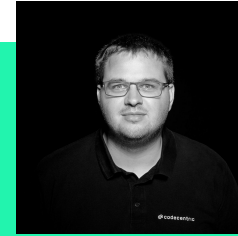


A man with a beard, wearing a black and green plaid shirt, is sitting at a desk in an office. He is looking up and to the right with a slight smile. In front of him is a laptop. To his left, there is a blue water bottle and some papers. In the background, there is a large monitor displaying a web application. The office has large windows and a modern design.

@codecentric

OpenShift
Installation &
Administration

Vorstellung



Tobias Derksen

- DevOps Consultant
- OpenShift Architect
- Red Hat Certified Engineer

Tag 1

@codecentric

Einführung
Cluster Konzeption
Installation
IPI & UPI
Custom Config
OpenShift CLI & Console
Cluster Updates





Einführung in OpenShift



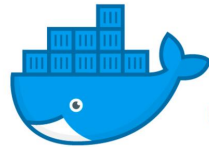
kubernetes



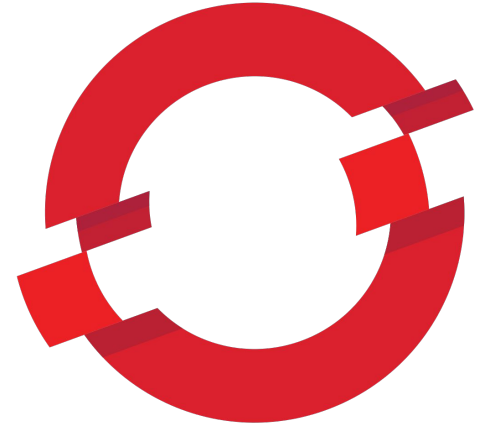
cri-o



redhat





docker





OPENSIFT

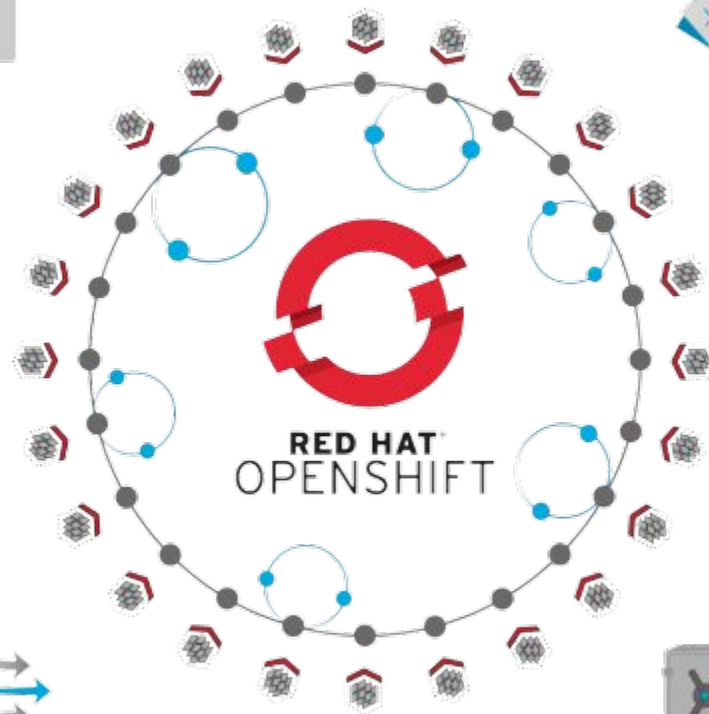
Self-Service 

Multi-language 

Automation 


Collaboration 

Multi-tenant 




 Standards-based

 Web-scale

 Open Source

 Enterprise Grade

 Secure

OpenShift ist ... kubernetes plus

- Routing
- Monitoring
- Logging
- Web Console (GUI)
- Image Builds
- Image Registry
- Built-in security
- Networking (SDN)

Mit Red Hat Subscription



Trusted Registry



Security Newsletter



Some extra Components
(Storage, Serverless, Virtualization, etc)



Enterprise Support

Begriffe

- Container
- Pod
- Node
- Project
- Namespace
- etcd

Cluster Konzeption

Node Roles



Bootstrap Node

- nur während der Installation
- stellt initiale Konfiguration für master nodes bereit
- wird nach der Installation gelöscht



Master Nodes

- control-plane
- Master API (control-plane)
- etcd
- Alles was für den Cluster wichtig ist



Worker Nodes

- Ingress Controller
- Image Registry
- Web Console
- User Workload

Minimale Anforderungen

hyper-threading cores are sufficient!

Operating System: **RH CoreOS**



Bootstrap Node

- 4 vCPU
- 16GB Memory
- 120GB Storage



Master Nodes

- 1 Node
- 4 vCPU
- 16GB Memory
- 120GB Storage



Worker Nodes

- 2 vCPU
- 8GB Memory
- 120GB Storage

Empfohlene Anforderungen

Control plane node sizing

The control plane node resource requirements depend on the number of nodes in the cluster. The following control plane node size recommendations are based on the results of control plane density focused testing.

Number of worker nodes	CPU cores	Memory (GB)
25	4	16
100	8	32
250	16	96



Bootstrap Node

- 4 vCPU
- 16GB Memory
- 120GB Storage



Master Nodes

- 3 Nodes
- 4 vCPU
- 16GB Memory
- 120GB Storage








Worker Nodes

- at least 2 Nodes
- at least 4 vCPU
- at least 16GB Memory
- 120GB Storage
increase storage on
higher resources

Und jetzt?

Wie viele Ressourcen brauche ich nun wirklich?

- Wie groß der Cluster sein muss, hängt von vielen verschiedenen Faktoren ab!
- Über diese Faktoren sollte man sich im Vorfeld Gedanken machen und dokumentieren
- Die Cluster Ressourcen müssen kontinuierlich überwacht werden um drohende Knappheit frühzeitig entgegen zu wirken
- Generell gilt, mehr Ressourcen pro Node ist effizienter als mehr Nodes (Stichwort: overhead)

-  Erwarteter Workload / Anforderungen der Applikationen
-  Hochverfügbarkeit
-  Automatische Skalierung
-  Cluster Updates
-  Capacity Reserven

Cluster Limits

Maximum type	3.x tested maximum	4.x tested maximum
Number of nodes	2,000	2,000
Number of pods ^[1]	150,000	150,000
Number of pods per node	250	500 ^[2]
Number of pods per core	There is no default value.	There is no default value.
Number of namespaces ^[3]	10,000	10,000
Number of builds	10,000 (Default pod RAM 512 Mi) - Pipeline Strategy	10,000 (Default pod RAM 512 Mi) - Source-to-Image (S2I) build strategy
Number of pods per namespace ^[4]	25,000	25,000
Number of services ^[5]	10,000	10,000
Number of services per namespace	5,000	5,000
Number of back-ends per service	5,000	5,000
Number of deployments per namespace ^[4]	2,000	2,000



Installation

IPI Installation

Installer-provisioned Infrastructure (IPI)

- Der Installer erstellt und startet die Maschinen
- Benötigt eine Automatisierung auf Infrastrukturebene (z.B. ein Hypervisor)
- Ggf. müssen vorher manuell Sachen vorbereitet werden (hier genau die Installationsanleitung aus der Doku beachten)
- Der Cluster ist dann in der Lage mit der entsprechenden API zu kommunizieren und diverse Automaten anzubieten (z.B. Storage provisioning, Scaling, Health Checks, etc)



Diverse Cloud Anbieter:
AWS, Microsoft Azure, Google Cloud



Red Hat Virtualization (oVirt)
Red Hat OpenStack Platform (RHOSP)



VMware vSphere
(Version 6.5 or higher)



Bare Metal



IBM Z / LinuxONE / Power Systems
(Besonderheiten beachten!)

UPI Installation

User-provisioned Infrastructure (UPI)

- Alle Infrastruktur Komponenten müssen manuell provisioniert werden
- Der Installer generiert die notwendigen Konfigurationen (Ignition-Configs)
- Die Nodes müssen mit CoreOS images gestartet werden und initialisieren sich dann selbst



Nodes



DNS



IP Configuration
(statische IPs oder DHCP)



Load Balancer



Storage

Installation Parameter

und worüber man sich Gedanken machen sollte

- Base Domain
- Cluster Name
- Pull Secret
- FIPS Mode



Restricted Network?



Internet Anbindung über Proxy?



Company CA certificate



Mirrored Registry

DNS Einträge

api. <cluster-name>.<base-domain>	A or CNAME	Public API Access
api-int. <cluster-name>.<base-domain>	A or CNAME	API access for nodes
*.apps. <cluster-name>.<base-domain>	A or CNAME	Ingress wildcard domain

Loadbalancer

- **API Loadbalancer**
 - API Port: 6443
 - MachineAPI Port: 22623
- **Application Loadbalancer**
 - HTTP Port: 80
 - HTTPS Port: 443

```
frontend api
  bind *:6443
  default_backend masters

backend masters
  balance source
  server master1 10.0.1.11:6443 check
  server master2 10.0.1.12:6443 check
  server master3 10.0.1.13:6443 check

frontend maschineconfig
  bind *:22623
  default_backend mastersmc

backend mastersmc
  balance source
  server master1 10.0.1.11:22623 check
  server master2 10.0.1.12:22623 check
  server master3 10.0.1.13:22623 check

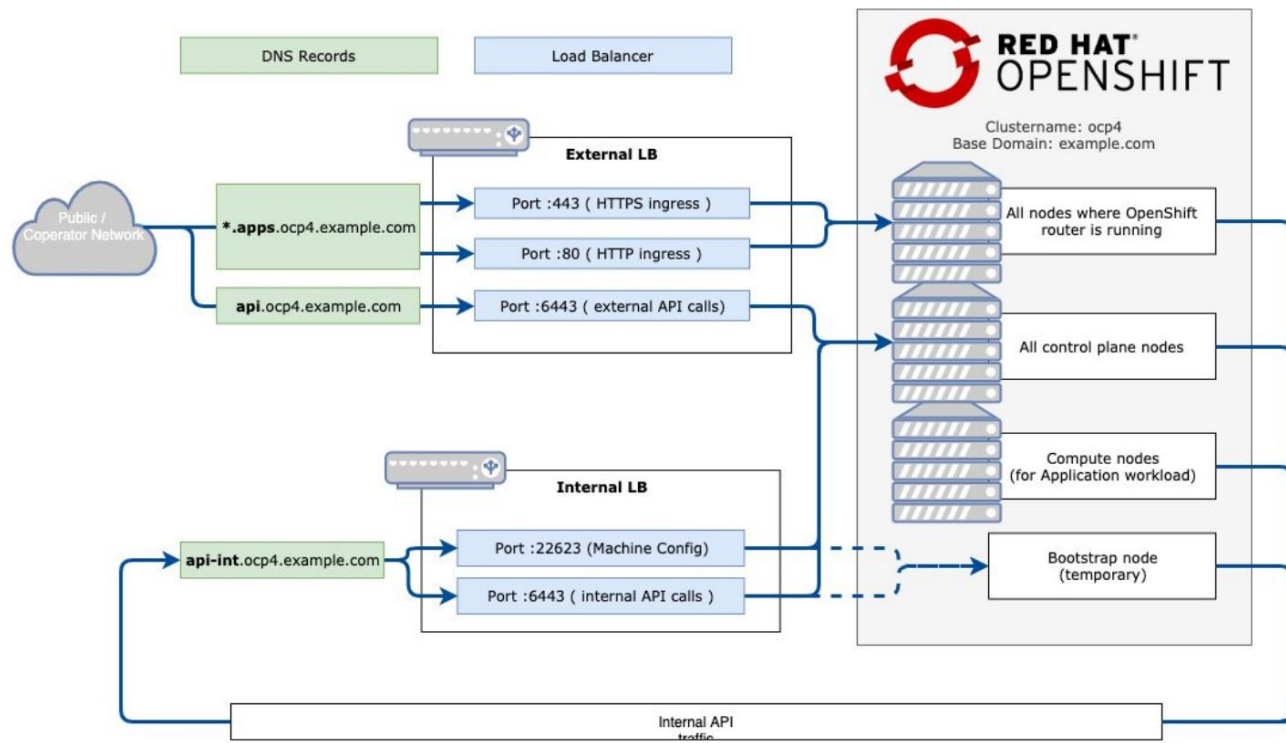
frontend router-https
  bind *:443
  default_backend worker-https

backend worker-https
  balance source
  server worker1 10.0.1.21:443 check
  server worker2 10.0.1.22:443 check
  server worker3 10.0.1.23:443 check

frontend router-http
  bind *:80
  default_backend worker-http

backend worker-http
  balance source
  server worker1 10.0.1.21:80 check
  server worker2 10.0.1.22:80 check
  server worker3 10.0.1.23:80 check
```

OpenShift 4 DNS & Load Balancer Overview



Netzwerk Anforderungen



Machine Network

- Node Network
- can be company network
- does not need to be separated



Cluster Network

- SDN Network
- controlled by Network Plugin
- hostPrefix (default: 23)
- default: 10.128.0.0/14



Service Network

- Cluster Services
- default: 172.30.0.0/16

Installer Commands

- openshift-install create cluster
- openshift-install create install-config
- openshift-install create manifests
- openshift-install create ignition-configs
- openshift-install wait-for bootstrap-complete
- openshift-install wait-for install-complete
- openshift-install destroy bootstrap
- openshift-install destroy cluster

Install Config

- Cloud / Hypervisor Config
- Machine customizations
- Network customizations
- FIPS mode
- Pull Secret
- SSH Key

```
apiVersion: v1
baseDomain: example.com 1
compute:
- hyperthreading: Enabled 2 3
  name: worker
  replicas: 0 4
controlPlane:
  hyperthreading: Enabled 2 3
  name: master 3
  replicas: 3 5
metadata:
  name: test 6
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 7
    hostPrefix: 23 8
  networkType: OpenShiftSDN
  serviceNetwork: 9
  - 172.30.0.0/16
platform:
  none: {} 10
fips: false 11
pullSecret: '{"auths": ...}' 12
sshKey: 'ssh-ed25519 AAAA...' 13
```

Pitfalls

- Installation muss innerhalb von 24 Stunden nach Generierung der Ignition Configs erfolgen
- Der Cluster darf die ersten 24 Stunden nicht ausgeschaltet werden
- Fehlende PTR Records
- Installationskonfiguration wird vom Installer geschluckt
- zu kleine IP Range für Node Networks (hostPrefix)



API Ressourcen

Alles nur Ressourcen ...

- Der komplette Zustand des Clusters wird durch verschiedene Ressourcen abgebildet
- Cluster Ressourcen
- Namespace Ressourcen
- Die Ressourcen werden im etcd gespeichert
- Die API selbst ist stateless
- Custom Resource Definitions (CRD)

```

1 kind: Pod
2 apiVersion: v1
3 metadata:
4   name: apiserver-5868544846-8rzfh
5   namespace: openshift-apiserver
6   labels:
7     app: openshift-apiserver-a
8   annotations:
9     create: 'false'
10 spec:
11   containers:
12     - name: openshift-apiserver-check-endpoints
13       command:
14         - cluster-kube-apiserver-operator
15         - check-endpoints
16       env:
17         - name: POD_NAME
18           valueFrom:
19             fieldRef:
20               apiVersion: v1
21               fieldPath: metadata.name
22         - name: POD_NAMESPACE
23           valueFrom:
24             fieldRef:
25               apiVersion: v1
26               fieldPath: metadata.namespace
27       ports:
28         - name: check-endpoints
29           containerPort: 17698
30           protocol: TCP
31       image: quay.io/openshift-release-dev/ocp-v4.0-art-dev
32   status:
33     containerStatuses:
34       - restartCount: 0
35         started: true
36         ready: true
37         name: openshift-apiserver
38     phase: Running
39

```

Wichtige Resource Typen

- Namespace
- Project
- ClusterRole
- User
- PersistentVolume
- Node
- Pod
- Deployment
- ConfigMap
- Secret
- Service
- Route

@codecentric

OpenShift CLI



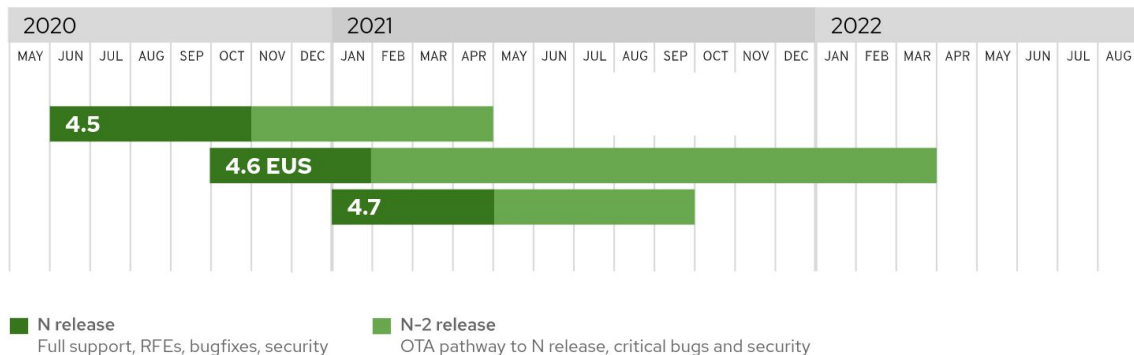
OpenShift Console (GUI)

@codecentric

Cluster Updates

Over-the-air Updates

- etwa alle 2 Wochen kommt ein Minor Update (z-stream)
- Rolling Update -> bedingt reboot aller Nodes
- cluster-version-operator aktualisiert alle Cluster Operatoren
- Die Operatoren aktualisieren dann die Komponenten
- CoreOS Update



Update Channels

- stable
- fast
- candidate
- eus (extended-update-support) (4.6 only)

Tag 2

@codecentric

Operatoren
User & Rechte Mgmt
Hochverfügbarkeit
Skalierung
Logging
Monitoring



@codecentric

Operatoren

Cluster Operatoren

- Operatoren managen und überwachen die Konfiguration aller Komponenten
- Prüft kontinuierlich ob der gewünschte Zustand dem aktuellen Zustand entspricht
- Wichtige Operatoren
 - apiserver-operator
 - etcd-operator
 - network-operator
 - machine-config-operator

User Operatoren

- Nachträglich installierter Operator
- Operator Marketplace (operatorhub.io)
- Operator Lifecycle Manager (OLM)
- Benötigt cluster-admin Rechte

OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase commercial software through [Red Hat Marketplace](#). You can your clusters to provide optional add-ons and shared services to your developers. After installation, the Operator capabilities will appear in the [Developer Catalog](#) providing a self experience.

All Items

Filter by keyword...

Community

3scale API Management
provided by Red Hat

3scale Operator to provision
3scale and publish/manage APIs

Community

akka

Akka Cluster Operator
provided by Lightbend, Inc.

Run Akka Cluster applications on
Kubernetes.

Community

Apache Spark Operator
provided by radanalytics.io

An operator for managing the
Apache Spark clusters and
intelligent applications that spa...

Community

WSO2

API Operator for Kubernetes
provided by WSO2

API Operator provides a fully
automated experience for cloud-
native API management of...

Community

APIcast
provided by Red Hat

APIcast is an API gateway built on
top of NGINX. It is part of the
3scale API Management Platform

Community

Apicurio Registry Operator
provided by Apicurio

Deploy and manage Apicurio
Registry on Kubernetes.

Install State
☐ Installed (0)
☐ Not Installed (134)

@codecentric

User Management

Overview

- OpenShift hat keine vollständige Userverwaltung
- User müssen durch externen Identity Provider authentifiziert werden
- Gruppen -> Liste von Usern
- Rechte können direkt an User oder an Gruppen vergeben werden

OpenShift Identity Provider



Statisch

- kubeadmin
- htpasswd



Directory Service

- LDAP
- Basic Auth
- Request Header



OAuth

- Github
- Gitlab
- Google
- OpenID Connect

LDAP Group Sync

- Mapping von LDAP Gruppen auf OpenShift Gruppen
- Manuelle Konfiguration
- Manuelle Synchronisierung

```
1 kind: LDAPSyncConfig
2 apiVersion: v1
3 url: ldap://LDAP_SERVICE_IP:389
4 insecure: false
5 groupUIDNameMapping:
6   "cn=admins,ou=groups,dc=example,dc=com": Administrators
7   "cn=users,ou=groups,dc=example,dc=com": Users
8 rfc2307:
9   groupsQuery:
10     baseDN: "ou=groups,dc=example,dc=com"
11     scope: sub
12     derefAliases: never
13     pageSize: 0
14   groupUIDAttribute: dn
15   groupNameAttributes: [ cn ]
16   groupMembershipAttributes: [ member ]
17   usersQuery:
18     baseDN: "ou=users,dc=example,dc=com"
19     scope: sub
20     derefAliases: never
21     pageSize: 0
22   userUIDAttribute: dn
23   userNameAttributes: [ mail ]
```

```
oc adm groups sync --sync-config=config.yaml --confirm
```

Emergency User

- **kubeadmin**
(wird bei der Installation generiert)
- **eigene Client CA**
(was im CN steht wird als Username genommen)
- **system:admin kubeconfig**
(wird vom Installer generiert)
- **eigener Service Account**
(kubeconfig generieren und abspeichern)

Rollen & Rechte

RBAC (role based access control)

- Zugriffsrechte im Cluster werden über Rollen gesteuert
- Rollen bestehen aus einer Auflistung von Rechten
- Rechte bestehen aus einem Verb und ein Ressourcetyp
(zum Beispiel: get pods)

Default Rollen:

- cluster-admin
- cluster-reader
- self-provisioner
- admin
- edit
- view

Roles & RoleBindings

- **Role** -> in einzelnen Namespaces
- **ClusterRole** -> im ganzen Cluster
- **RoleBinding** -> bindet Role an Subject (namespace bezogen)
- **ClusterRoleBinding** -> bindet ClusterRole an Subject

- ClusterRoles können auch in RoleBindings genutzt werden
- Zulässige Subjects: User, Group, ServiceAccount

```
1 kind: ClusterRoleBinding
2 apiVersion: rbac.authorization.k8s.io/v1
3 metadata:
4   name: cluster-admin
5 subjects:
6   - kind: Group
7     apiGroup: rbac.authorization.k8s.io
8     name: 'system:masters'
9 roleRef:
10   apiGroup: rbac.authorization.k8s.io
11   kind: ClusterRole
12   name: cluster-admin
13
```

Achtung: OpenShift akzeptiert RoleBindings auch wenn die Role und / oder das Subject nicht existiert

ServiceAccounts

- Technische User des Clusters
- Ordner: `/var/run/secrets/kubernetes.io/serviceaccount`
- Beinhaltet:
 - Cluster Root Zertifikat
 - Cluster Service Zertifikat
 - Access Token für ServiceAccount
 - Aktueller Namespace
- API URL: `https://kubernetes.default.svc`

@codecentric

Hochverfügbarkeit

Hochverfügbarkeit

- Zwingend erforderlich:
 - 3 Master
 - 2 Worker
- etcd nutzt Raft Algorithmus zur Vermeidung von Split-Brain
- Es darf maximal 1 Master ausfallen!
- Reserve Capacity überwachen!!!!



Anzahl der Nodes



Reserve Cluster Capacity



Load Balancer & DNS



Persistent Storage



externe Image Sources

Hochverfügbarkeit im Cluster

- Replicas der Pods
- Resource Allocation / Quality of Service
- PodDisruptionBudget
- Health Checks

```
1 kind: PodDisruptionBudget
2 apiVersion: policy/v1beta1
3 metadata:
4   name: router-default
5   namespace: openshift-ingress
6 spec:
7   selector:
8     matchLabels:
9       app: router
10   maxUnavailable: 50%
11   minAvailable: 2
12
```

Health Checks

- **Liveness Probe**

Checks whether the container is alive

If fail, container is restarted

- HTTP GET
- Shell command
- Open TCP ports

- **Readiness Probe**

Checks whether the container is able to accept traffic

If fail, container will not get any traffic from service layer

Failing is a totally valid option

- Expect that any pod is killed by kubernetes at any time
- Allow your container to fail ... as early as possible

Reasons why a pod is killed:

- Manual interaction (Admin, Developer, etc)
- Node failure or maintenance
- Network issues
- Pod / Container out-of-memory
- **Node out-of-memory**



Machine Management

Machine Management Teil 1



Node

- Entitäten auf denen Workload läuft
- Wird vom Scheduler genutzt
- Kann Bedingungen für Workload haben



Machine

- 1-1 Mapping zu Node
- Repräsentiert eine Maschine
- Enthält alle Parameter dieser Instanz



MachineSet

- Menge an Maschinen mit gleichen Parametern
- Wird zum skalieren des Cluster benutzt

Machine Management Teil 2



MachineHealthCheck

- Erkennt kaputte Maschinen
- Provisioniert neue Maschinen wenn HealthCheck fails



MachineConfig

- einzelne Ignition Configs
- Möglichkeit eigene Software oder Konfigurationen einzubauen



MachineConfigPool

- Mehrere Machine Configs
- Kümmt sich um Konfiguration der Nodes
- Änderung erzwingt Neuinitialisierung der Nodes

Custom Machine Config

```
1  apiVersion: machineconfiguration.openshift.io/v1
2  kind: MachineConfig
3  metadata:
4    name: 99-testfile
5    labels:
6      machineconfiguration.openshift.io/role: worker
7  spec:
8    config:
9      ignition:
10       version: 3.1.0
11      storage:
12       files:
13       - contents:
14         source: data:text/plain;charset=utf-8;base64,RGllcyBpc3QgZWluIFRlc3QK
15         group:
16           name: root
17         mode: 420
18         overwrite: true
19         path: /etc/testfile
20         user:
21           name: root
```

Skalierung

Skalierung über MachineSets

- einstellen der gewünschten Anzahl an Nodes pro MachineSet
- Dauer der Provisionierung ist von Provider abhängig

Cluster Autoscaler & Machine Autoscaler

- einstellen der gewünschten Anzahl an Nodes pro MachineSet
- Dauer der Provisionierung ist von Provider abhängig

Limitierungen

- Pods werden nicht automatisch verschoben (Descheduler!)
- Cluster macht kein Lastausgleich zwischen den Nodes

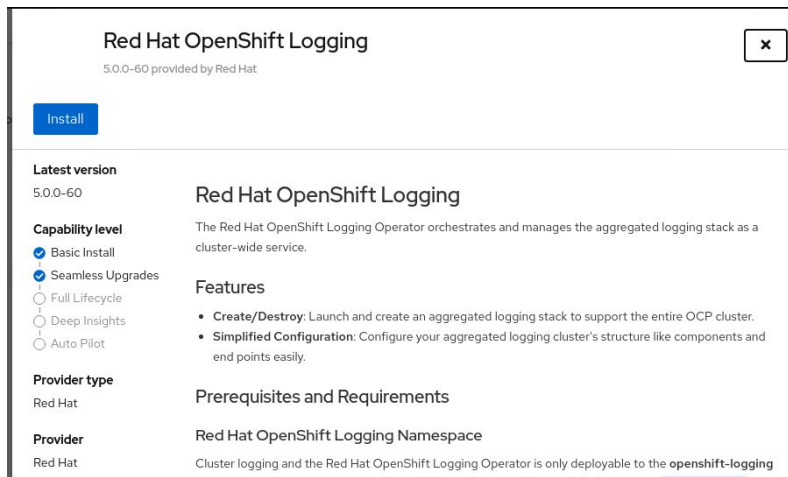
Logging

Logging?

- Container loggen nach stdout / stderr
- Logs hängen am Pod. Pod weg = Logs weg
- Cluster Logs
- Audit Log

OpenShift Logging Stack

- Fluentd
- ElasticSearch
- Kibana



- OpenShift Logging Operator (nur mit Subscription)

Alternativen

- Grafana Loki
- ELK - Elasticsearch / Logstash / Kibana / Beats

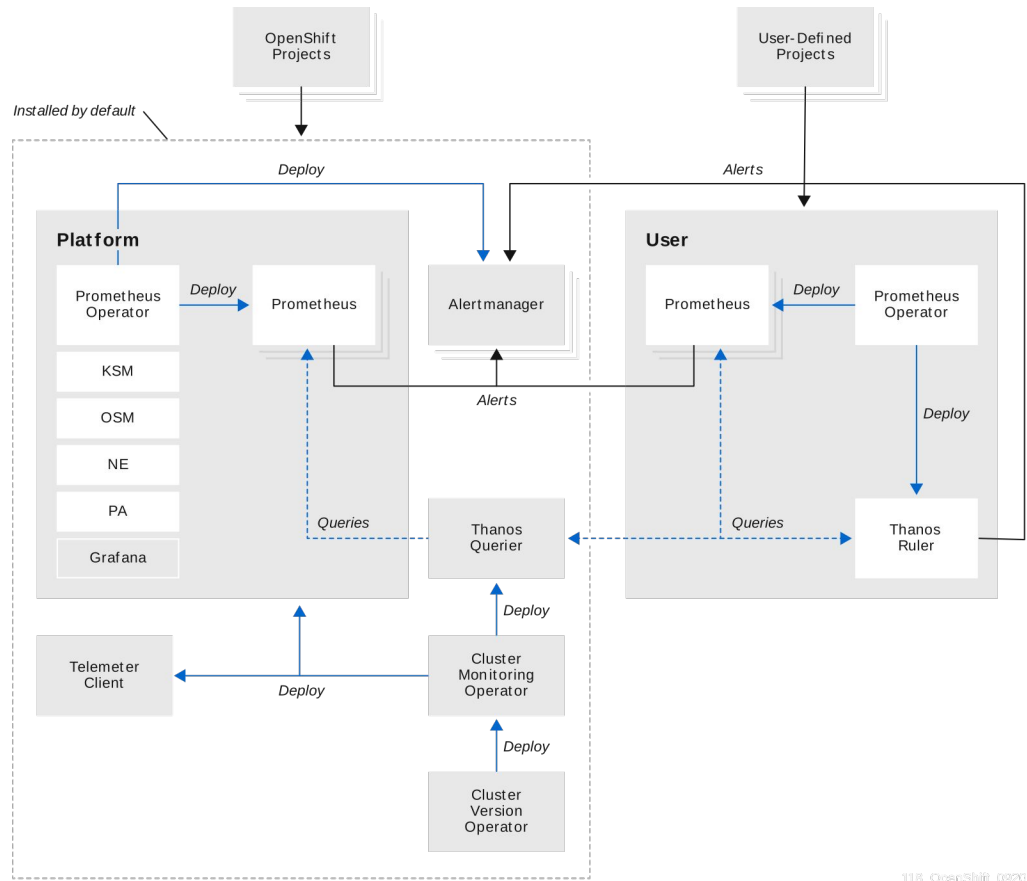


elastic



@codecentric

Monitoring



118_OpenShift_0920

Cluster Monitoring Config

- ConfigMap
- Name: cluster-monitoring-config
- Namespace: openshift-monitoring

```
1 kind: ConfigMap
2 apiVersion: v1
3 metadata:
4   name: cluster-monitoring-config
5   namespace: openshift-monitoring
6 data:
7   config.yaml: |
8     enableUserWorkload: false
9     alertmanagerMain:
10       volumeClaimTemplate:
11         metadata:
12           name: alertmanager
13         spec:
14           storageClassName: block-storage
15           resources:
16             requests:
17               storage: 1Gi
18   prometheusK8s:
19     retention: 21d
20     volumeClaimTemplate:
21       metadata:
22         name: prometheus
23       spec:
24         storageClassName: block-storage
25         resources:
26           requests:
27             storage: 100Gi
```

Monitoring eigener Services

- UserWorkloadMonitoring muss aktiviert sein
- ServiceMonitor
- PodMonitor

```
1  apiVersion: monitoring.coreos.com/v1
2  kind: ServiceMonitor
3  metadata:
4    name: alertmanager
5    namespace: openshift-monitoring
6  spec:
7    endpoints:
8      - interval: 30s
9        port: web
10       scheme: http
11    namespaceSelector: {}
12    selector:
13      matchLabels:
14        alertmanager: main
15
```

Tag 3

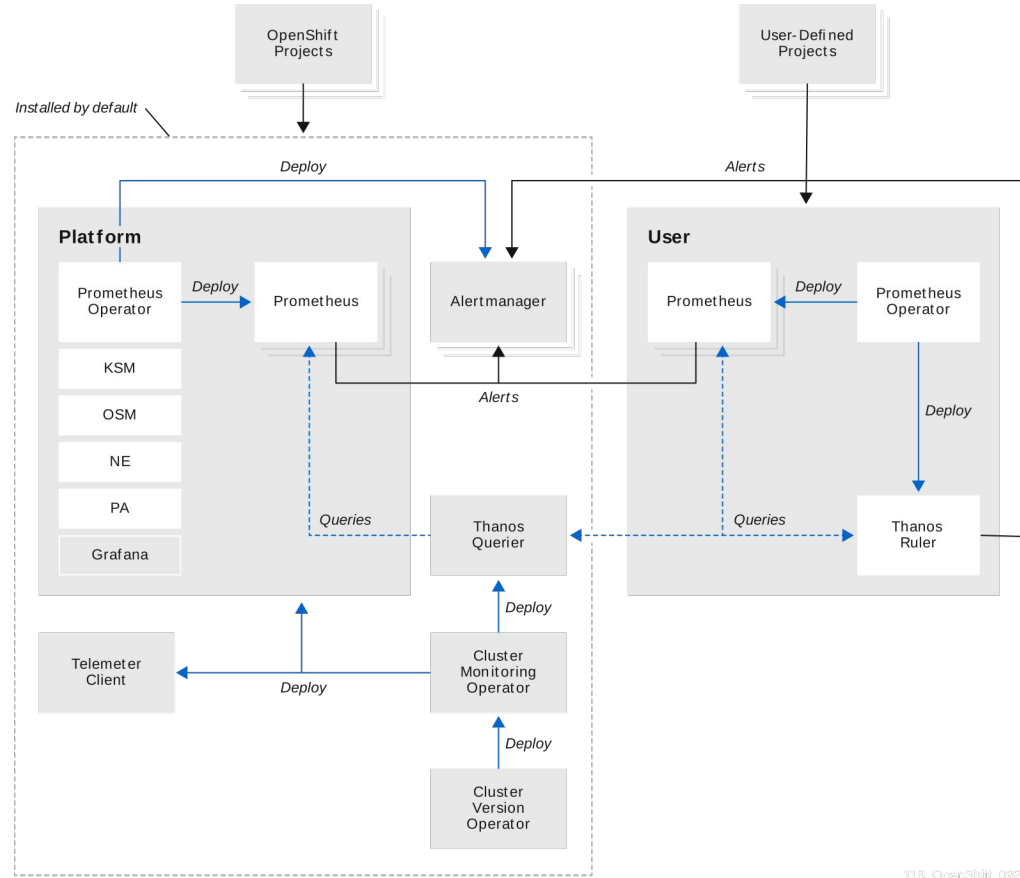
@codecentric

Alerting
Persistent Storage
Networking
Cluster Security
Backup & Restore
Best Practices



@codecentric

Alerting



Alertmanager Channels

- E-Mail
- Webhook
- PagerDuty
- Slack
- VictorOps
- Pushover
- Wechat

```
1  global:
2    resolve_timeout: 5m
3    slack_api_url: https://hooks.slack.com/services/.....
4  receivers:
5    - name: default
6    - name: slack
7      slack_configs:
8        - channel: '#operations'
9          send_resolved: true
10 route:
11   group_by:
12     - alertname
13     - namespace
14   group_interval: 5m
15   group_wait: 1m
16   receiver: default
17   repeat_interval: 48h
18   routes:
19     - receiver: default
20       match:
21         alertname: Watchdog
22     - receiver: slack
23       repeat_interval: 4h
24       match:
25         severity: critical
26     - receiver: slack
27       group_wait: 5m
28       group_interval: 15m
29       match:
30         severity: warning
31     - receiver: slack
32       repeat_interval: 24h
33       match:
34         severity: high
```

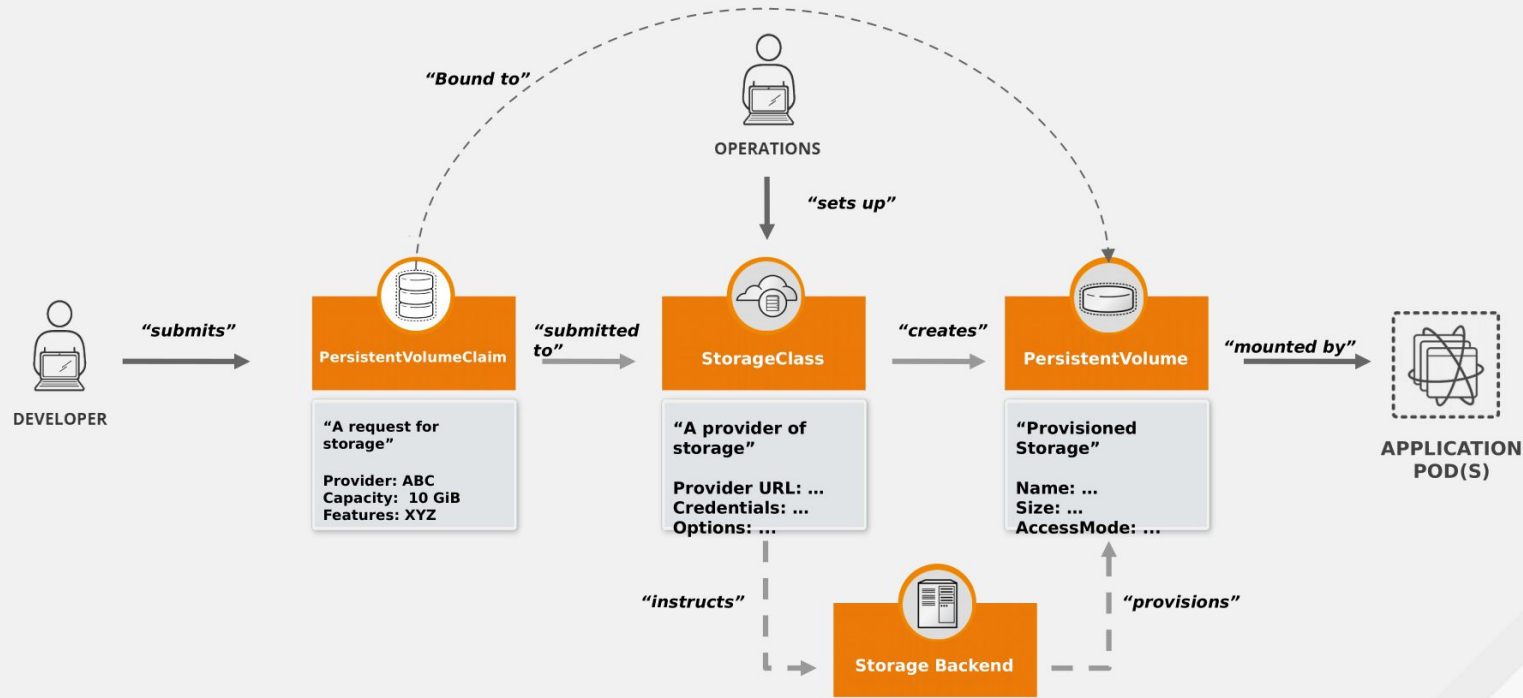
Eigene Alerts

```
1  apiVersion: monitoring.coreos.com/v1
2  kind: PrometheusRule
3  metadata:
4    name: cluster-update-available
5    namespace: openshift-cluster-version
6  spec:
7    groups:
8      - name: cluster-updates
9        rules:
10         - alert: ClusterUpdateAvailable
11           annotations:
12             message: >-
13               A new cluster update is available on your selected channel
14           expr: |
15             cluster_version_available_updates > 0
16           for: 5m
17           labels:
18             severity: info
```



Persistent Storage

OPENSIFT PERSISTENT STORAGE FRAMEWORK



Persistent Storage Provider

- HostPath
- EmptyDir (Ephemeral Storage)
- LocalStorage

- NFS
- iSCSI
- Ceph (OpenShift Container Storage)
- Diverse Cloud / Hypervisor Features

Access Modes

- Read Only (ROX)
- Read Write Once (RWO)
- Read Write Many (RWX)

Storage Classes

- verschiedene Arten von Storage
- verschiedene Parameter am Storage Provider

OpenShift Container Storage

- Bietet Block Storage, File Storage und Object Storage
- basiert auf Ceph
- benötigt extra subscription

	Recommended	Minimum (TP)
CPU	30	24
Memory	72 GB	72 GB
Storage Devices	3 x 2 TB	3 x 0,5 TB
Usable Storage	2 TB	0,5 TB

@codecentric

Networking

Network Plugins

- OpenShift SDN
- OVN-Kubernetes
- vendor plugins

Feature	OpenShift SDN	OVN-Kubernetes
Egress IPs	Supported	Supported
Egress firewall ^[1]	Supported	Supported
Egress router	Supported	Not supported
Kubernetes network policy	Partially supported ^[2]	Supported
Multicast	Supported	Supported

1. Egress firewall is also known as egress network policy in OpenShift SDN. This is not the same as network policy egress.

2. Does not support egress rules and some `ipBlock` rules.

OpenShift SDN

- default network plugin
- auf Basis von Open vSwitch (OVS)
- network isolation modes
 - Subnet
 - NetworkPolicy
 - Multitenant

OVN-Kubernetes

- fully supported
- auf Basis von Open Virtual Network (OVN)
- IPsec zwischen den Nodes (ab 4.7)
- wird in Zukunft default
- unterstützt NetworkPolicy vollständig

Network Policy

```
1  apiVersion: networking.k8s.io/v1
2  kind: NetworkPolicy
3  metadata:
4    name: api-allow-http-and-https
5    namespace: default
6  spec:
7    podSelector:
8      matchLabels:
9        app: api
10   ingress:
11     - from:
12         - podSelector:
13             matchLabels:
14               role: monitoring
15         ports:
16           - protocol: TCP
17             port: 80
18           - protocol: TCP
19             port: 443
20
```

```
1  apiVersion: networking.k8s.io/v1
2  kind: NetworkPolicy
3  metadata:
4    name: web-allow-production
5    namespace: default
6  spec:
7    podSelector:
8      matchLabels:
9        app: web
10   ingress:
11     - from:
12         - namespaceSelector:
13             matchLabels:
14               env: production
15
```

@codecentric

Cluster Security

Security Context Constraints (SCC)

- Kontrolliert die Rechte die ein Pod anfordern kann
- Ohne SCC werden erweiterte Rechte vom Scheduler zurückgewiesen
- Erlaubt Pods:
 - Zugriff auf Host Dateisystem
 - Zugriff auf Host Netzwerk
 - Starten als spezifischer User, bzw Root
 - Setzen von SELinux context
 - Erweiterte Möglichkeiten mit Linux-Gruppen
 - Erlauben bestimmter Linux Capabilities

Was man **NIEMALS** tun sollte ...

- Rechte an den default Service Account geben
- SCC an den default Service Account geben
- “privileged” SCC vergeben
- *Container als root laufen lassen weil man zu faul ist es richtig zu machen*

oc adm policy add-scc-to-user privileged -z default

etcd Encryption

- Transparente Verschlüsselung
- verschlüsselt Secrets, ConfigMaps, Routes, AccessToken
- verschlüsselt nur Values und keine Keys

```
1  apiVersion: config.openshift.io/v1
2  kind: APIServer
3  metadata:
4    name: cluster
5  spec:
6    audit:
7      profile: Default
8    encryption:
9      type: aescbc
```

Image Config

- allowed registries
- insecure registry
- registries for import

default registry: docker.io

```
1  apiVersion: config.openshift.io/v1
2  kind: Image
3  metadata:
4    name: cluster
5  spec:
6    allowedRegistriesForImport:
7      - domainName: docker.elastic.co
8      - domainName: quay.io
9      - domainName: registry.redhat.io
10     - domainName: registry.access.redhat.com
11    registrySources:
12      insecureRegistries:
13        - my-private-registry.com
14      allowedRegistries:
15        - docker.elastic.co
16        - quay.io
17        - registry.access.redhat.com
18        - registry.redhat.io
19        - 'image-registry.openshift-image-registry.svc:5000'
20
```



Backup & Restore

Backup etcd

- Der ganze Cluster State ist im etcd
- etcd Backup Script auf Master Nodes
- Man braucht nur ein etcd node Backup

Backup

Recovery

etcd Backup niemals in neuen Cluster zurückspielen

Alternative Backup Options

- Snapshots der Master Nodes
- Snapshot aller Nodes
- Velero
- Automatisierung + CI/CD Pipeline

@codecentric

Best Practices

Nennenswerte Fakten

- Traffic zwischen Cluster Komponenten und der API ist verschlüsselt
- Traffic zwischen Application Pods ist nicht verschlüsselt
- Der Cluster läuft in UTC
- Container laufen by default in UTC
- Container laufen normalerweise mit User ID > 1M

Cluster Konfiguration

- LDAP über Keycloak anschließen
- self-provisioner Rechte wegnehmen
- blockieren von Docker Hub
- Infrastructure Nodes
- nicht mit cluster-admin Rechten übertreiben

Regelmäßige Aufgaben

- Cluster Update
- etcd defrag
 - muss regelmäßig auf jedem etcd node gemacht werden
 - einer nach dem anderen mit Zeit dazwischen
 - der Leader als letztes

Common Cluster Tasks

Tipps & Tricks

- oc debug node
- etcd storage performance
- ImageStream Reference Policy
- Copy images to registry with port-forwarding
- learn.openshift.com

Ende

@codecentric

Channels to watch

- Deutscher OpenShift Slack Channel: openshift-de.slack.com
- OpenShift Anwender: openshift-anwender.de
- OpenShift Blog: blog.openshift.com
- OpenShift TV: openshift.tv


Events

- Nächstes OpenShift Anwender Treffen (TBD)
- KubeCon Europe (Virtual Event, 04.05. - 07.05.)
- KubeCon North America (Los Angeles, 12.10. - 15.10.)



 **codecentric AG**
Am Mittelhafen 14
48155 Münster

 **Tobias Derksen**
DevOps Consultant
tobias.derksen@codecentric.de
www.codecentric.de

 **Telefon: +49 (0) 170 2295 733**

github.com/lukeelten/openshift-administration

