Kubernetes Clusters as a Service

Gardener

The Kubernetes Botanist
Disclaimer

The information in this presentation is confidential and proprietary to SAP and may not be disclosed without the permission of SAP. Except for your obligation to protect confidential information, this presentation is not subject to your license agreement or any other service or subscription agreement with SAP. SAP has no obligation to pursue any course of business outlined in this presentation or any related document, or to develop or release any functionality mentioned therein.

This presentation, or any related document and SAP's strategy and possible future developments, products and or platforms directions and functionality are all subject to change and may be changed by SAP at any time for any reason without notice. The information in this presentation is not a commitment, promise or legal obligation to deliver any material, code or functionality. This presentation is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. This presentation is for informational purposes and may not be incorporated into a contract. SAP assumes no responsibility for errors or omissions in this presentation, except if such damages were caused by SAP’s intentional or gross negligence.

All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.
Strategic Open Source Cloud Native Projects
Contributions and Commitments from SAP
SUSE – SAP Collaboration

SUSE actively contributes to the Gardener project
Gardener readily runs on SUSE OpenStack Cloud
SUSE supports SUSE Linux Enterprise Server for Gardener
Our **Gardener** Mission Statement

Provide Secure Kubernetes Clusters-as-a-Service to SAP and customers homogeneously everywhere with minimal TCO.

[https://gardener.cloud](https://gardener.cloud)
Project Gardener: Consistent Kubernetes at Scale
Apache 2.0 license, open source

https://gardener.cloud

lingua franca

Multicluster DevOps UI

Hyperscale Clouds, Public Clouds, Private Clouds, On-Premise
<table>
<thead>
<tr>
<th>Cloud Provider</th>
<th>Platform</th>
<th>Cluster Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Web Services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seed-operator-aws</td>
<td>used by 3 clusters</td>
<td></td>
</tr>
<tr>
<td>trial-secret-aws</td>
<td>(currently unused)</td>
<td>Owner: garden-trial</td>
</tr>
<tr>
<td>Microsoft Azure Cloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seed-operator-az</td>
<td>used by 1 cluster</td>
<td>7532a3f4-1027-4b01-910b-48bbdb02d1496</td>
</tr>
<tr>
<td>trial-secret-gcp</td>
<td>(currently unused)</td>
<td>Owner: garden-trial</td>
</tr>
<tr>
<td>Google Cloud Platform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seed-operator-gcp</td>
<td>used by 1 cluster</td>
<td></td>
</tr>
<tr>
<td>trial-secret-gcp</td>
<td>(currently unused)</td>
<td>Owner: garden-trial</td>
</tr>
<tr>
<td>OpenStack</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seed-operator-es-eude-200</td>
<td>used by 1 cluster</td>
<td>CP_K8S_PROD / sap-k8s-canary</td>
</tr>
<tr>
<td>Alibaba Cloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>seed-operator-alicloud</td>
<td>used by 1 cluster</td>
<td></td>
</tr>
<tr>
<td>trial-secret-alicloud</td>
<td>(currently unused)</td>
<td>Owner: garden-trial</td>
</tr>
</tbody>
</table>
How is the Gardener implemented?

Well, what is the definition of Kubernetes anyway?

Kubernetes is an open-source system for **automating deployment, scaling, and management** of containerized applications/software.

as Kubernetes is software itself, ...

We use Kubernetes to deploy, host, and operate Kubernetes. 
End-User clusters are “seeded” into already existing clusters. ➔ **Inception, or Kubeception**
Common Kubernetes Cluster Setup

The **green machines** host the control plane, often in HA and on separated hardware (usually underutilized or, worse, overutilized).

The **blue machines** host the actual workload and are managed by Kubernetes (usually pretty well utilized).
Gardener Kubernetes Cluster Setup

Zooming into the Seed Cluster reveals... Multiple Shoot Cluster Control Planes

Gardener Machine Controller Manager
- Machine Provisioning
- Self-Healing
- Auto-Update
- Auto-Scaling
Kubernetes Cluster

Master

Control Plane

Worker

Container Runtime
- Leonardo
- Data Hub
- HANA

Business Workloads

Target IaaS/Account

Admins/Users/Customers

Leonardo

Data Hub

HANA

Business Workloads
100% Kubernetes

Open Source

CNCF Officially Certified

Kubernetes In Kubernetes In Kubernetes!

Hybrid Cloud

Homogeneous Infrastructure

Architecture in Three Components

Gardener Runs the Gardener
A Kubernetes controller responsible for managing custom resources

End-User Cluster Shoot Cluster Contains Shoot Cluster’s Control Plane as Workload
Only Worker Nodes

What Is Gardener?

@Antheajung

An Extended API Server & A Bundle of Kubernetes Controllers

That Defines and Manages New API Objects Used for Management of Kubernetes Cluster

A Service to Manage Large-Scale Kubernetes Cluster

Interact With Central UI

Gardener Dashboard

Command Line Client

Written in Go
Shoot Cluster

Account/Network Prep

Garden Cluster

Seed Cluster

Administrator

kubectl

Worker

Kubelet + Container Runtime

Gardener Dashboard

Gardener API Server

G Controller Manager

Gardener

Garden Cluster API LB

Ingress LB

Storage

[K8s] DS, RS, SS, J, ...

[CRD] Shoot, Seed, ...

IaaS

Monitoring

Logging

Terraformer

S

J

Extension Option

Private Cloud & Legal Geographies

End-User

kubectl

Worker

Kubelet + Container Runtime

etcd Main

BckUp SS

etcd Events SS

API Server

VPN D

Scheduler D

Controller Manager D

Addon Manager D

Machine Controller

Shoot Cluster Control Plane

Monitoring VPN

Logging

Target IaaS

Target IaaS/Account

Garden Cluster

Seed Cluster

Shoot Cluster

IaaS

Logging

Monitoring

Add-on Manager

Controller Manager

Scheduler

etcd Events

etcd Main

API Server

IPFS

Shoot Cluster

Shoot Cluster API LB

Seed Cluster API LB

Garden Cluster API LB

Ingress LB

Storage

[K8s] DS, RS, SS, J, ...

[CRD] Shoot, Seed, ...

IaaS

Logging

Monitoring

IaaS

Target IaaS

Target IaaS/Account

Garden Cluster

Seed Cluster

Shoot Cluster

IaaS

Logging

Monitoring

IaaS

Target IaaS

Target IaaS/Account

Garden Cluster

Seed Cluster

Shoot Cluster

IaaS

Logging

Monitoring

IaaS

Target IaaS

Target IaaS/Account
Gardener is recursive!
Security

We are paranoid about security

Gardener is a “Secure by default” environment

Regular Penetration tests (internal & 3rd party)

Constantly adding 2nd lines of defense

Working on pre-disclosure process for vendors

Past hardening activities:

Project “Karydia”

Automatically apply “more” secure defaults to deployments.
- seccomp profiles, AppArmor, SELinux, …
- automountServiceAccountToken
- Network policies

Attempt to prevent users from doing very dangerous things
- Limit pod creation with PSPs

No one-size-fits all, different profiles for different use cases
- Is 3rd party "untrusted" code allowed?

Karydia will not forbid anything: users can overrule Karydia’s decision to deny something.

Open Source development: https://github.com/karydia/karydia

Get involved!
Extensibility
Gardener Enhancement Proposal (GEP)

Gardener enhancement proposal (GEP) for Gardener extensibility
https://github.com/gardener/gardener/blob/master/docs/proposals/01-extensibility.md

- Operator (Gardenlet)
- Workflow engine orchestrating special purpose controllers with CRDs
- Control Plane Relocation
- Ring: High Availability across DCs and for on-premise
Currently, centralized but hierarchical setup

SAP Cloud Platform

Gardener

Seed 1

Seed 2

Seed 3
Next: de-centralized setup (local cluster operator)
Let's order a Kubernetes Cluster

```yaml
apiVersion: garden.sapcloud.io/v1
kind: Shoot
metadata:
  name: leiwen-aws
  namespace: garden-scp
spec:
  dns:
    domain: leiwen-aws.scp.k8s.sapcloud.io
    hostedZoneID: Z3IKKBW5GMZJOI
    kind: aws
  infrastructure:
    kind: aws
    region: eu-central-1
    secret: scp-k8s-dev
    vpc:
      cidr: 10.250.0.0/16
      kubernetesVersion: 1.8.5
      ...
  seedName: seed-aws-eu1
  workers:
    - autoScalerMax: 5
      autoScalerMin: 2
      machineType: m4.xlarge
      name: cpu-worker
      volumeSize: 50Gi
      volumeType: gp2
```
Actually, these are several Controllers, but for Simplicity they are represented as one
Gardener „Hierarchy“ of Resources/Responsibilities

Non-technical names

Legend
- Garden
- Seed
- Shoot
Machine Controller Manager

(Cluster API Project (https://github.com/kubernetes-sigs/cluster-api))
The Problem

- Provisioning and de-provisioning of nodes is out of the scope of standard Kubernetes right now
- Gardener was using terraform scripts for provisioning and this is proving unmanageable
- No mechanism existed to smoothly scale clusters or upgrade cluster nodes for all providers

The Solution

- Machine Controller Manager (MCM) provides a Kubernetes-native declarative way to describe the relevant aspects of the nodes required in the Kubernetes cluster
- It enables support for different cloud providers by the way of modular plugins
- It enables easy scaling of the cluster and upgrade of cluster nodes
MCM Model

Model for Kubernetes deployments (Deployment, ReplicaSet, Pod) works great so why not use if for machines?
MCM Custom Resource Objects

**AWS-Machine-Class (Template)**
- Name: v1
- Machine Type: t2.large
- Disk Size: 50GB
- Secret: test-secret

**Secret**
- Name: test-secret
- Cloudconfig: abc....xzy
- AccessKeyId: abc123
- SecretAccessKey: xyz789

**Machine**
- Name: test-machine
  - MachineClass: v1

**Machine-set**
- Name: test-ms
  - Replicas: 3
  - MachineClass: v1

**Machine-deployment**
- Name: test-md
  - Replicas: 3
  - UpdateStrategy: Rolling
  - MachineClass: v1
Working of MCM
Now assume that all the nodes resources are nearly consumed and a new pod is created.
Getting Started
Gardener Setup Projects

https://github.com/gardener/landscape-setup
current stable project
(not maintained anymore)

https://github.com/gardener/garden-setup
„Virtual“ Gardener
Pre-requisite is an existing Kubernetes cluster
(new setup, expect some hick-ups)
Key Takeaways
Key Takeaways

Gardener is a mature platform, SAP runs productive workloads on it.

We currently require ~1FTE to run 2000 clusters.

Gardener has arrived in the community and is well known by now.

Several public and private cloud providers have productive offerings based on Gardener.

Gardener is on its way to become the open standard multi-cloud, multi-cluster Kubernetes solution.
Thank you.

Contact information:
Dirk Marwinski
dirk.marwinski@sap.com

Project Gardener
SAP Cloud Platform Kubernetes

https://gardener.cloud