource vereisten.

RKE (Rancher Kubernetes Engine): Ingezet voor productie clusters met geavanceerde functies. Het biedt flexibiliteit en integreert goed met Rancher voor cluster beheer. Voordelen: schaalbaarheid en compatibiliteit met zowel on-premise als cloud-omgevingen.

KubeOne: Toegepast voor grote, geautomatiseerde clusters. Dit platform automatiseert het beheer van Kubernetes-clusters op meerdere platforms en werkt naadloos samen met Terraform. Voordelen: hoge schaalbaarheid en eenvoudige automatisering.

Kubeadm: Gebruikt voor situaties waarin maximale controle over Kubernetes vereist is. Geschikt voor experts die alles zelf willen configureren. Voordelen: maximale flexibiliteit, maar complex in gebruik.

We kiezen voor Kubeadm omdat het een betere en efficiënte manier is om Kubernetes in te richten. Het bevat geen overbodige functies en heeft uitgebreide documentatie. We implementeren twee clusters: één voor de private omgeving en één voor de public omgeving. Dit cluster draait op Ubuntu Server VM's zoals hierboven beschreven.

Gebruik van GitOps voor Cluster Beheer

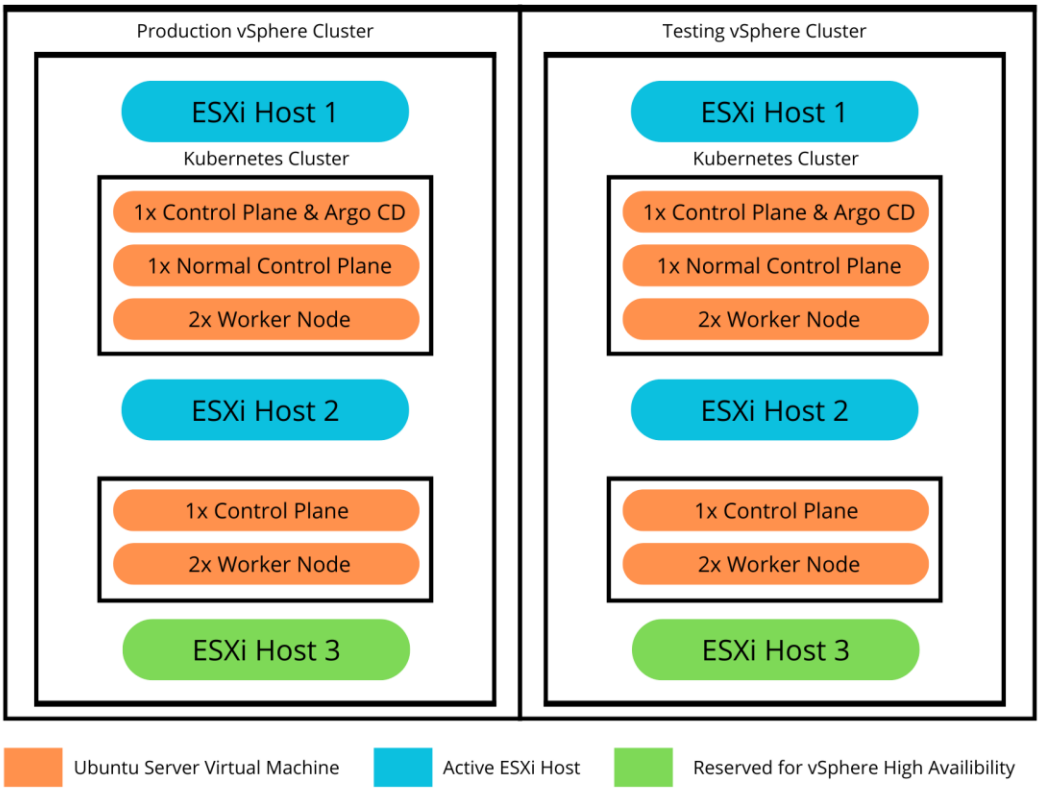
Om Kubernetes-clusters efficiënt en gestructureerd te beheren, maken we gebruik van een GitOps-tool. Dit stelt ons in staat om klant clusters te beheren via een Git-repository, waardoor wijzigingen gecontroleerd en geautomatiseerd worden doorgevoerd. We maken om een doordachte keuze te maken een vergelijking van de volgende beschikbare tools:

Argo CD: Een eenvoudige GitOps-tool die speciaal is ontworpen voor Kubernetes en automatisch YAML-bestanden uit Git synchroniseert met je Kubernetes-cluster. Het biedt een gebruiksvriendelijke UI waarmee je deployments kunt monitoren en problemen kunt oplossen, en het ondersteunt verschillende deployment-strategieën zoals rolling updates, blue/green en canary deployments. Kortom, Argo CD is perfect voor teams die een visuele, Kubernetes-native GitOps-oplossing zoeken.

Flux CD: Is een lichtgewicht GitOps-tool die naadloos werkt met Kubernetes en YAML-bestanden uit Git synchroniseert zonder extra overhead. Het heeft geen complexe gebruikersinterface, maar biedt krachtige CLI-functionaliteit waarmee je efficiënt kunt werken. Flux CD is ideaal voor teams die een minimalistische, CLI-gebaseerde GitOps-oplossing willen die zich richt op efficiëntie en eenvoud.

Jenkins X: Combineert GitOps met geautomatiseerde CI/CD-pijplijnen en automatiseert builds, tests en deployments terwijl het applicaties promoveert tussen omgevingen zoals dev, staging en production. Het is geschikt voor teams die al bekend zijn met Jenkins of een alles-in-één oplossing nodig hebben die ook GitOps ondersteunt. Jenkins X ondersteunt deployment-strategieën zoals rolling updates en blue/green deployments, waardoor het een uitstekende keuze is voor teams die een volledige CI/CD-oplossing met GitOps-functionaliteit nodig hebben.

Voor het beheer van onze Kubernetes-deployments kiezen we uiteindelijk voor Argo CD. Deze GitOps-tool biedt een eenvoudige en betrouwbare synchronisatie tussen onze Git-repositories en Kubernetes-clusters, met een gebruiksvriendelijke interface voor eenvoudig beheer. Argo CD wordt geïmplementeerd op één van de control plane-machines, waarbij slechts één instance per vSphere-cluster nodig is. Dankzij vSphere High Availability wordt de vereiste redundantie automatisch gewaarborgd, waardoor de beschikbaarheid en betrouwbaarheid van onze infrastructuur gegarandeerd blijven.


Production vSphere Cluster
ESXi Host 1
Kubernetes Cluster
1x Control Plane & Argo CD
1x Normal Control Plane
2x Worker Node
ESXi Host 2
1x Control Plane
2x Worker Node
ESXi Host 3
Testing vSphere Cluster
ESXi Host 1
Kubernetes Cluster
1x Control Plane & Argo CD
1x Normal Control Plane
2x Worker Node
ESXi Host 2
1x Control Plane
2x Worker Node
ESXi Host 3
Ubuntu Server Virtual Machine
Active ESXi Host
Reserved for vSphere High Availibility

In deze paragraaf beschrijven we welke applicaties en services we binnen ons Kubernetes-cluster implementeren. Om een duidelijke scheiding tussen verschillende functies duidelijk te maken, hebben we het cluster opgedeeld in drie afzonderlijke namespaces:

Management Namespace:

De Management Namespace bevat essentiële componenten die nodig zijn voor de infrastructuur en het beheer van onze applicaties. Binnen deze namespace draaien:

- Onze API-services, die de communicatie tussen de webserver en andere systemen faciliteren.
- Een Docker registry, waarmee we container-images opslaan en beheren.
- De database, waarin de benodigde gegevens voor onze applicaties worden opgeslagen en beheerd.

Client Namespace:

De Client Namespace is specifiek bedoeld voor klantgerichte applicaties. Binnen deze namespace worden de client pods gehost, waarin de applicaties van onze klanten draaien. Dit zorgt ervoor dat de klantomgevingen goed gescheiden blijven van de beheersystemen en interne infrastructuur.

ArgoCD Namespace:

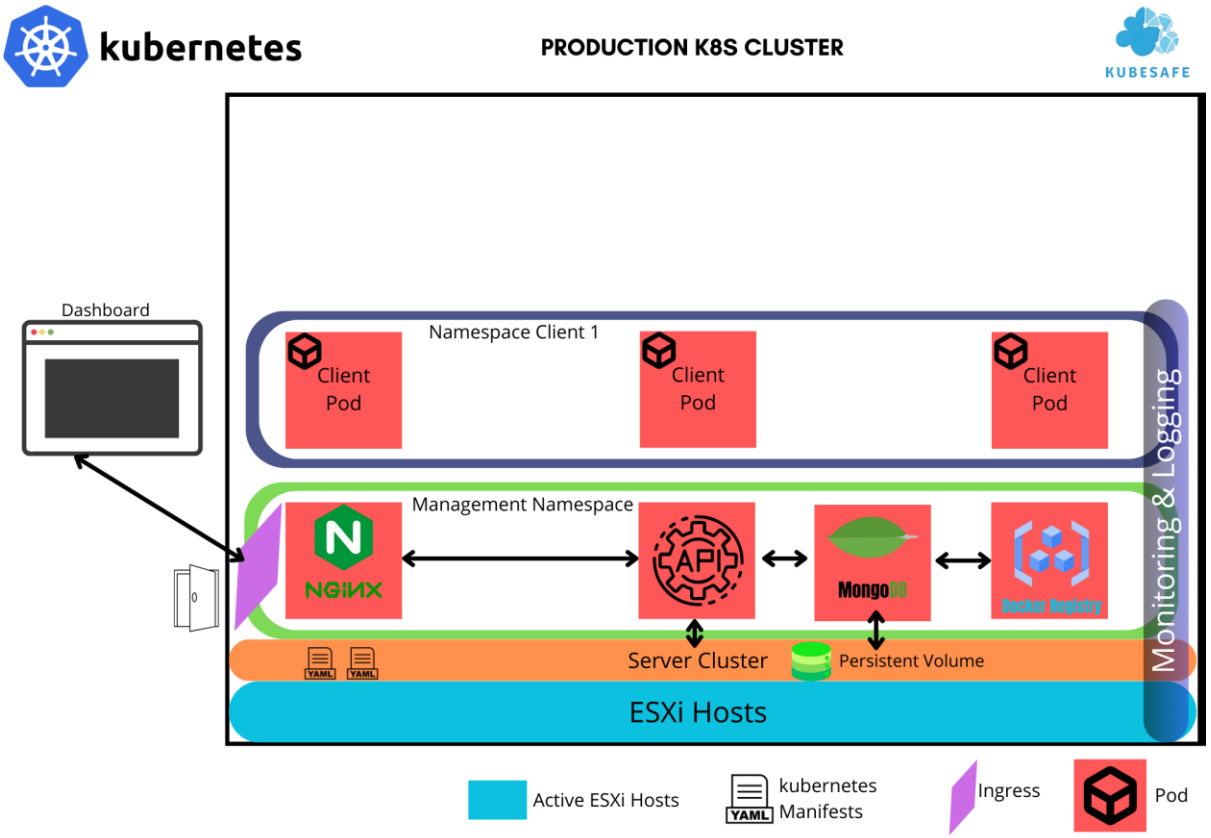
Tot slot implementeren we binnen de ArgoCD Namespace tools voor continue integratie en deployment. Hier streven we ernaar om ArgoCD en Helm te implementeren, zodat we het Kubernetes-cluster op een efficiënte en geautomatiseerde manier kunnen beheren.

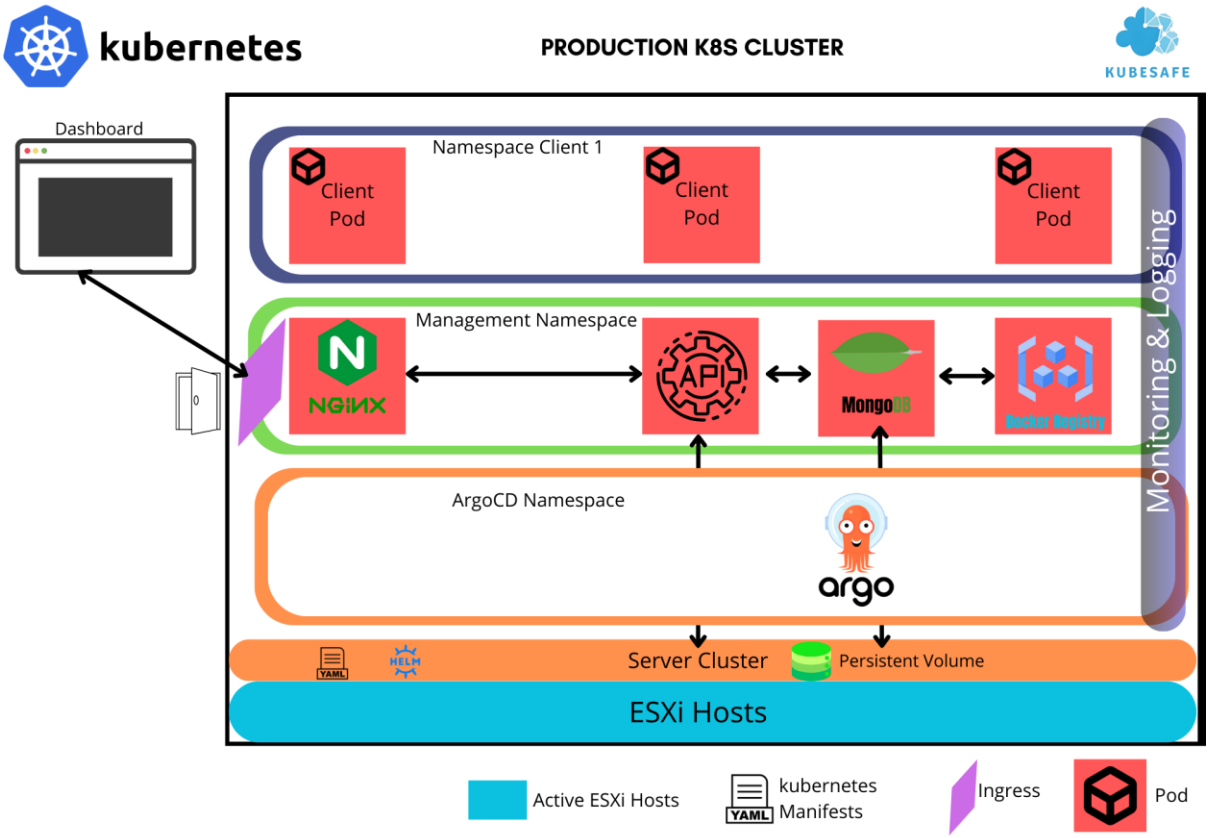
- ArgoCD stelt ons in staat om GitOps-principes toe te passen, waardoor applicatie-updates en configuratiewijzigingen via een versiebeheersysteem kunnen worden beheerd.
- Helm vereenvoudigt het beheren en uitrollen van Kubernetes-applicaties met behulp van templates en charts.

YAML

In de beginfase richten we ons op het handmatig schrijven van Docker-manifests binnen de YAML-configuratiebestanden. Daarnaast implementeren we lichte automatisering met Bash-scripts om herhalende taken te vereenvoudigen en de implementatie te stroomlijnen.

Naarmate de infrastructuur groeit, verschuiven we naar een high-level automatisering aanpak. Dit houdt in dat we deployments automatisch genereren op basis van templates en een dashboard. Hierdoor wordt het beheer van Kubernetes-clusters efficiënter, vermindert de kans op fouten en versnelt het uitrollen van nieuwe applicaties.


kubernetes
PRODUCTION K8S CLUSTER
KUBESAFE
Dashboard
Namespace Client 1
Client Pod
Client Pod
Client Pod
Management Namespace
NGINX
API
MongoDB
Docker Registry
Monitoring & Logging
Server Cluster
Persistent Volume
ESXi Hosts
Active ESXi Hosts
kubernetes Manifests
Ingress
Pod


kubernetes
PRODUCTION K8S CLUSTER
KUBESAFE
Dashboard
Namespace Client 1
Client Pod
Client Pod
Client Pod
Management Namespace
NGINX
API
MongoDB
Docker Registry
Monitoring & Logging
ArgoCD Namespace
argo
Server Cluster
Persistent Volume
ESXi Hosts
Active ESXi Hosts
kubernetes Manifests
Ingress
Pod

CIS Control 1 - Enterprise Asset Management:

- 1.1: Lansweeper wordt ingezet voor het automatisch bijhouden van een actuele en gedetailleerde inventaris van alle netwerkapparaten.
- 1.3: Lansweeper fungeert als actieve discovery tool om nieuwe apparaten automatisch toe te voegen aan de inventaris.

CIS Control 2 - Software Asset Management:

- 2.1: Lansweeper beheert ook de software-inventaris en controleert op beschikbare updates. Handmatig beheer voor niet-gedetecteerde software.
- 2.2: Lansweeper monitort de ondersteuning van software en markeert end-of-life (EOL) software als ongeautoriseerd, tenzij er uitzonderingen zijn.

CIS Control 3 - Data Protection:

- 3.2: Gevoelige data wordt geïdentificeerd en beheerd met Apache Atlas. Minimaal jaarlijkse evaluatie en toegangscontrole implementatie.
- 3.3: Toegangscontrole in MongoDB via Role-Based Access Control (RBAC), periodieke controles en netwerkbeveiliging via VPN/firewall.

CIS Control 4 - Secure Configuration:

- 4.4: Firewall Beheer via vSphere Client, met logging en periodieke tests.
- 4.8: Overbodige diensten in Kubernetes en ESXi worden geïdentificeerd en uitgeschakeld voor extra beveiliging.

CIS Control 5 - Account Management:

- 5.1: Accounts worden beheerd in een MongoDB-container met versleutelde wachtwoorden en automatische kwartaalcontroles.
- 5.2: Wachtwoordbeleid met minimaal 8 tekens en verplichte Multi-Factor Authentication (MFA).

CIS Control 6 - Access Control Management:

- 6.4 & 6.5: MFA wordt geïmplementeerd met Google Authenticator en Authelia voor remote toegang.
- 6.8: Role-Based Access Control (RBAC) wordt beheerd met OpenLDAP en Keycloak.

CIS Control 7 - Continuous Vulnerability Management:

- 7.4: Patchbeheer met Ansible.
- 7.5: Interne vulnerability scans met OpenVAS, Nikto en Lynis.
- 7.6: Externe scans met OWASP ZAP en Wapiti.

CIS Control 8 - Audit Log Management:

- 8.2: Audit logs worden verzameld met Auditd en Syslog-ng.
- 8.9: Logbeheer via ELK Stack, Graylog en Fluentd.
- 8.11: Loganalyse met Wazuh, GoAccess en Osquery.

CIS Control 10 - Malware Defenses:

- 10.1: Lynis wordt ingezet voor anti-malware en security auditing.
- 10.2: Automatische updates voor Lynis en regelmatige audits.

CIS Control 11 - Data Recovery:

- 11.1: Gedocumenteerd data herstelproces met versleutelde back-ups op een geïsoleerde server.
- 11.2: Wekelijkse geautomatiseerde back-ups met integriteitscontroles.

Samenvatting Tools:

CIS	Tool Naam	Functie
1	Lansweeper	Inventory Management
2	EOL Monitoring	Automatische monitoring van Lansweeper met foutmelding
3	Apache Atlas	Gevoelige data voor classificatie
6	Authelia	MFA
	Keycloak	RBAC
7	Ansible	Patch management
	OpenVAS	Internal Scans
	OWASP ZAP	External Scans
8	Auditd & Syslog-ng	Log collectie
	Graylog	Centralized Logging
	Wazuh	Audit log Reviews
10	Lynis	Anti-malware

Brown, K. (2023, 13 maart). KubeADM vs Minikube, Pros and cons. linuxconfig.org. Geraadpleegd op
21 maart 2025, via https://linuxconfig.org/kubeadm-vs-minikube-pros-and-cons

Lecallier, C. (2023, 19 december). Kubeadm vs MiniKube, Kind & K3S: Kubernetes tools | Padok.
minikube-kubeadm-kind-k3s. Geraadpleegd op 21 maart 2025, via
https://cloud.theodo.com/en/blog/minikube-kubeadm-kind-k3s

Creating a cluster with kubeadm. (2024, 16 oktober). Kubernetes. Geraadpleegd op 21 maart 2025,
via https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/

Team, C. O. (2024, 19 augustus). Argo CD. Codefresh. Geraadpleegd op 21 maart 2025, via https://codefresh.io/learn/argo-cd/

Team, C. O. (2024a, juli 22). Argo CD vs. Flux: 6 Key Differences and How to Choose.

Codefresh.

Geraadpleegd op 21 maart 2025, via
https://codefresh.io/learn/argo-cd/argo-cd-vs-flux-6-key-differences-and-how-to-choose/

Why Kubermatic? (z.d.). Kubermatic. Geraadpleegd op 21 maart 2025, via
https://www.kubermatic.com/products/kubermatic-kubernetes-platform/why-kubermatic/

Overview of RKE | RKE1. (2023, 15 maart). RKE. Geraadpleegd op 21 maart 2025, via
https://rke.docs.rancher.com/

Welcome! (z.d.). Minikube. Geraadpleegd op 21 maart 2025, via
https://minikube.sigs.k8s.io/docs/

Introducing Jenkins X: a CI/CD solution for modern cloud applications on Kubernetes. (2018, 19
maart). Introducing Jenkins X: a CI/CD solution for modern cloud applications on Kubernetes.
Geraadpleegd op 21 maart 2025, via https://www.jenkins.io/blog/2018/03/19/introducing-jenkins-x/