Individuelle Zabbix Installationen in On-Premises Kubernetes Umgebungen

IntelliTrend GmbH

Kontakt: Wolfgang Alper

www.intellitrend.de

wolfgang.alper@intellitrend.de

ZABBIX PREMIUM PARTNER

Beginnen wir ”einfach”

-

Zabbix Installation mit Docker

Zabbix Installation mit Docker

Docker Kommandozeile

```
docker run --name my-zabbix-server-mysql \
-e DB_SERVER_HOST="my-mysql-server" \
-e MYSQL_USER="mydbuser" \
-e MYSQL_PASSWORD="mydbpass" \
-d zabbix/zabbix-server-mysql:ubuntu-6.4-latest
```

Weitere optionale Umgebungsvariablen

```
...
ZBX_LISTENPORT=10051
ZBX_STARTREPORTWRITERS=1
ZBX_WEBSERVICEURL=http://zabbix-web-service:10053/report
...
```

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Zabbix Installation in Kubernetes

Installationsoptionen in Kubernetes

- Installation von Zabbix (ähnlich wie mit Docker) manuell mit YAML basierenden Konfiguration Dateien unter Verwendung von „kubectl“

  Oder

- Installation von Zabbix unter Verwendung von Templates, Konfigurations-Repositories und synchronizierten Konfigurationen mit “Helm”

Was ist Helm

- Helm wird häufig “The package manager for Kubernetes” genannt
- Helm ist ein K8s Konfigurations-Repository und Template System
- Helm bündelt verschiedene Konfigurationsdateien in einem Release
- Helm erlaubt konsistente Upgrades mit Versionierung
- Helm erlaubt flexible Konfigurationen über die „values.yaml“ Datei

- Offizielles Zabbix K8s **Monitoring** Helm Chart:
  https://git.zabbix.com/projects/ZT/repos/kubernetes-helm/browse

- Community Zabbix K8s **Deployment** Helm Chart:
  https://github.com/zabbix-community/helm-zabbix
  (Ehemals repo: https://github.com/cetic/helm-zabbix)

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Helm – Templates and Values

Helm Values

```
# Default values for zabbix.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

# **Zabbix Server** configurations
zabbixserver:
  # -- Enables use of **Zabbix Server**
  enabled: true
  # -- Number of replicas of ``zabbixserver`` module
  replicaCount: 1
  # -- optional set true open a port direct on node where zabbix server runs
  hostPort: false
  # -- optional set hostIP different from 0.0.0.0 to open port only on this IP
  hostIP: 0.0.0.0
  resources: {}
  image:
    # -- Zabbix server Docker image name
    repository: zabbix/zabbix-server-pgsql
    # -- Tag of Docker image of Zabbix server
    tag: ubuntu-6.0.0
    # -- Pull policy of Docker image
    pullPolicy: IfNotPresent
    # -- List of dockerconfig secrets names to use when pulling images
    pullSecrets: []
  # -- Address of database host
  DB_SERVER_HOST: "172.20.22.100"
  # -- Port of database host
  DB_SERVER_PORT: "5432"
  # -- User of database
  POSTGRES_USER: "zabbix"
  # -- Password of database
  POSTGRES_PASSWORD: "zabbix"
```

Helm Template

```
{{- if .Values.zabbixserver.enabled }}
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: {{ template "zabbix.fullname" . }}-zabbix-server
  labels:
    app: {{ template "zabbix.fullname" . }}-zabbix-server
    app.kubernetes.io/name: zabbix-server
    helm.sh/chart: {{ include "zabbix.chart" . }}
    app.kubernetes.io/instance: {{ .Release.Name }}-zabbix-se
    app.kubernetes.io/managed-by: {{ .Release.Service }}-zabb
spec:
  replicas: {{ .Values.zabbixserver.replicaCount }}
  serviceName: {{ template "zabbix.fullname" . }}
  selector:
    matchLabels:
      app: {{ template "zabbix.fullname" . }}-zabbix-server
  template:
    metadata:
      labels:
        app: {{ template "zabbix.fullname" . }}-zabbix-server
        app.kubernetes.io/name: zabbix-server
        helm.sh/chart: {{ include "zabbix.chart" . }}
        app.kubernetes.io/instance: {{ .Release.Name }}-zabbi
        app.kubernetes.io/managed-by: {{ .Release.Service }}-
```

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Installation von Zabbix in K8s unter Verwendung von Helm

Installation von Zabbix mit Helm

Installation des Zabbix Helm Repository

```
[serveradmin@k8s-master demo]$ helm repo add cetic https://cetic.github.io/helm-charts
"cetic" has been added to your repositories

[serveradmin@k8s-master demo]$ helm repo update
 Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "metallb" chart repository
...Successfully got an update from the "nfs-subdir-external-provisioner" chart repository
...Successfully got an update from the "cetic" chart repository
...Successfully got an update from the "prometheus-community" chart repository
Update Complete. *Happy Helming!*
```

Erstellung der “values_zabbix.yaml” (skeleton mit Defaults)

```
[serveradmin@k8s-master demo]$ helm show values cetic/zabbix > zabbix_values.yaml
```

Installation von Zabbix mit Helm

Edit zabbix_values.yaml:
- Disable Postgres deployment since we use an external DB
- Set DB_SERVER_HOST to address of external DB
- Set username, password and database name as required

```
# **PostgreSQL** configurations
postgresql:
# -- Create a database using Postgresql
  enabled: false
  auth:
  # -- Enable remote access to "postgres" user
  enablePostgresUser: true
  # -- Password of "postgres" user
  postgresPassword: "zabbix_pwd"
  # -- User of database
  username: "zabbix"
  # Password of database
  password:   "zabbix"
  # -- Name of database
  database: "zabbix"
```

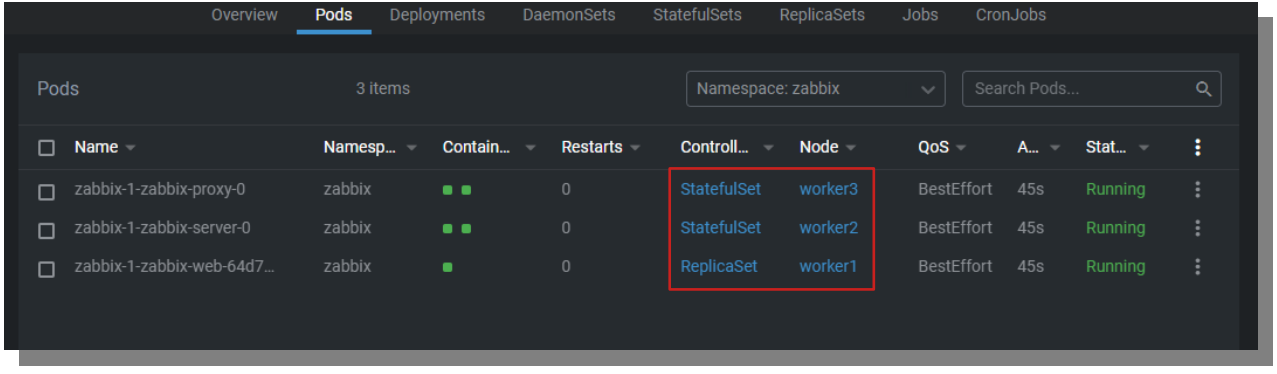
```
# **Zabbix Server** configurations
zabbixserver:
# -- Enables use of **Zabbix Server**
  enabled: true
  # -- Number of replicas of "zabbixserver"
  replicaCount: 1
 # -- optional set true open a port direct
  hostPort: false
  # -- optional set hostIP
  hostIP: 0.0.0.0
  resources: {}
  image:
  # Zabbix server Docker image name
  repository: zabbix/zabbix-server-pgsql
  # -- Tag of Docker image of Zabbix server
  tag: ubuntu-6.0.0
  # -- Pull policy of Docker image
  pullPolicy: IfNotPresent
  # -- List of dockerconfig secrets
  pullSecrets: []
  # -- database
  DB_SERVER_HOST: "172.20.20.100”
  DB_SERVER_PORT: "5432”
  POSTGRES_USER: "zabbix"
  POSTGRES_PASSWORD: "zabbix"
```

Installation von Zabbix mit Helm

Erstellung eines Helm Releases „zabbix-1“ im Namespace „zabbix“:

```
[serveradmin@k8s-master demo]$ helm install -n zabbix --create-namespace -f zabbix_values.yaml zabbix-1 cetic/zabbix
NAME: zabbix-1
LAST DEPLOYED: Thu Apr 14 10:07:32 2022
NAMESPACE: zabbix
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace zabbix -l "app=zabbix-1-zabbix-web" -o jsonpath="{.items[0].metadata.name}")
  export CONTAINER_PORT=$(kubectl get pod --namespace zabbix $POD_NAME -o jsonpath="{.spec.containers[0].ports[0].containerPort}”)
  echo "Visit http://127.0.0.1:8888 to use your application"
  kubectl --namespace zabbix port-forward $POD_NAME 8888:$CONTAINER_PORT
```

Pods vom Release „zabbix-1“ nach Installation

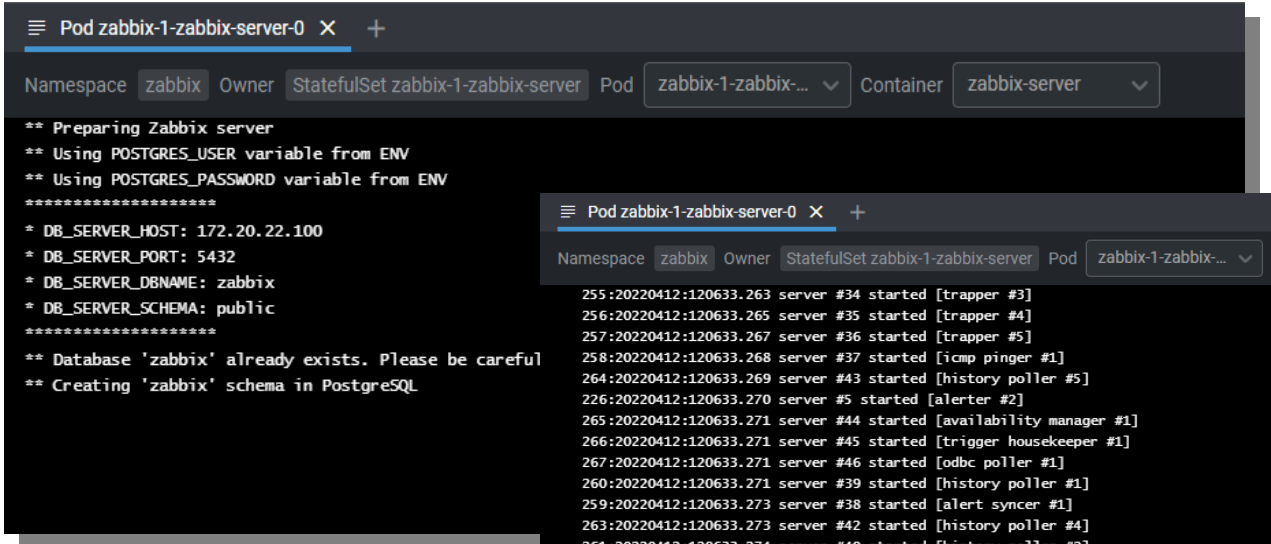


ZABBIX PREMIUM PARTNER

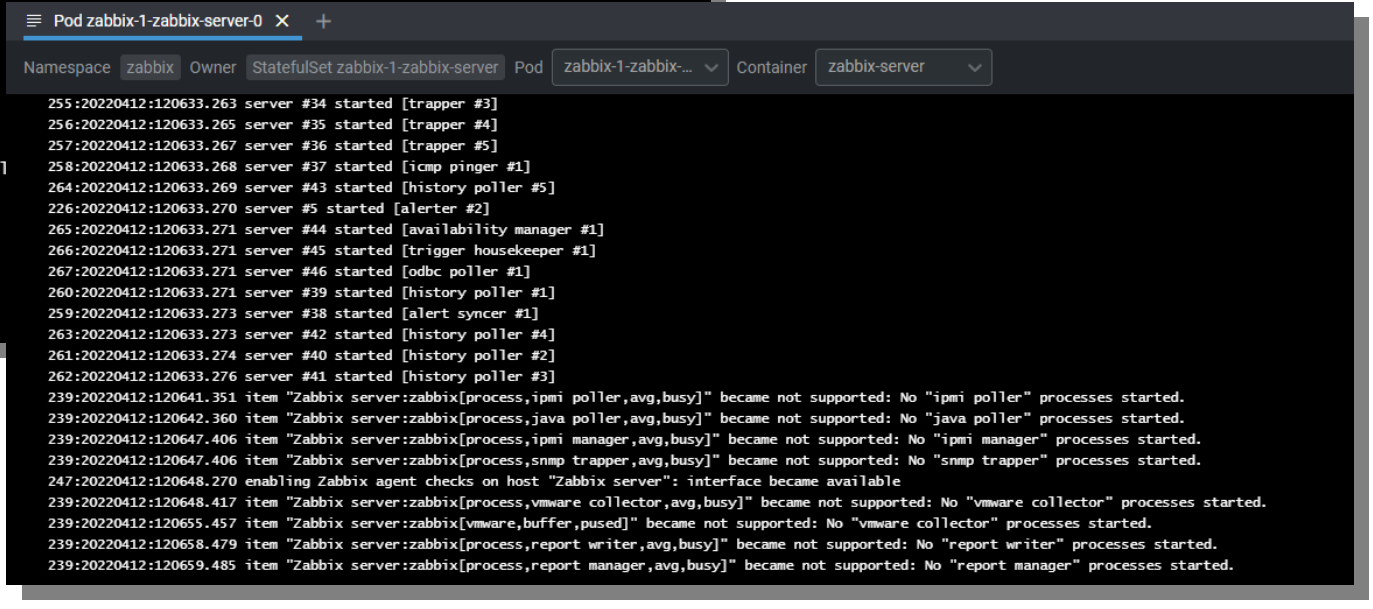
IntelliTrend IT-Services

Start der neuen Zabbix Instanz

Zabbix Server in „zabbix-1“, Initialisierung der DB:



Zabbix Server in „zabbix-1“, Start:

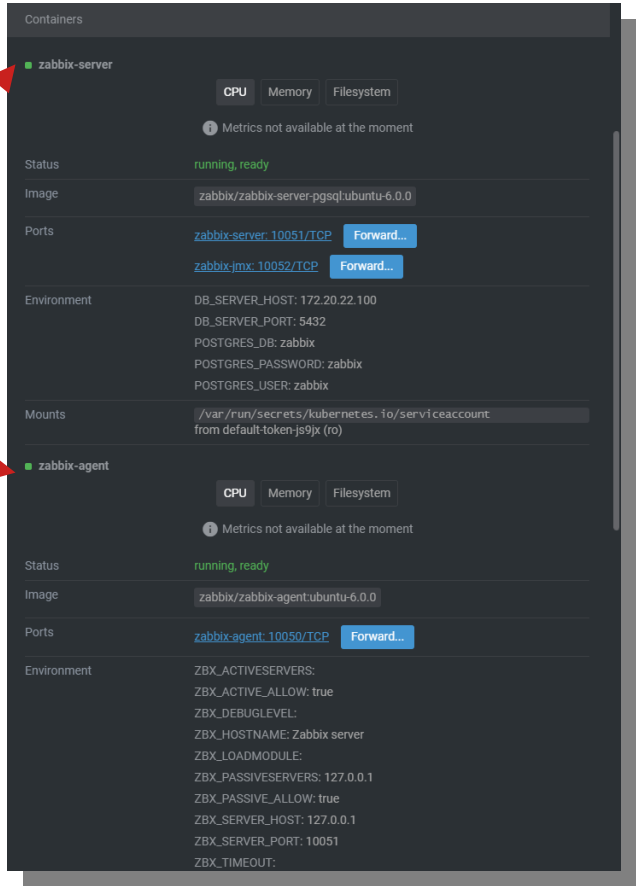


ZABBIX PREMIUM PARTNER

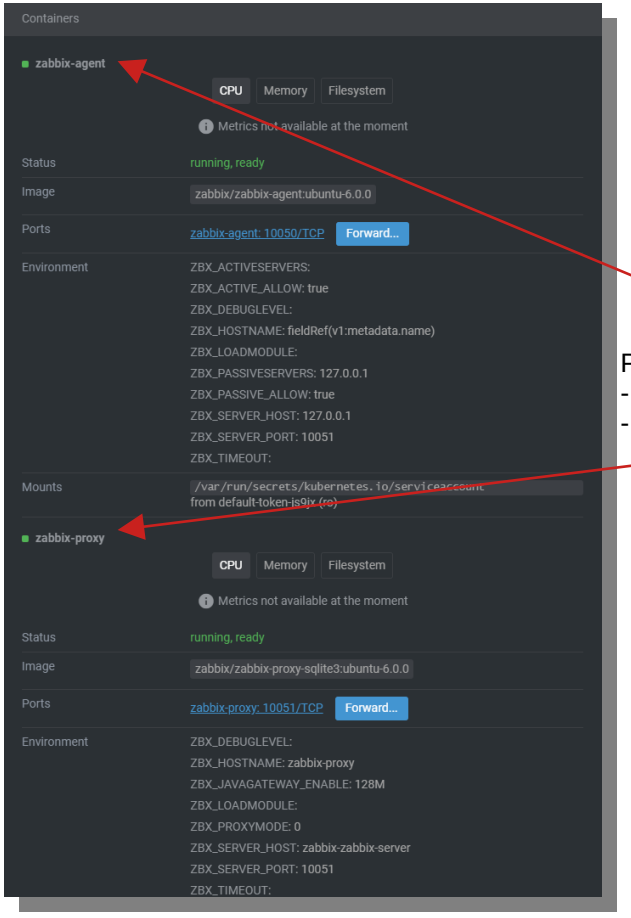
IntelliTrend IT-Services

Zabbix Instanz – Pods

Pod „zabbix-1-zabbix-server-0“
- Container zabbix-server
- Container zabbix-agent



Pod „zabbix-1-zabbix-proxy-0“
- Container zabbix-agent
- Container zabbix-proxy



ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Services
Erreichbarkeit außerhalb des Clusters

Services – Erreichbarkeit außerhalb des Clusters

- HTTP-basierte Dienste können über einen Reverse-Proxy wie nginx oder Traefik geroutet werden.
- Nicht-HTTP-basierte TCP-/UDP-Dienste können nicht einfach geroutet werden, sie benötigen eine eigene externe IP-Adresse

Cloud-Dienste wie AWS, Azure oder GCS bieten Load Balancer als Lösung für dieses Problem

Wie lässt sich dieses Problem in einer selbst gehosteten Cloud-Umgebung lösen?

Services – Erreichbarkeit außerhalb des Clusters

Lösung: MetalLB

- MetalLB ist ein Layer-2-Netzwerk-Load-Balancer, der in selbst gehosteten Cloud-Umgebungen eingesetzt werden kann
- MetalLB weist den Cluster-Diensten externe IP-Adressen zu und verlässt sich nicht auf URLs oder Reverse Proxies

```
[serveradmin@k8s-master demo]$ helm repo add metallb https://metallb.github.io/metallb
"metallb" has been added to your repositories

[serveradmin@k8s-master demo]$ helm repo update

[serveradmin@k8s-master demo]$ helm install metallb metallb/metallb --namespace metallb \
 --create-namespace -f metallb_values.yaml --version "0.11.0"
NAME: metallb
LAST DEPLOYED: Thu Apr 14 12:08:35 2022
NAMESPACE: metallb
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES: MetalLB is now running in the cluster.
LoadBalancer Services in your cluster are now available on the IPs you
defined in MetalLB's configuration:

config:
  address-pools:
  - addresses:
    - 172.20.22.110-172.20.22.120
    name: default
    protocol: layer2
```

Add repo

Update repo

Install

Address pool

https://github.com/metallb/metallb

Reconfiguration des Deployments mit MetalLB

Editiere zabbix_values.yaml, verwende service/type “LoadBalancer” anstatt “ClusterIP”

```
# -- Password of database
POSTGRES_PASSWORD: "zabbix"
# -- Name of database
POSTGRES_DB: "zabbix"
service:
  # -- Type of service in Kubernetes cluster
  type: LoadBalancer
  # -- Cluster IP for Zabbix server
  clusterIP:
  # -- Port of service in Kubernetes cluster
  port: 10051
  # NodePort of service on each node
  nodePort: 31051
  # -- Annotations for the zabbix-server service
  annotations: {}
  # metallb.universe.tf/address-pool: production-public-ips
# -- Extra environment variables. A list of additional environment variables.
extraEnv: {}
```

Zabbix Services via MetalLB

Aktualisiere das Release „zabbix-1“ nach der Änderung in zabbix_values.yaml

```
[serveradmin@k8s-master demo]$ helm upgrade -n zabbix -f zabbix_values.yaml zabbix-1 cetic/zabbix
Release "zabbix-1" has been upgraded. Happy Helming!
NAME: zabbix-1
LAST DEPLOYED: Tue Apr 26 15:18:37 2022
NAMESPACE: zabbix
STATUS: deployed
REVISION: 11
TEST SUITE: None
NOTES:

Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace zabbix -l "app=zabbix-1-zabbix-web" -o jsonpath="{.items[0].metadata.name}")
  export CONTAINER_PORT=$(kubectl get pod --namespace zabbix $POD_NAME -o jsonpath="{.spec.containers[0].ports[0].containerPort}”)
  echo "Visit http://127.0.0.1:8888 to use your application"
  kubectl --namespace zabbix port-forward $POD_NAME 8888:$CONTAINER_PORT
```

Zabbix Services via MetalLB

Der Zabbix Server verwendet nun den Type „LoadBalancer“ und ist über eine externe IP-Adresse erreichbar

Services — 4 items — Namespace: zabbix — Search Services...

Name	Namespace	Type	Cluster IP	Ports	External IP	Selector	Age	Status
zabbix-1-zabbix-agent	zabbix	ClusterIP	10.100.49.101	10050/TCP	-	app=zabbix-1-zabbi...	31m	Active
zabbix-1-zabbix-proxy	zabbix	ClusterIP	10.108.144.73	10051/TCP	-	app=zabbix-1-zabbi...	31m	Active
zabbix-1-zabbix-server	zabbix	LoadBalancer	10.103.236.54	10051:30539/TCP, 10...	172.20.22.110	app=zabbix-1-zabbi...	31m	Active
zabbix-1-zabbix-web	zabbix	NodePort	10.101.55.186	80:31390/TCP	-	app=zabbix-1-zabbi...	31m	Active

Zabbix Server ist außerhalb des Clusters zugänglich:

- External IP 172.20.22.110 (zugewiesen von MetalLB)
- Port 10051

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Real world challenges

Kubernetes - Real world challenges

Bei dem Wunsch, einfach Anwendungen in einem Kubernetes-Cluster bereitzustellen, können in der Praxis einige Herausforderungen auftreten:

- Nicht jeder ist mit kubectl, Helm-Charts usw. vertraut.
- Wie kann man konsistente Anwendungskonfigurationen sicherstellen?
- Wie kann man Konfigurationsänderungen verfolgen?
- Wie verwaltet man Passwörter/Geheimnisse in Konfigurationsdateien?
- Wie kann man Nutzern/Admins erlauben, eine Anwendung einfach bereitzustellen?

Wir brauchen Bausteine zur Schaffung eines benutzerfreundlicheren Ansatzes

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Baustein: Flux

(Verwaltung und Anwendung von Konfigurationen)

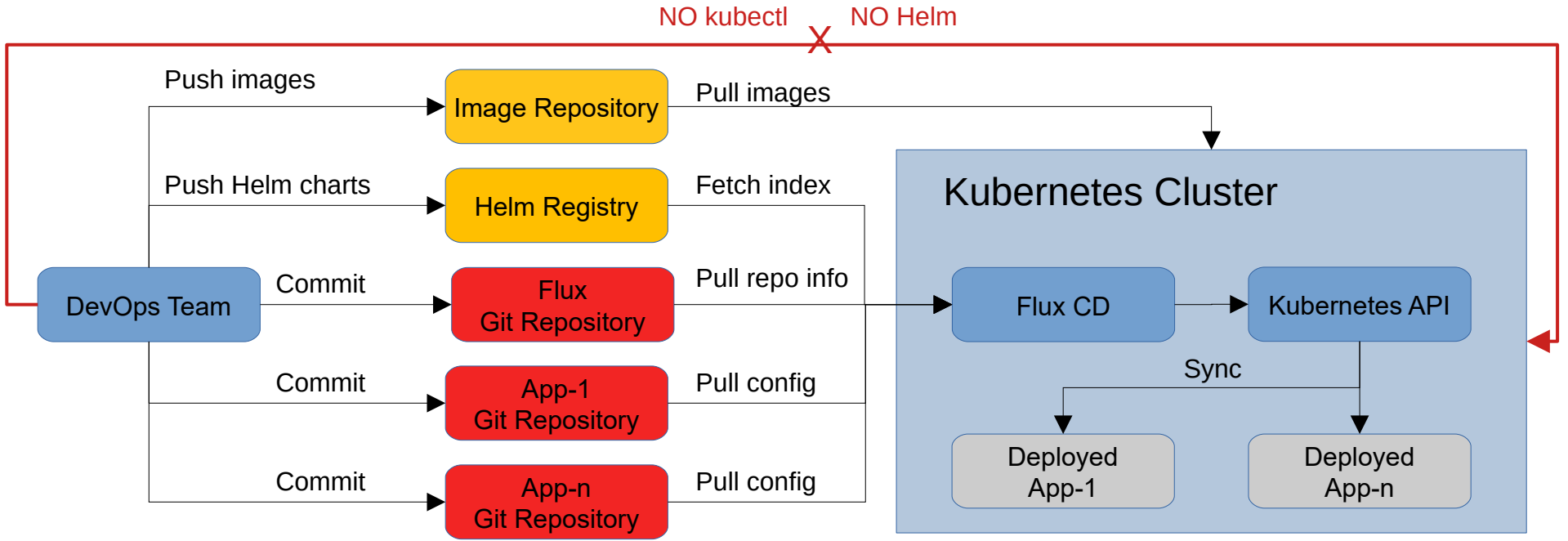
Baustein - Flux

Flux bietet eine GitOps-Lösung für Kubernetes

- Die K8-Konfigurationen werden mit Git-Repositories synchronisiert. Dies erlaubt:
  - Versionierung
  - Einfache Rollbacks
  - Konsistente Konfigurationszustände
- Flux wird im Cluster bereitgestellt und steuert den Cluster über die Kubernetes-API
- Anstelle von "kubectl apply" oder "helm install" werden Änderungen festgeschrieben und in Git-Repositories übertragen
- Flux fragt diese Git-Repositories nach Änderungen ab und führt bei Bedarf automatische Rollouts im Cluster durch

See https://github.com/fluxcd/flux2

Baustein - Flux/Git



Flux Git Repository

- Flux base configuration
- References to App repositories

App Git Repositories

- App Helm release config
- Other K8s YAML config files

In unserem Fall sind die Anwendungen einzelne Zabbix-Instanzen

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Baustein: SOPS

(Verwaltung and Verwendung von “Secrets”)

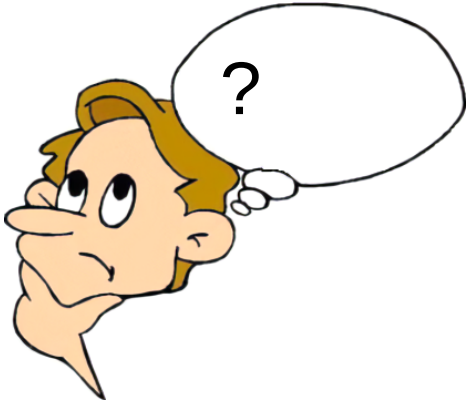
Warum SOPS

- Anwendungen erfordern Anmeldeinformationen (Datenbanken, API Tokens, E-Mail usw.), die in Konfigurationsdateien für GitOps gespeichert sind.
- Allerdings: Anmeldedaten im Klartext in Git sind keine gute Idee!
- Verschlüsselung verwenden?

  → Ja: Aber, eingesetzte Anwendungen müssen weiterhin in der Lage sein, Anmeldedaten zu lesen
- Wohin mit den Schlüsseln für die verschlüsselten Anmeldeinformationen?

  → Huhn-und-Ei-Problem
- Mehrere Benutzer müssen möglicherweise Anmeldedaten bearbeiten

  → Wie können die Schlüssel sicher geteilt werden?



Baustein - SOPS

Mozilla SOPS (Secrets OPerationS) verwaltet verschlüsselte Informationen in Textdateien

- Sensible Werte werden verschlüsselt und in spezielle Base64-Strings kodiert
- Verwendung von asymmetrischer Kryptographie mit einem oder mehreren öffentlichen/privaten Schlüsselpaaren verschiedener Benutzer
- Flux erhält den privaten Schlüssel und dekodiert verschlüsselte YAML-Dateien “on the fly”
- Nur Besitzer der privaten Schlüssel können die verschlüsselten Werte einsehen und bearbeiten
- Support für Flux, Hashicorp Vault, AWS KMS, Azure Key Vault, GCP KMS, Age, PGP etc.

```
myapp1: ENC[AES256_GCM,data:Tr7o=,iv:1=,aad:No=,tag:k=]
app2:
    db:
        user: ENC[AES256_GCM,data:CwE4O1s=,iv:2k=,aad:o=,tag:w==]
        password: ENC[AES256_GCM,data:p673w==,iv:YY=,aad:UQ=,tag:A=]
    # private key for secret operations in app2
    key: |-
        ENC[AES256_GCM,data:Ea3kL5O5U8=,iv:DM=,aad:FKA=,tag:EA==]
an_array:
- ENC[AES256_GCM,data:v8jQ=,iv:HBE=,aad:21c=,tag:gA==]
- ENC[AES256_GCM,data:X10=,iv:o8=,aad:CQ=,tag:Hw==]
- ENC[AES256_GCM,data:KN=,iv:160=,aad:fI4=,tag:tNw==]
sops:
    kms:
    -   created_at: 1441570389.775376
        enc: CiC....Pm1Hm
        arn: arn:aws:kms:us-east-1:656532927350:key/920aff2e-c5f1-4040-943a-047fa387b27e
    -   created_at: 1441570391.925734
        enc: Ci...awNx
        arn: arn:aws:kms:ap-southeast-1:656532927350:key/9006a8aa-0fa6-4c14-930e-a2dfb916de1d
    pgp:
    -   fp: 85D77543B3D624B63CEA9E6DBC17301B491B3F21
        created_at: 1441570391.930042
        enc: |
            -----BEGIN PGP MESSAGE-----
            hQIMA0t4uZHfl9qgAQ//UvGAwGePyHuf2/zayWcloGaDs0MzI+zw6CmXvMRNPUsA
                                ...=oJg5
            -----END PGP MESSAGE-----
```

See https://github.com/mozilla/sops

SOPS – Einfaches Beispiel

```
serveradmin@dev-k8s-master:~/sops$ cat credentials.yaml
username: "Admin"
password: "Very secret!"
serveradmin@dev-k8s-master:~/sops$ sops -e credentials.yaml
[PGP]    WARN[0000] Deprecation Warning: GPG key fetching from a keyserver within sops will be removed in a future version of sops.
username: ENC[AES256_GCM,data:ampSnwE=,iv:hVVYu22D4m/0HAccJ+1pY5wgpOdGY13NG4LjKGirOzk=,tag:UTTRhSgN4WBm79boaefAxQ==,type:str]
password: ENC[AES256_GCM,data:+Fd9GlguMhB8o13s,iv:Yd47/6YPKNJwWl17sso7nymAeanQ3hiERXLw4Nrx/ts=,tag:cskV9EZBGw2Fn9zNrWG8Pw==,type:st
sops:
    kms: []
    gcp_kms: []
    azure_kv: []
    hc_vault: []
    age: []
    lastmodified: "2022-04-26T14:50:21Z"
    mac: ENC[AES256_GCM,data:9zI2f6pIPH22+nSgN0rJphiTrv1OGx/RG4grDzbC+NdjQpV06Ie4jzC+XA71aJXj7wEInk4YLgq7yjhPaFoMV1yVn415cPb1lEA2dR
    pgp:
        - created_at: "2022-04-26T14:50:20Z"
          enc: |
            -----BEGIN PGP MESSAGE-----

            hQEMAyUpShfNkFB/AQf/U16Wq+X6oO0rI8b/Dpremc5McHa/6Bz43V28PIhqSlIC
            xLPkR0yqXU4BhOzp11bLsviDLz+msAny7l3T91y9thS/VOSbwj6mhzHc2CvTAL7g
            8CuXcAhGk70vTh8DH7N/L3zE55l4wLJ0+gnbn7CroxpoVPLbliite07qf3zIYBdv
            xGDBab0jUua/Y7Z/wO456IA6RMpB9vEe9WR1GE5iItt8a/BAsi9pWmG9vXN3UPUD
            Ub4pWl+c3REqmxShVzuzOsDQLCerBqk3F4PTIXozCuWKVz44kKWD4WIcgZC3dFFR
            Ot/JVkPdhHYjtOc1Q4AsIUayXVziKbwxaNcTgN/ed9JeAYPKTdtGFUtHRzro3rk9
            5jAtpDSV88EcXqLkw36oYT/IU+bzPaFyWA8Gj+R5WPyPhXyWqNeLNvAjwoPc4oVG
            aJ+hvthvLd/JNvpPaLYxvAmfTxdfoOP4Zz7Cupq9dw==
            =j9Xw
            -----END PGP MESSAGE-----
          fp: FBC7B9E2A4F9289AC0C1D4843D16CEE4A27381B4
    unencrypted_suffix: _unencrypted
    version: 3.7.1
serveradmin@dev-k8s-master:~/sops$
```

Plaintext input file

Encrypted master key to decrypt credentials

SOPS ermöglicht die Verwaltung und Speicherung von “Secrets” in Text Dateien

SOPS – Komplexes Beispiel

```
# data collection
StartPingers: 5
StartPollers: 100
# database section
DBName: ENC[AES256_GCM,data:hOPj6ocNN723iOEHWHlJzw==,iv:fYFp9eVMBlNyhasoLSfRI7nfwplhiUojY37ydLbKy5k=,tag:SXvjSKQQfnUahtTrdcHPBw==,type:str]
DBUser: ENC[AES256_GCM,data:22v5h0ouLO4=,iv:VDq/MptROK/4E7nlBnzKzCjvS3iU19yzjJgN0mwS2Ns=,tag:z6E+7gnjBsTG+H4cGFw3Gw==,type:str]
DBPassword: ENC[AES256_GCM,data:+ljE8VGD/94JNm2oEEDjIB81,iv:lV1TZaKLc1I4xRkG1PHQlEfafAh0Tkh3PIxxyTtk5do=,tag:xYUO0dYUW3cBrrzNpM7LRg==,type:str]
sops:
    kms: []
    gcp_kms: []
    azure_kv: []
    hc_vault: []
    age:
        - recipient: age1tqltjgmzlaw7eek27sdveygt6w2cer3zad6sn3eqtrjfrvrs5dtqwwq079
          enc: |
            -----BEGIN AGE ENCRYPTED FILE-----
            YWdlLWVuY3J5cHRpb24ub3JnL3YxCi0+IFgyNTUxOSBOSjJtazhYNkJJQXgvaTZO
            V3FZbTduNDFOTVFEYlNvZlVqS3ZmM1E3RldBCjl5Tm50N3BGbVAreXlwT2tOS3Vl
            dEJGWHJ1WU1ZY2UxRlRXT0gzRFNza0EKLS0tIHRpK1Q5UHpRV3FLcGpjZWpGSzZk
            Ykt0aXNkb0ozUWREZFZTWFB5NWYvMVkK0pHn13XD4Fy6xvD98Iq3HaSsnKUoEEtG
            d+iSaktouuXgB2lBie2ZW3d9Lb8xmXBAiWLvoxQbcvE1S/Gk0/Lfhg==
            -----END AGE ENCRYPTED FILE-----
        - recipient: age1r3lnjrmykm2kawe59vwttn37czvgxmnd0eypfhrpcgck5fgxrufqnxe0nn
          enc: |
            -----BEGIN AGE ENCRYPTED FILE-----
            YWdlLWVuY3J5cHRpb24ub3JnL3YxCi0+IFgyNTUxOSBIaEpRdmNHeksvcVF4b3N4
            cHZnc3lGMS8ydnJsUW1qSkw2c2dKQ0QydkJrCld2S0VsN3h4TEpUTTdwYlZzWGJK
            NXoyeVpHNXl4OWEzdFozR3hBbkptYVkKLS0tIDllK2dBOHFXdzdVdkowMWtZcjVt
            MU5CeXBhdUFGcTdlZHBobEpDcFFOWUkKulBletkn4pLYnDeyz2ZW1ekExWVtceVt
            mngvV5CbAwLIs/Qm2eS2F/ahH8E7kIJ+JtVoRcIqIXOyrX3WpYk57g==
            -----END AGE ENCRYPTED FILE-----
    lastmodified: "2023-08-30T12:59:52Z"
    mac: ENC[AES256_GCM,data:z/1iUEIQreDvKGWmqbgSRODCxDSulFkxPbcjfUIh3OpYHnbGx7OrR/zvyDqsaT+uLzRJCMItyxTKUikPfePjisJrlBWm0Dq8Ib9nB/
sPMQ3QiBIzUxAmi46QzV4TYhdNrgAlE/A0g+NDGvOpIn2cQBE/iHagH9k89vtqKOd3gzc=,iv:bHVROt9fqt4Q8mrE6EZrSpLjwiwxt7Lxg4asf6oiNyI=,tag:bgCn8Cxcs+KYAP8PBMLRZQ==,type:str]
    pgp: []
    encrypted_regex: ^DB.+$
    version: 3.7.3
```

Left unencrypted

Encrypted by SOPS

Two different recipients with their own public keys and encrypted masterkeys

Verschiedene Benutzer/Anwendungen können die Datei mit Ihrem Key verwalten

Baustein: Cloud Deploy

(Verwaltung von nativen Cloud-Diensten)

Baustein Cloud Deploy

Fragen wir ChatGPT: "Was ist Cloud Deploy?"

Baustein Cloud Deploy

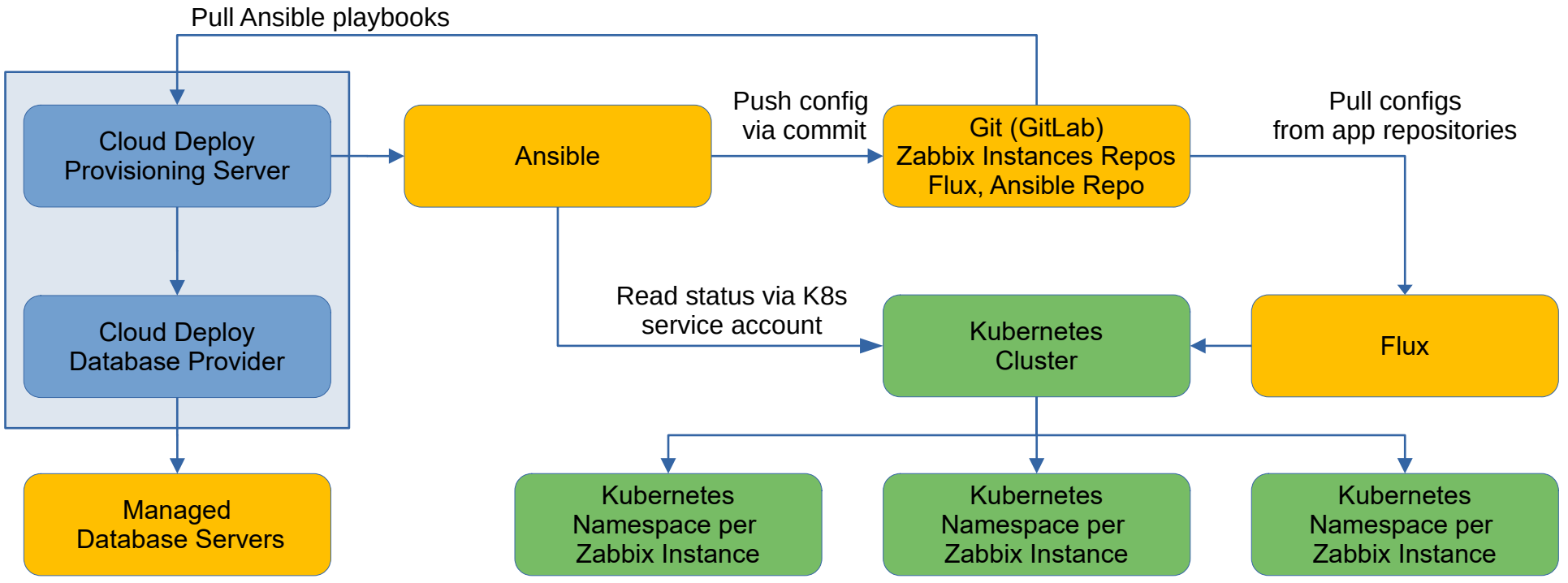
Antwort von ChatGPT:

“Cloud-Deploy" ist kein spezifischer Begriff oder eine Technologie, sondern ein weit gefasster Begriff, der sich auf den Prozess der Bereitstellung von Anwendungen oder Diensten auf einer Cloud-Infrastruktur bezieht.

Bei der Cloud-Bereitstellung wird Ihre Softwareanwendung oder Ihr System mit Hilfe von Cloud-Ressourcen und -Diensten über das Internet / lokale Netz für Benutzer oder Kunden verfügbar gemacht.

Cloud Deploy Interna

Cloud Deploy – Provisioning Design

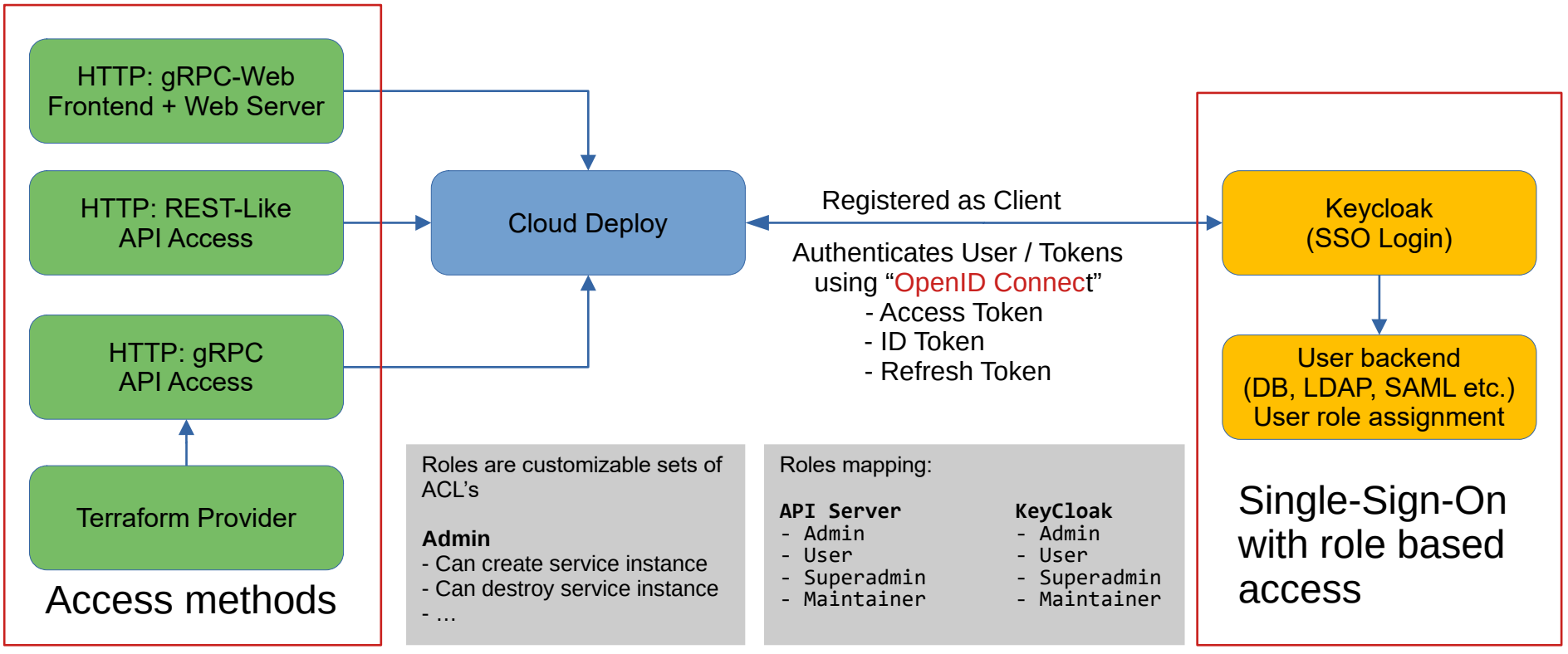


Cloud Deploy verwaltet Anwendungsbereitstellungen in Kubernetes

ZABBIX
PREMIUM PARTNER

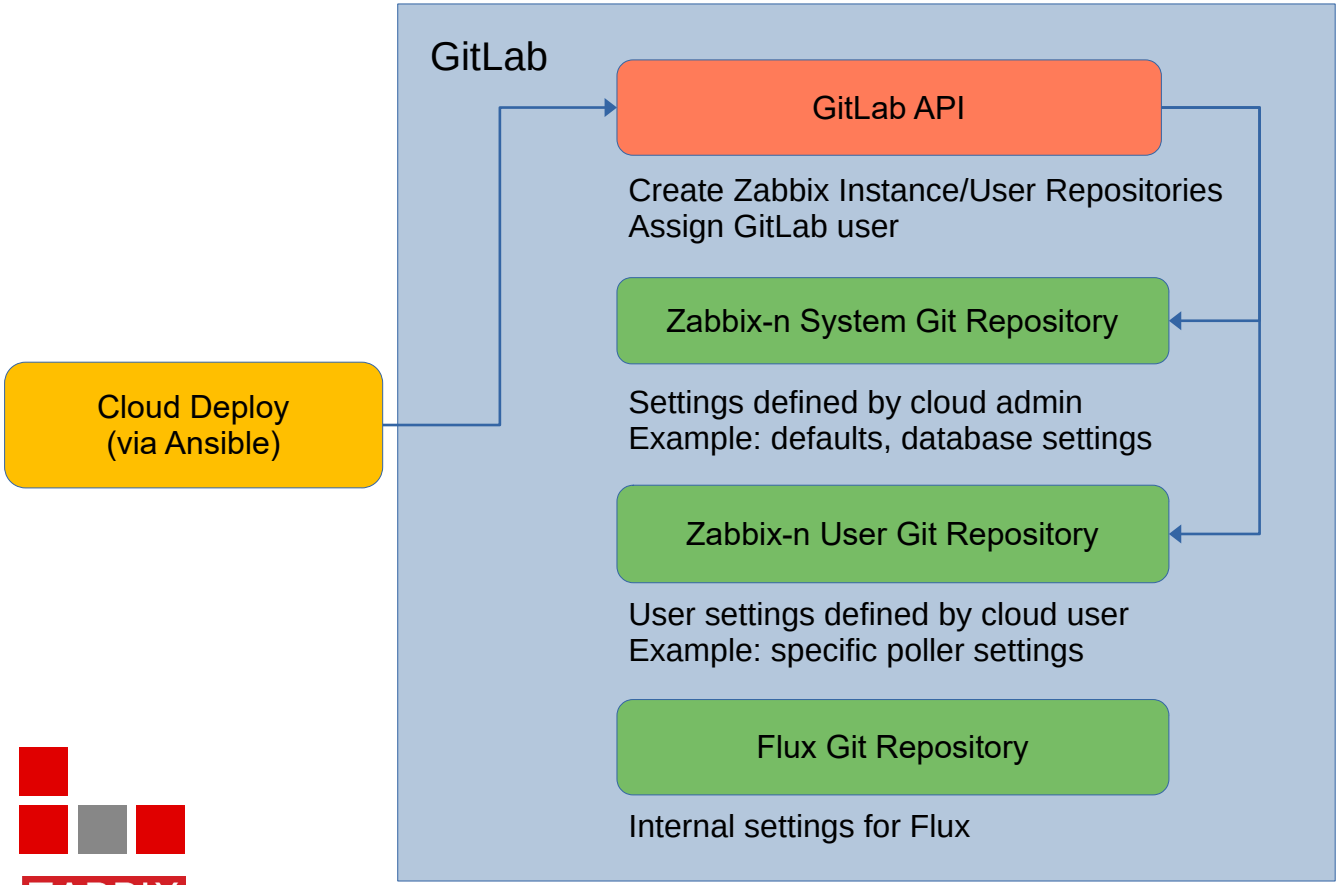
IntelliTrend
IT-Services

Cloud Deploy – Auth Design und API Zugriff



In unserem Fall sind Dienste einzelne Zabbix-Instanzen (einschließlich Frontend)

Cloud Deploy - Git Setup und Workflow - Setup



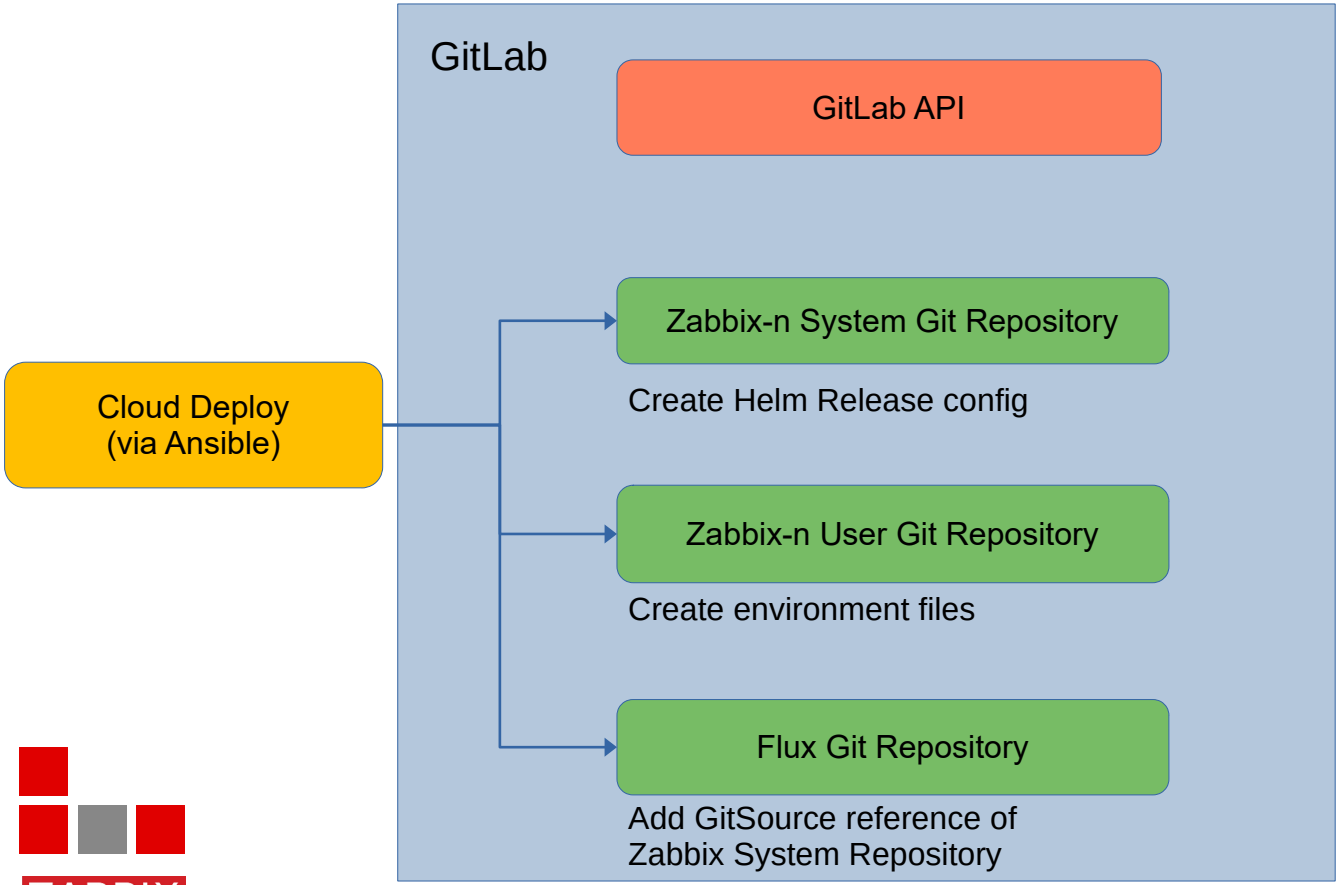
Cloud Deploy erstellt einen eigenen Satz Git-Repositories für jede Zabbix-Installation

- System Git repository
- User Git repository

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy - Git Setup und Workflow - Config



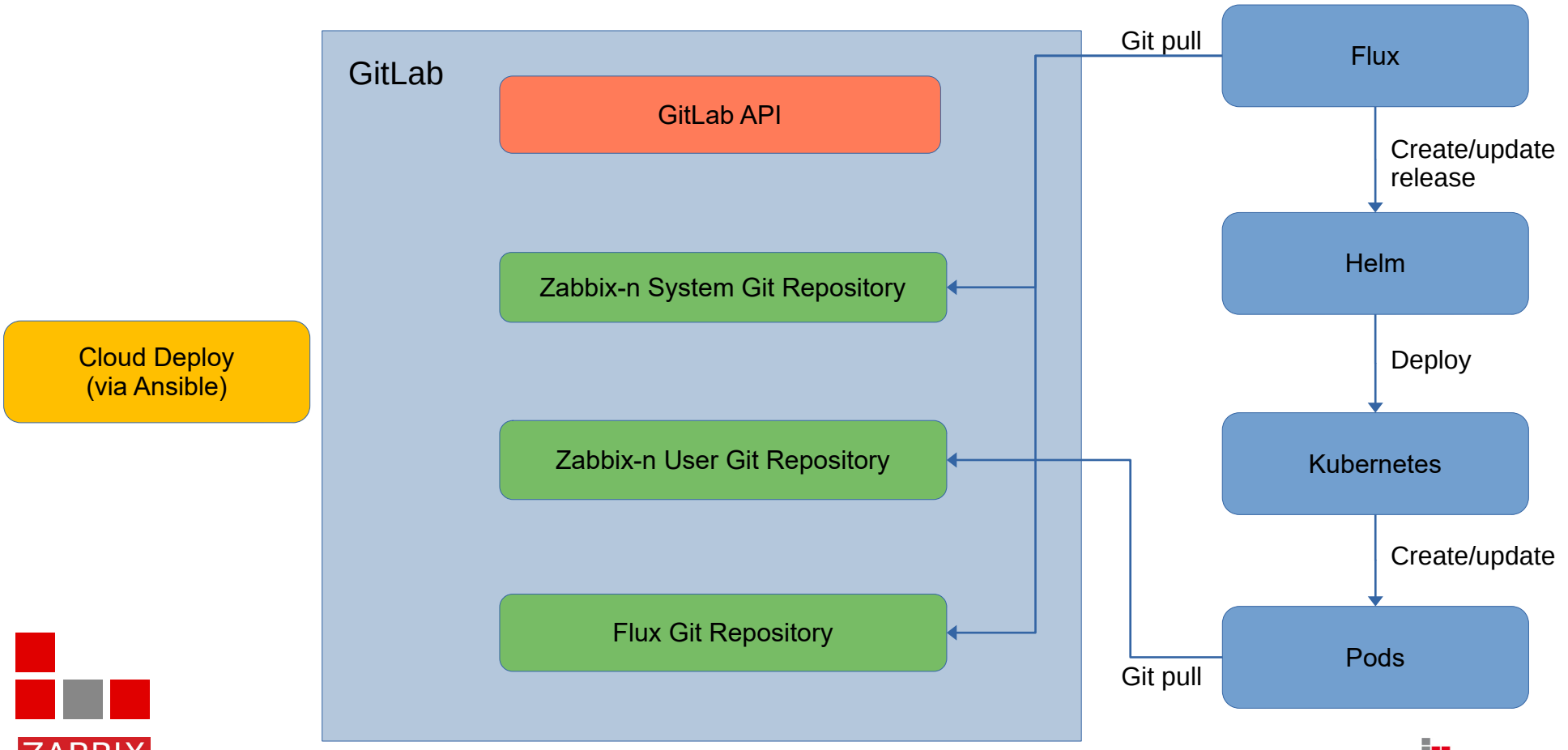
Cloud Deploy überträgt erstellte Dateien und Konfigurationen automatisch an Git

- System Git repository
- User Git repository
- Flux Git repository

ZABBIX
PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy - Git Setup und Workflow - Deploy



Cloud Deploy – Erstellte Flux Repository Dateien

Zabbix › Flux

master | flux / zabbix / my-zabbix / source.yaml | Find file | Blame | History | Permalink

Factory 'my-zabbix' install
Zabbix Provider Bot authored 1 month ago — 9ec2d3f4

source.yaml 305 bytes | Open in Web IDE | Replace | Delete

```
1  apiVersion: source.toolkit.fluxcd.io/v1
2  kind: GitRepository
3  metadata:
4    name: my-zabbix-sysrepo
5    namespace: flux-system
6  spec:
7    interval: 1m0s
8    ref:
9      branch: master
10   secretRef:
11     name: zabbix-selfprov-git-auth
12 url: ssh://git@gitlab.loc:8022/zbx/factories/my-zabbix-system.git
```

App-Repo →

Config: Register the Zabbix System Git repository in Flux

Cloud Deploy – Erstellte Flux Repository Dateien

Zabbix › Flux

master | flux / zabbix / my-zabbix / sync.yaml | Find file | Blame | History | Permalink

Factory 'my-zabbix' install
Zabbix Provider Bot authored 1 month ago — 9ec2d3f4

sync.yaml 356 bytes | Open in Web IDE | Replace | Delete

```
1  apiVersion: kustomize.toolkit.fluxcd.io/v1
2  kind: Kustomization
3  metadata:
4    name: my-zabbix-kustomization
5    namespace: flux-system
6  spec:
7    decryption:
8      provider: sops
9      secretRef:
10       name: zabbix-selfprov-sops-secret
11   interval: 1m0s
12   path: ./
13   prune: true
14   serviceAccountName: ''
15   sourceRef:
16     kind: GitRepository
17     name: my-zabbix-sysrepo
```

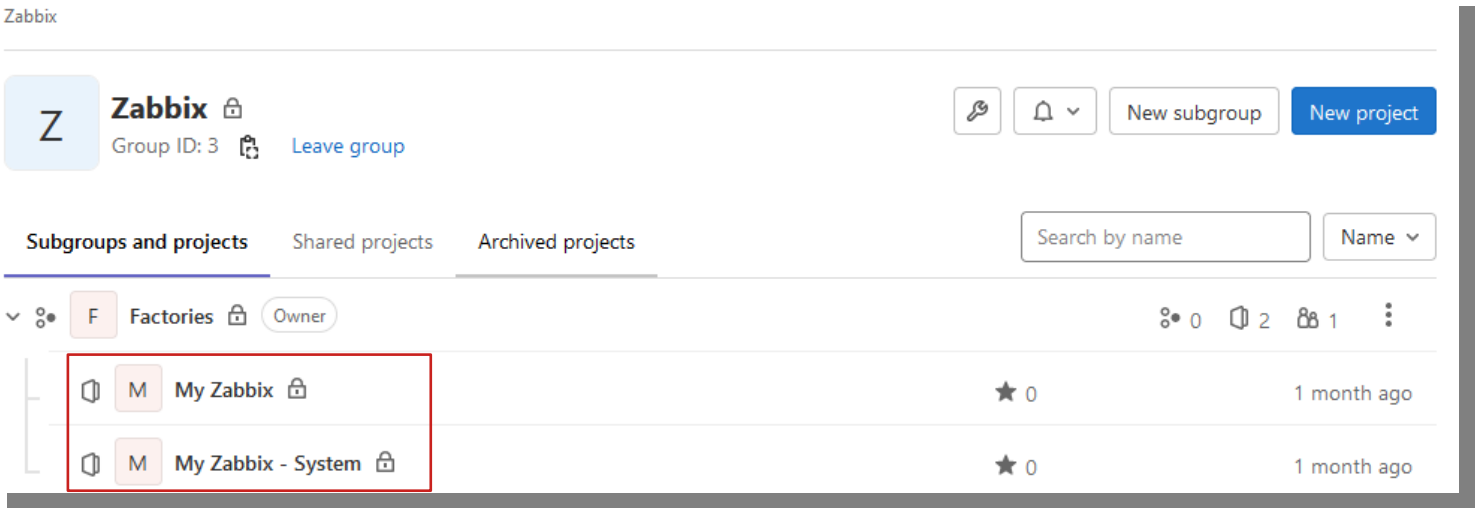
SOPS →

Config: Kustomization Object um die Flux-Konfiguration aus dem zuvor registrierten Zabbix-Repository zu lesen

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy - Zabbix Instanz Repositories



Neue Repositories, die von Cloud Deploy über Ansible für jede neue Zabbix-Instanz erstellt werden

ZABBIX PREMIUM PARTNER

Cloud Deploy - Zabbix System Repository

Zabbix › Factories › My Zabbix - System

master | my-zabbix-system / zabbix.yaml

Added load balancer IP address
Zabbix Provider Bot authored 1 month ago

zabbix.yaml 4.11 KiB

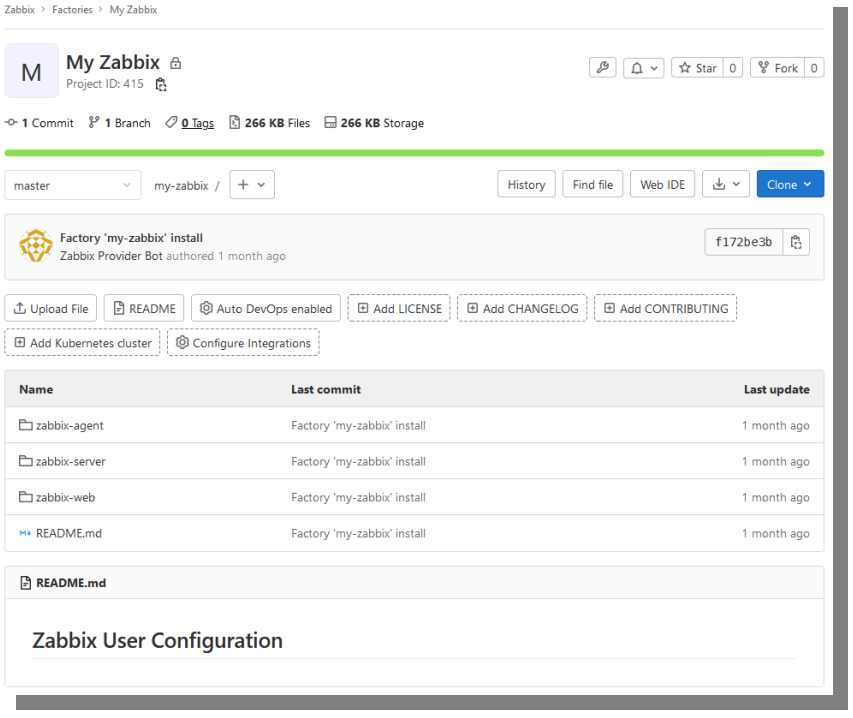
```yaml
apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
    name: my-zabbix
    namespace: flux-system
spec:
    chart:
        spec:
            chart: ./
            sourceRef:
                kind: GitRepository
                name: zabbix-helm-chart
            version: 1.0.1
    interval: 1m0s
    releaseName: my-zabbix
    serviceAccountName: ""
    targetNamespace: my-zabbix
    values:
```

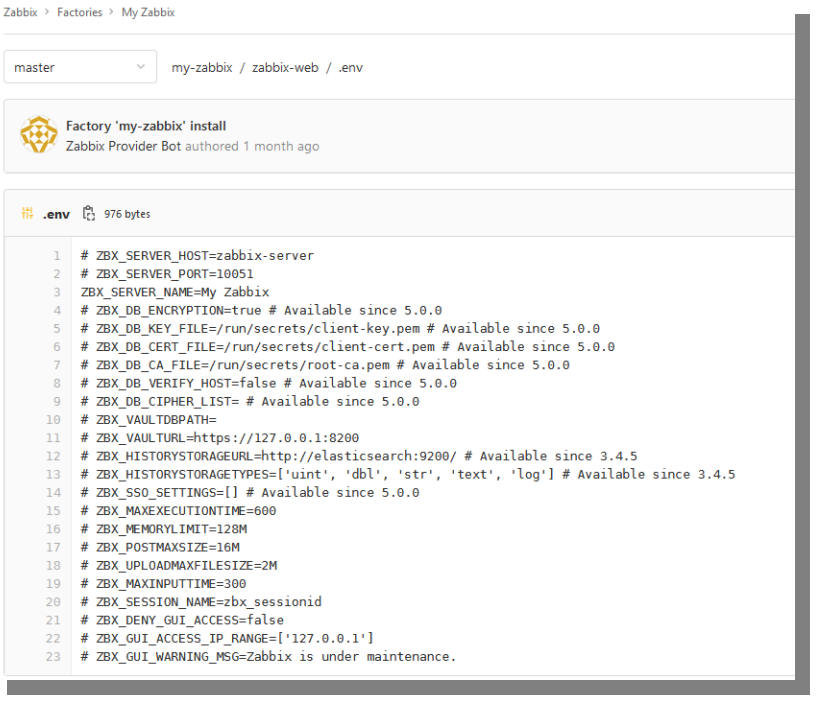
SOPS

```yaml
values:
    postgresql:
        database: zabbix
        host: 172.20.22.50
        password: ENC[AES256_GCM,data:Nom1iZ/1k9xbYw==,iv:mqG41142+uFdQZwwxPOi0m2dWRnJh19mnefMe1LE8BI=,tag:2
        port: "5431"
        username: postgres
    secretMount:
        - defaultMode: "0444"
          mountPath: /tmp/git-config
          name: git-auth-config
    zabbixAgent:
        env:
            GIT_CONFIG_REPO: ssh://git@gitlab.loc:8022/zbx/factories/my-zabbix.git
            GIT_CONFIG_UPDATE: "1689929585"
        image:
            pullPolicy: Always
            pullSecrets:
                - regcred
            registry: registry.loc
            repository: zabbix/zabbix-agent2
            tag: alpine-6.0.17
    zabbixServer:
        env:
            GIT_CONFIG_REPO: ssh://git@gitlab.loc:8022/zbx/factories/my-zabbix.git
            GIT_CONFIG_UPDATE: "1689929585"
        image:
            pullPolicy: Always
            pullSecrets:
                - regcred
            registry: registry.loc
            repository: zabbix/zabbix-server-pgsql
            tag: alpine-6.0.17
        service:
            loadBalancerIP: 172.20.22.65
            type: LoadBalancer
    zabbixWeb:
```

Von Cloud Deploy erstellte Helm Release-Konfiguration
Enthält die Werte für den Helm-Chart

Cloud Deploy - Zabbix User Repository



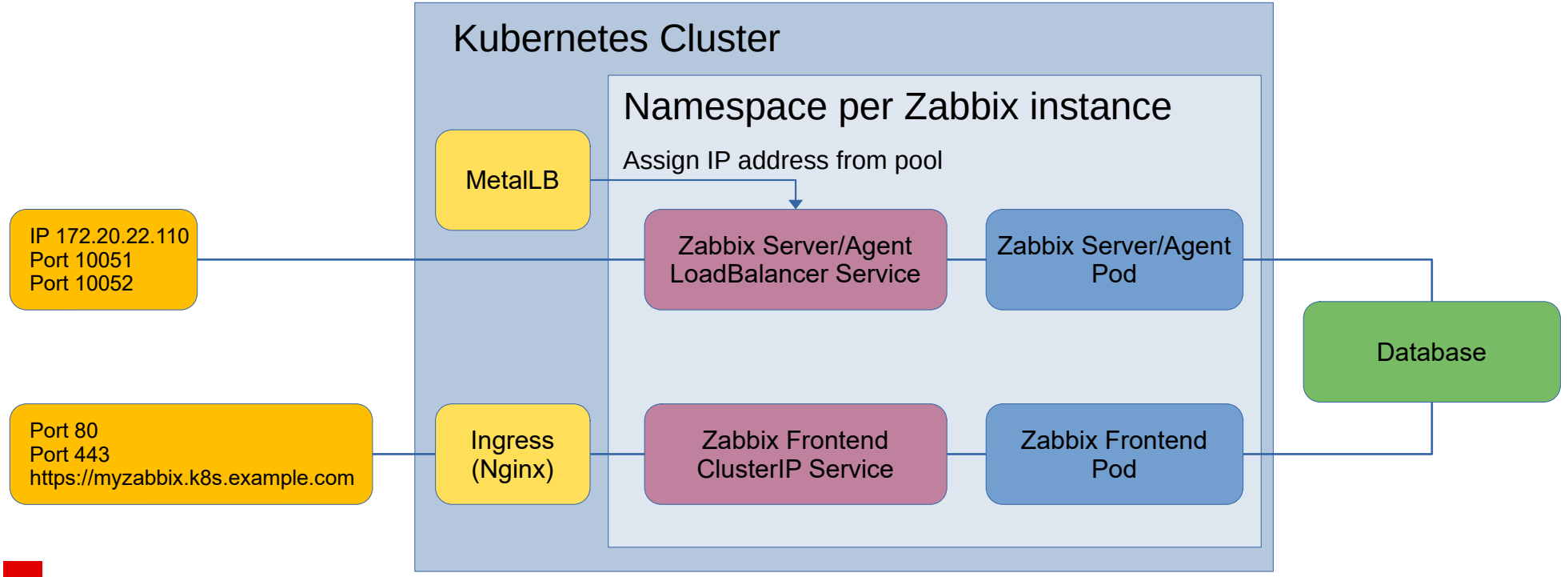


Das Benutzer-Repository enthält Umgebungsdateien mit den benutzerspezifischen Konfigurationen

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy - Cluster-Einrichtung nach Abgleich



ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy User Portal

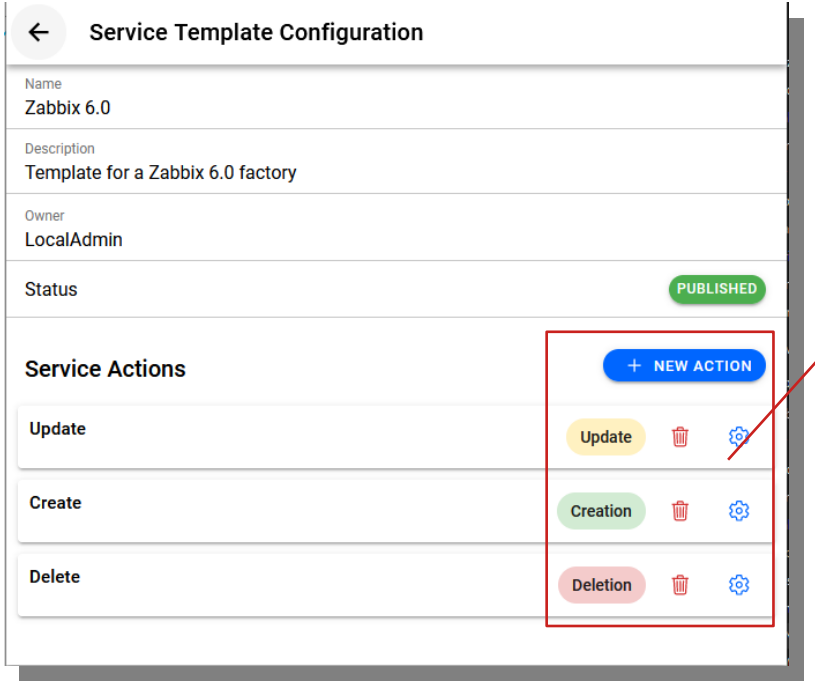
Cloud Deploy - User Portal

- Webbasiertes Portal ermöglicht die Bereitstellung neuer Dienste in Kubernetes (hier Zabbix)
- Services basieren auf Service Templates
- Service Templates definieren Provisioning Actions zum Erstellen, Aktualisieren, Löschen und Migrieren eines Services
- Jede Provisioning Action kann mehrere Tasks haben
- Unterstützt SSO and RBAC

ZABBIX
PREMIUM PARTNER

IntelliTrend
IT-Services

Cloud Deploy – Service Templates

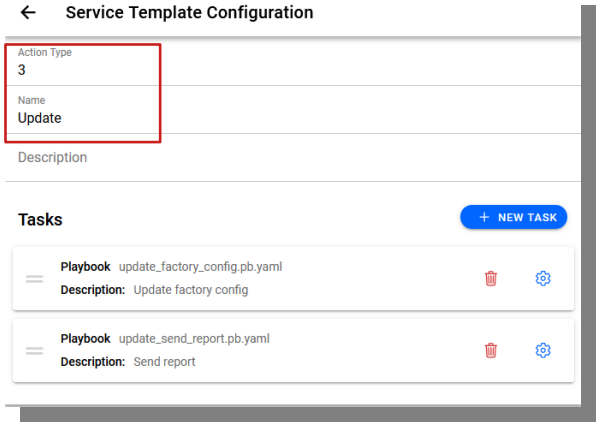


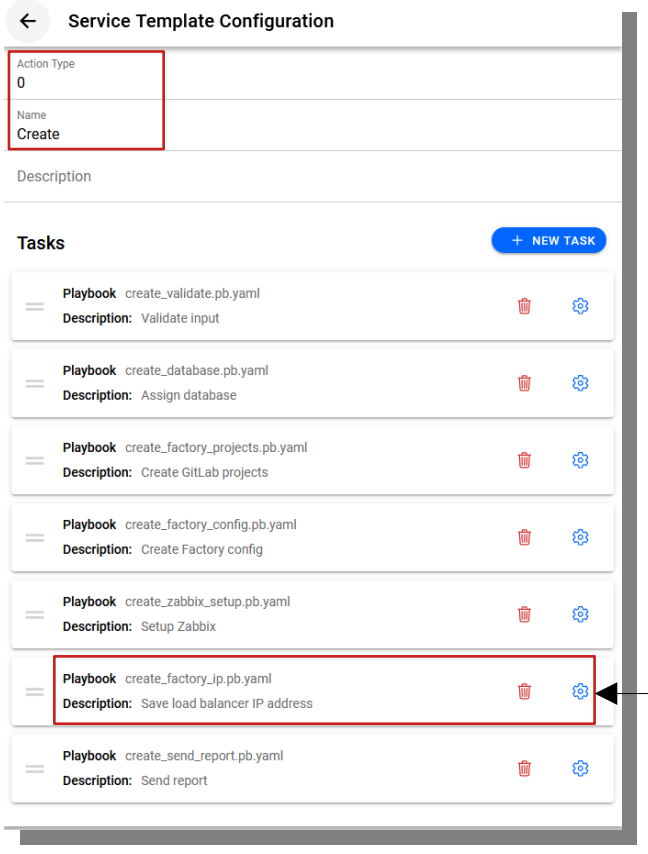
Aktionen werden Service spezifisch definiert und stehen dem Benutzer für die Provisionierung zur Verfügung

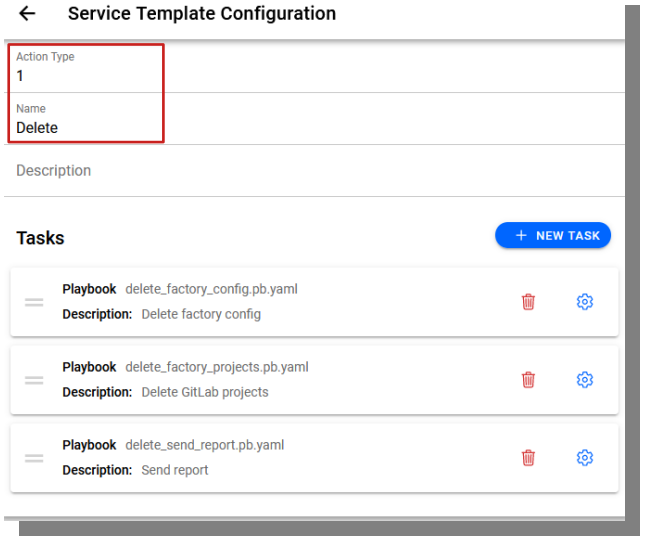
Cloud Deploy Template für ein Zabbix 6.0 Deployment

ZABBIX PREMIUM PARTNER

Cloud Deploy – Service Templates / Tasks







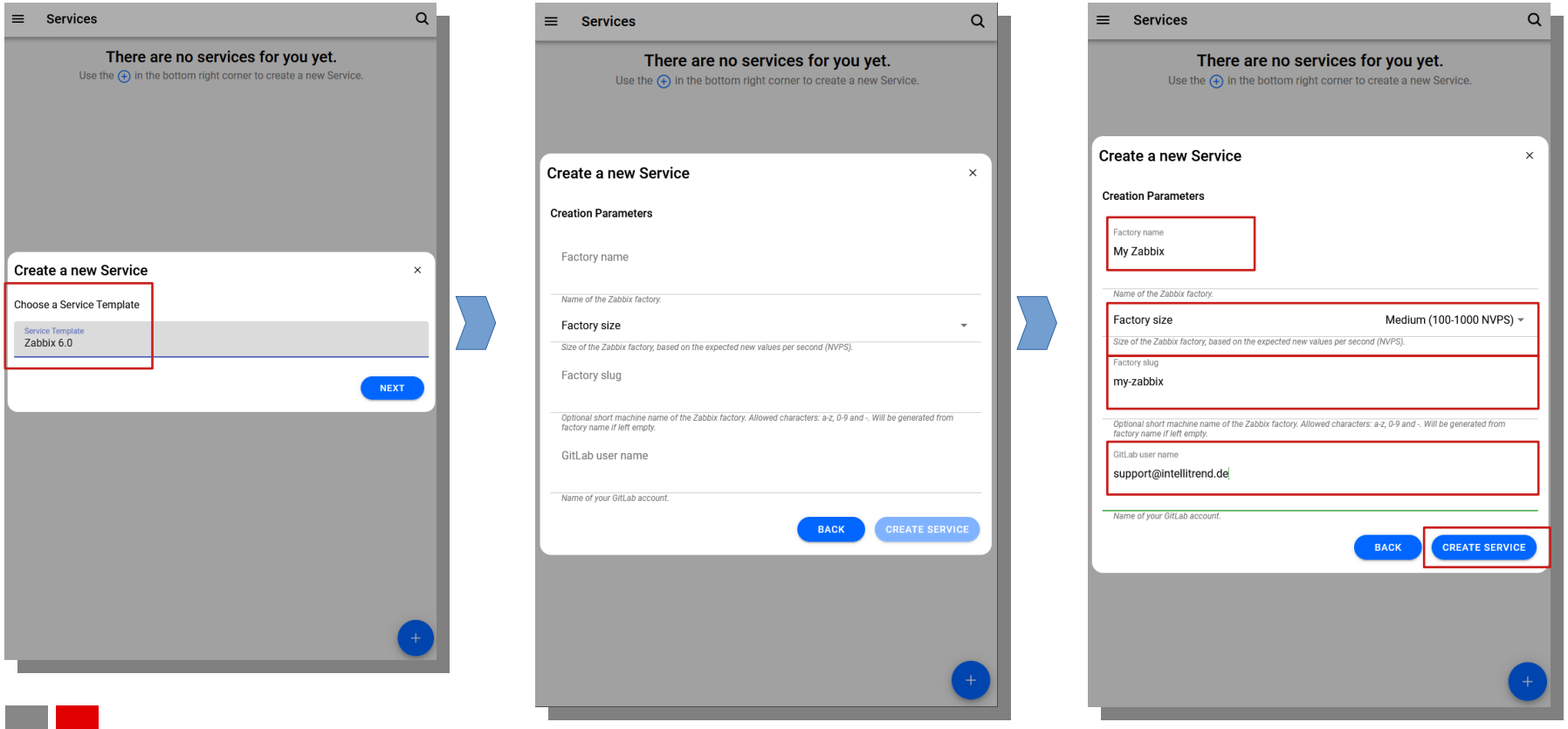
Service Konfiguration für:
Aktualisieren, Erstellen
und Löschen eines Services

Ansible Playbook mit Task
spezifischer Konfiguration

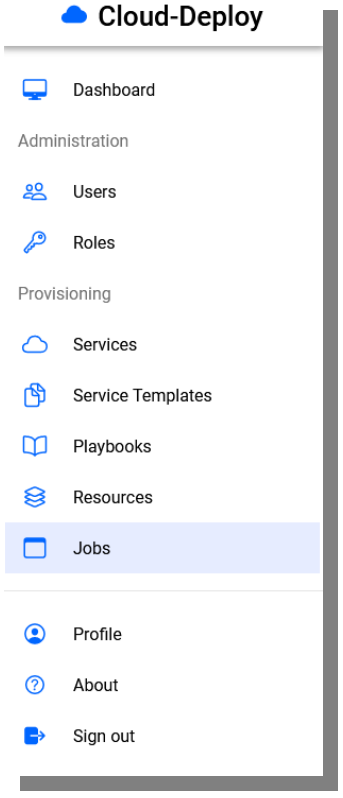
ZABBIX
PREMIUM PARTNER

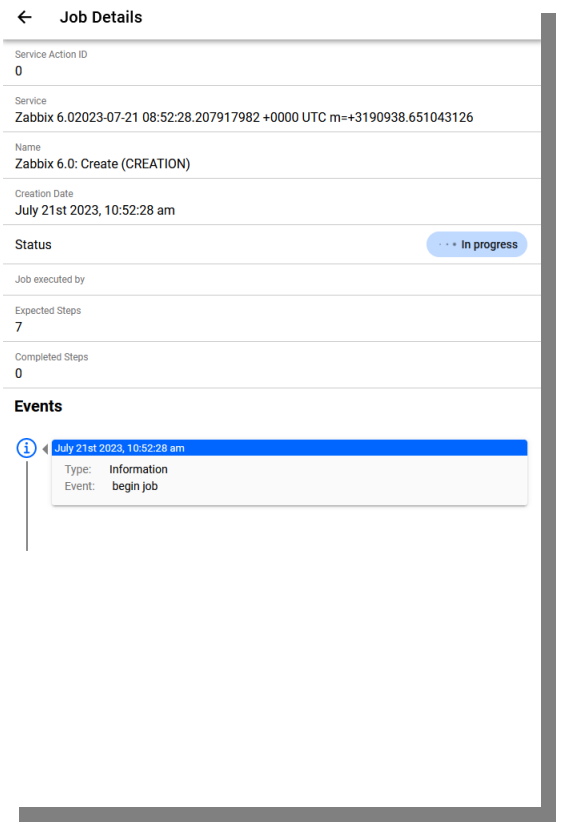
IntelliTrend
IT-Services

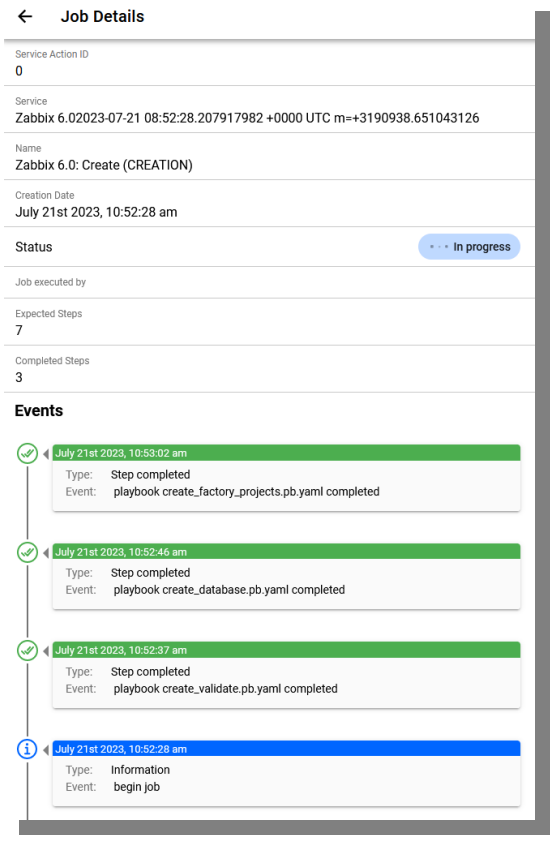
Cloud Deploy – Erstellung eines neuen Service



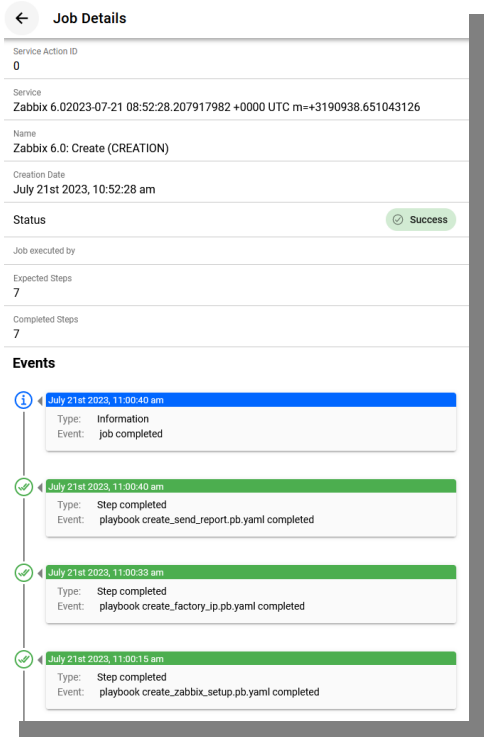
Cloud Deploy – Job Status in Real-Time



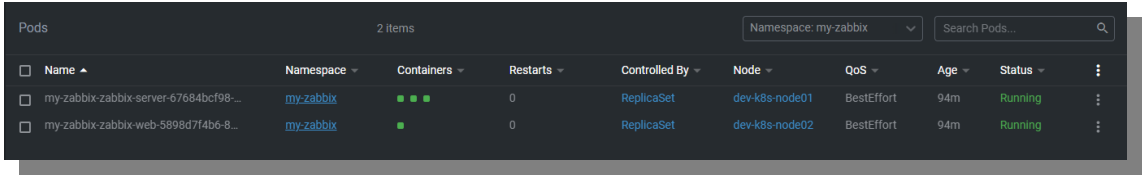




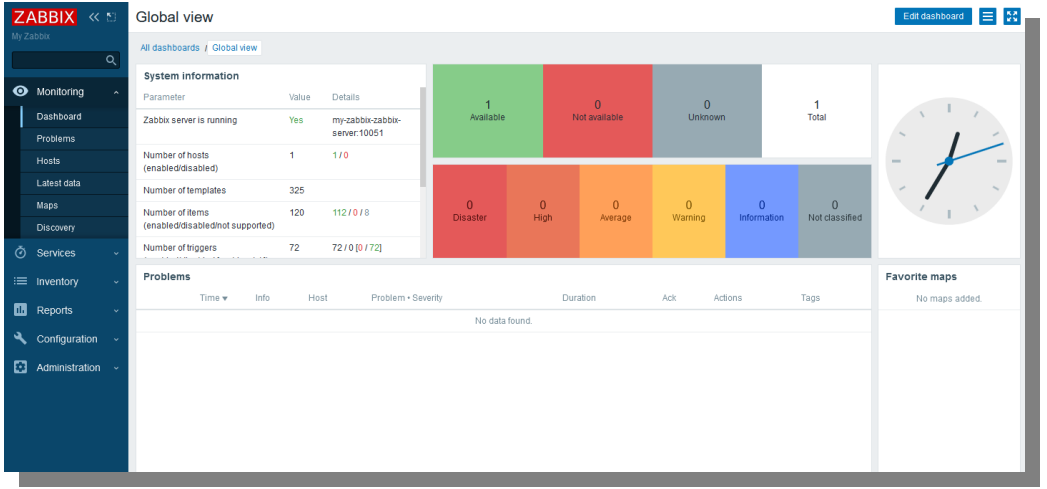
Cloud Deploy – Deployment abgeschlossen



Job successful



Pods erstellt im K8s Cluster



Zabbix Frontend nach Login

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy APIs

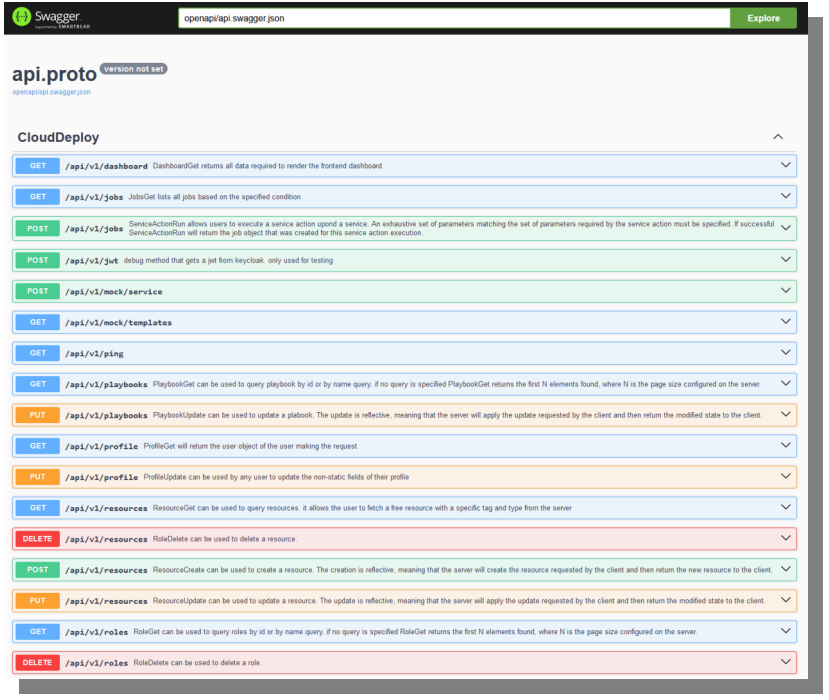
Cloud Deploy - APIs

Alle Vorgänge, die im Benutzerportal ausgeführt werden (und mehr), können auch über APIs ausgeführt werden, was die automatische Bereitstellung signifikant vereinfacht.

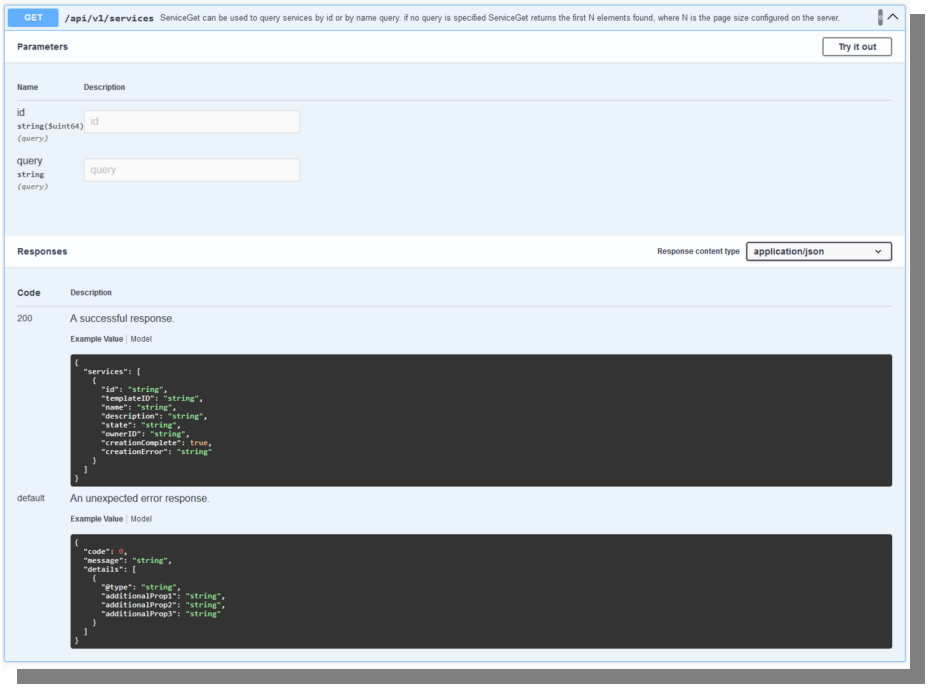
Available APIs:

- REST-Like HTTP API
- gRPC HTTP API
- Terraform

Cloud Deploy – REST-Like API



Dokumentation



Live testing mit swagger

Basiert auf OpenAPI Specifikation, unterstützt Swagger - https://swagger.io/

Cloud Deploy APIs - Terraform

Cloud Deploy – Terraform Provider

```
terraform {
  required_providers {
    clouddeploy = {
      source  = "intellitrend.de/cloud-deploy/cloud-deploy"
      version = "1.0.0"
    }
  }
}

# Configuration for the Cloud Deploy provider
provider "clouddeploy" {
  user        = "admin"
  password    = "admin"
  server_addr = "cloud-deploy.loc"
  server_port = 8090
}
```

Cloud Deploy kann mit Terraform verwendet werden – https://www.terraform.io/

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy – Terraform Service Beispiel

```
# Configuration for a service
resource "clouddeploy_service" "zabbix-60-small" {
  # name of the service template this service is based on,
  # changes after creation cause a re-deployment of the service
  template = "Zabbix 6.0"
  # optional name of the service. if not specified, the name is derived
  # from the template name and the creation time
  name = "Zabbix Small Instance"
  # optional description details for the service
  description = "A small Zabbix instance"
  # optional flag to wait for service jobs to finish
  # note: may cause timeouts on long running tasks
  block = false
  # parameters that are passed to action playbooks,
  # changes will cause service update actions to run on this service
  parameters = {
    "factory_name" = "Zabbix small instance"
    "factory_slug" = "zabbix-small"
    "factory_size" = "small"
    "gitlab_user"  = "admin"
    "email"        = "support@intellitrend.de"
  }
}
```

ZABBIX
PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy – Terraform Apply

```
$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # clouddeploy_service.zabbix-60-small will be created
  + resource "clouddeploy_service" "zabbix-60-small" {
      + block       = false
      + description = "A small Zabbix instance"
      + id          = (known after apply)
      + name        = "Zabbix Small Instance"
      + parameters  = {
          + "email"        = "support@intellitrend.de"
          + "factory_name" = "Zabbix small instance"
          + "factory_size" = "small"
          + "factory_slug" = "zabbix-small"
          + "gitlab_user"  = "admin"
        }
      + status      = (known after apply)
      + template    = "Zabbix 6.0"
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + zabbix-60-small-status = (known after apply)

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

clouddeploy_service.zabbix-60-small: Creating...
clouddeploy_service.zabbix-60-small: Creation complete after 1s [id=180]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

zabbix-60-small-status = "Job Zabbix 6.0: Create (CREATION): begin job"
```

Apply Terraform state
(returns immediately)

Cloud Deploy – Terraform Refresh

```
$ terraform refresh
clouddeploy_service.zabbix-60-small: Refreshing state... [id=180]

Outputs:

zabbix-60-small-status = "Job Zabbix 6.0: Create (CREATION): playbook create_validate.pb.yaml completed"

$ terraform refresh
clouddeploy_service.zabbix-60-small: Refreshing state... [id=180]

Outputs:

zabbix-60-small-status = "Job Zabbix 6.0: Create (CREATION): playbook create_database.pb.yaml completed"

$ terraform refresh
clouddeploy_service.zabbix-60-small: Refreshing state... [id=180]

Outputs:

zabbix-60-small-status = "Job Zabbix 6.0: Create (CREATION): playbook create_factory_config.pb.yaml completed"

$ terraform refresh
clouddeploy_service.zabbix-60-small: Refreshing state... [id=180]

Outputs:

zabbix-60-small-status = "Ready"
```

Überprüfung des Fortschritt mit “refresh”

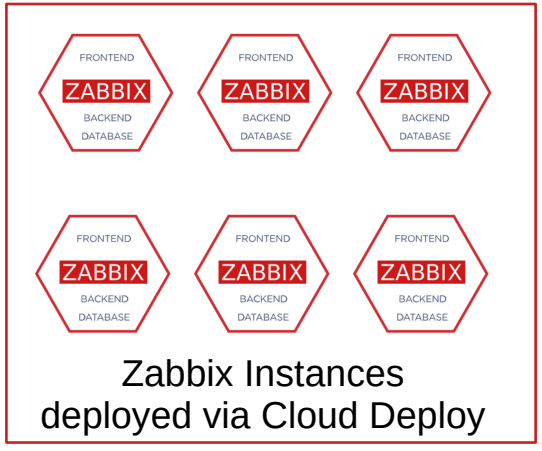
ZABBIX PREMIUM PARTNER

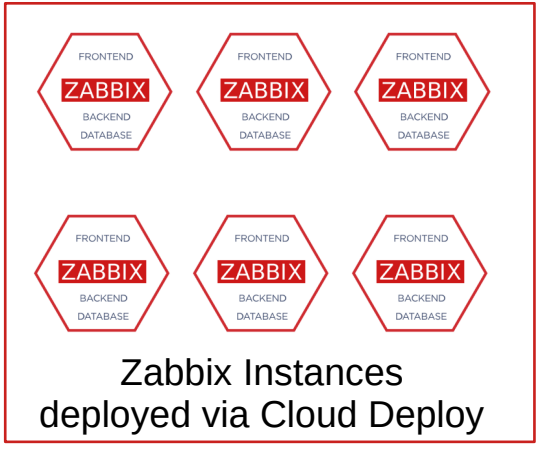
IntelliTrend IT-Services

“One to rule them all” Monitoring der installierten Zabbix Instanzen

Monitoring of deployed Zabbix Instances

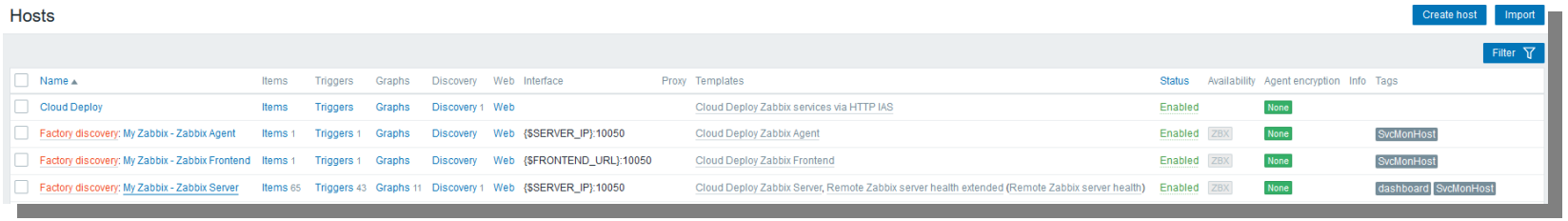
- Ein zentraler "Master"-Zabbix-Server überwacht alle installierten Zabbix-Instanzen
- Jede Zabbix-Instanz definiert einen Dienst auf dem Master, einschließlich Zabbix Server, Zabbix Agent und Frontend
- Der Master Server hat Zugriff auf die internen Metriken jeder Zabbix-Instanz
- Jede Zabbix-Instanz hat ein eigenes Dashboard

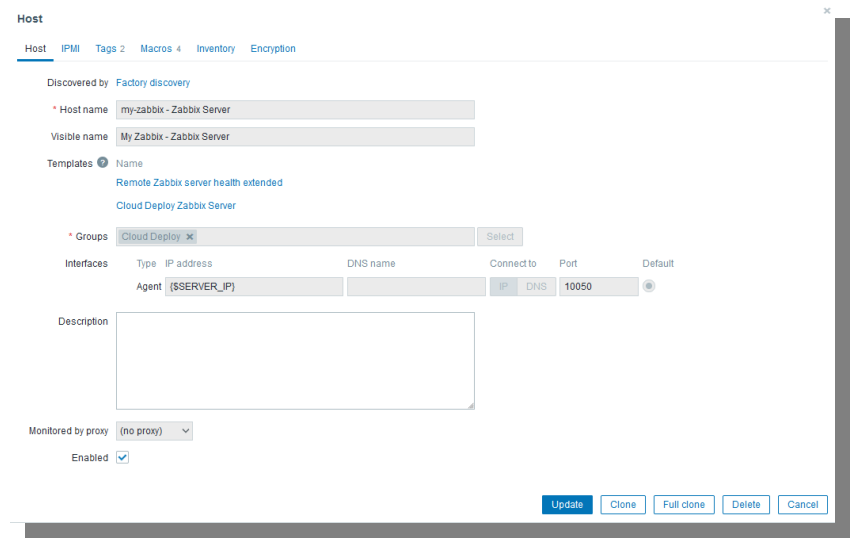




ZABBIX PREMIUM PARTNER

Cloud Deploy – LLD Support



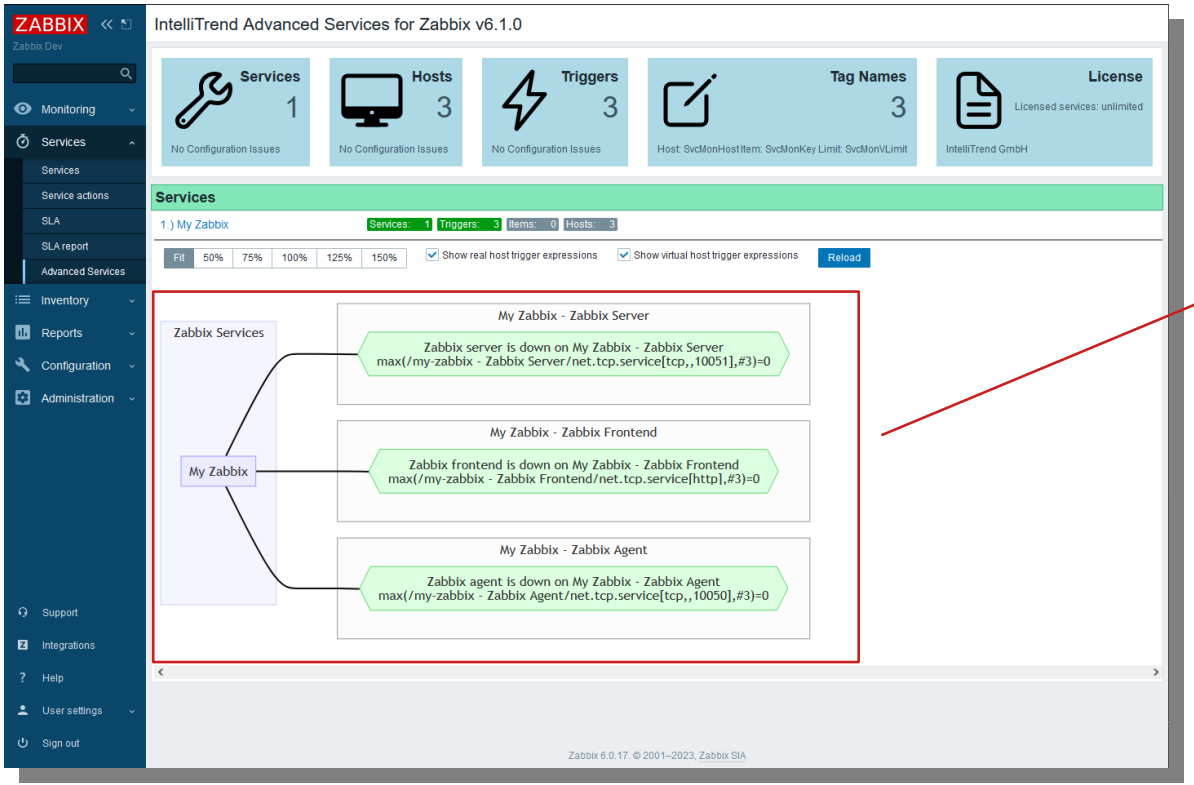


Von Cloud Deploy bereitgestellte Dienste können über LLD “discovered” und von Zabbix überwacht werden.

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy – Service Definition pro Instanz



Pro Zabbix-Service wird automatisch ein Flowchart aus der jeweiligen Service Konfiguration erstellt.

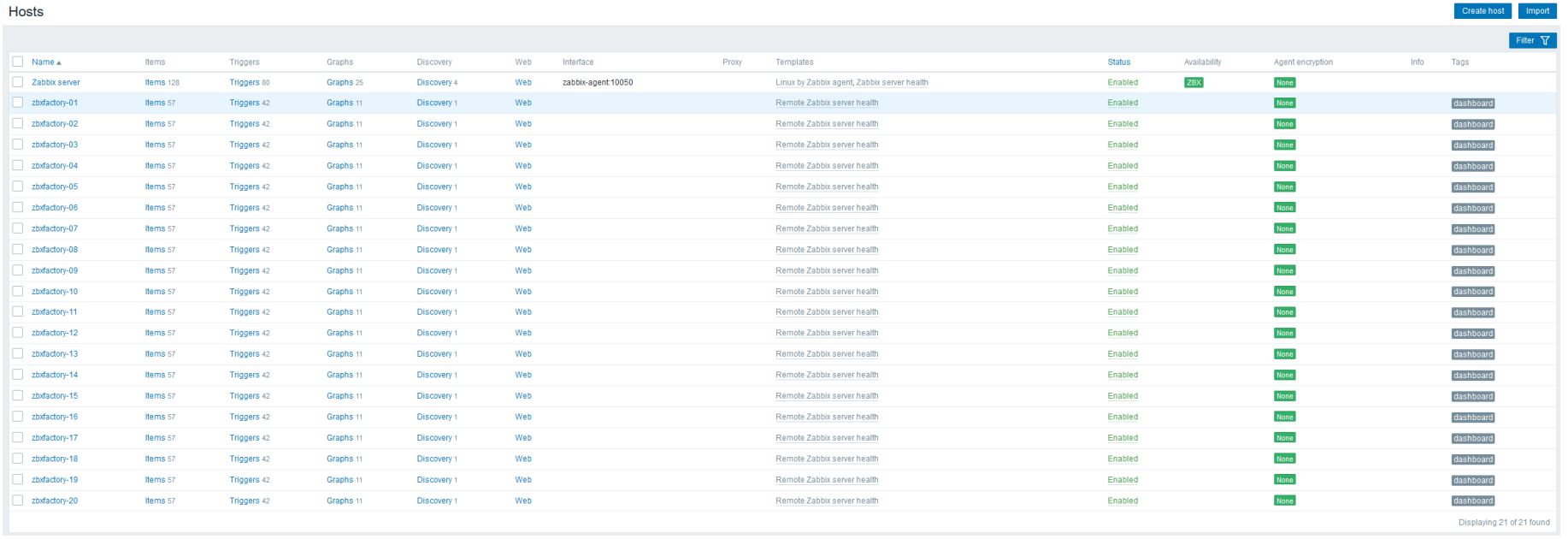
Der Service Status und die zugehörigen Trigger-Stati werden in Echzeit angezeigt.

Auf dem Master für jede Zabbix-Instanz erstellter Service mit grafischer Darstellung

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy – Zabbix Instanzen auf dem Master



Hosts

Create host | Import

Filter

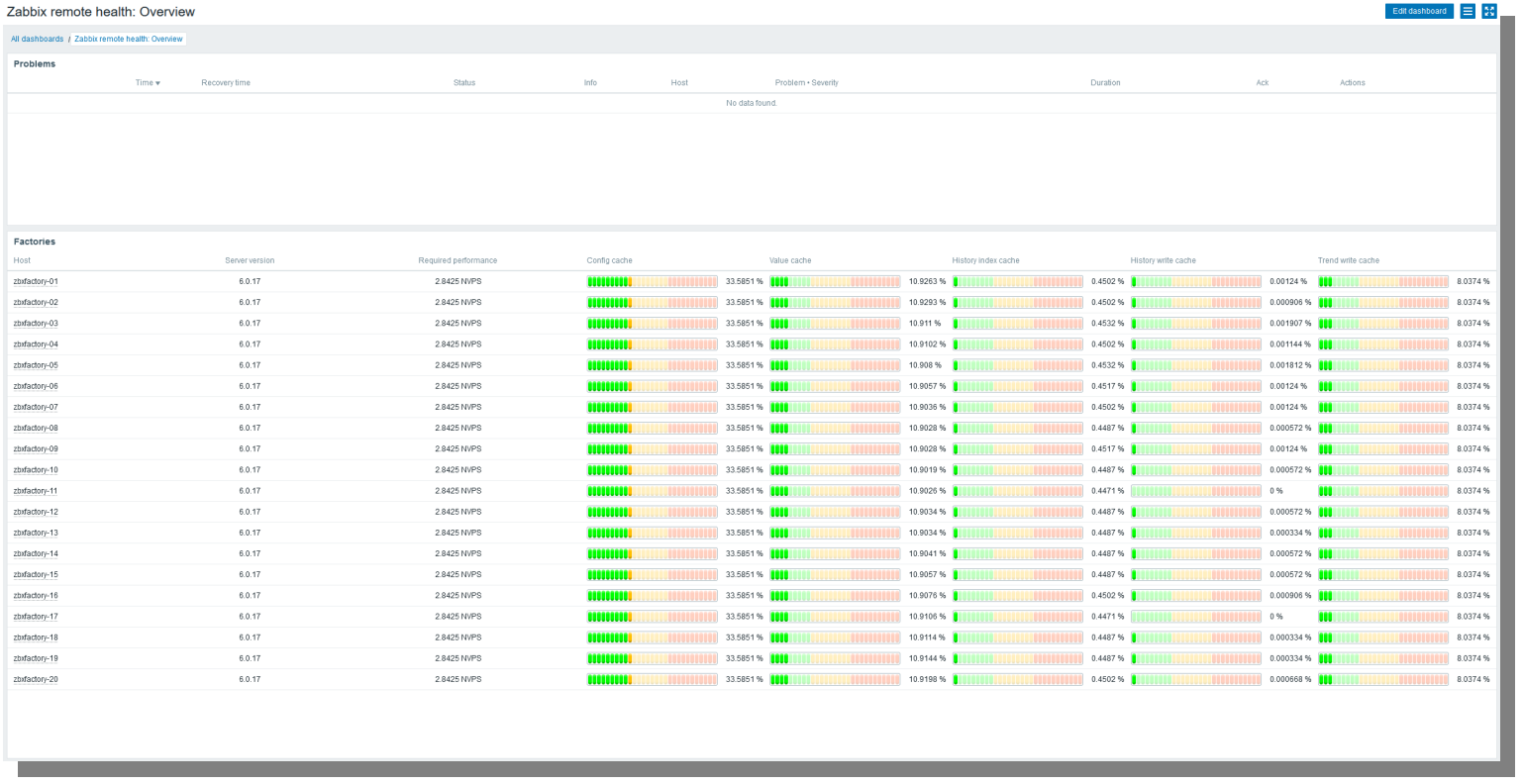
Name ▲	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates	Status	Availability	Agent encryption	Info	Tags
Zabbix server	Items 128	Triggers 80	Graphs 25	Discovery 4	Web	zabbix-agent:10050		Linux by Zabbix agent, Zabbix server health	Enabled	ZBX	None		
zbxfactory-01	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-02	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-03	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-04	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-05	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-06	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-07	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-08	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-09	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-10	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-11	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-12	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-13	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-14	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-15	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-16	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-17	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-18	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-19	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard
zbxfactory-20	Items 57	Triggers 42	Graphs 11	Discovery 1	Web			Remote Zabbix server health	Enabled		None		dashboard

Displaying 21 of 21 found

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Cloud Deploy – Zabbix Instanzen auf dem Master



“Health Status Dashboard” aller Zabbix Instanzen auf dem Master

ZABBIX PREMIUM PARTNER

IntelliTrend IT-Services

Individuelle Zabbix Installationen in On-Premises Kubernetes Umgebungen

Danke!

IntelliTrend GmbH

www.intellitrend.de

Kontakt: Wolfgang Alper

wolfgang.alper@intellitrend.de

ZABBIX PREMIUM PARTNER