TRAINING GOALS:

Linux containers are changing the way companies think about service development and deployment. Containers play a vital role in the modern data-center, and Docker is leading the way. And Kubernetes is the leading open-source system for automating deployment, scaling and management of containerized applications.

Participants will first gain a basic understanding of Linux containers and proceed with learning the most important features of Docker Community Edition (CE) as well as their installation, initial setup and daily administration.

The second part of the course introduces participants to the basic concepts and architecture of Kubernetes, its initial install, setup and access control, Kubernetes Pods and Workloads, Scheduling and node management, Accessing the applications, Persistent storage in Kubernetes and finally its Logging, Monitoring and Troubleshooting facilities.

The third part deals with Helm, the Kubernetes Package Manager.

This course doesn’t only prepare delegates for the daily administration of Docker & Kubernetes systems but also for the official Certified Kubernetes Administrator (CKA) and Certified Kubernetes Application Developer (CKAD) exams of the Cloud Native Computing Foundation (CNCF).

Course objectives: 1 day Docker + 3 days Kubernetes + 1 day Helm

Structure: 50% theory 50% hands on lab exercises

Audience:

System administrators, developers and DevOps who want to understand and use Docker and Kubernetes in enterprise and cloud environments.

CONSPECT:

PART I. DCK-101 Docker Intro

- CONTAINER TECHNOLOGY OVERVIEW
  - Application Management Landscape
  - Application Isolation
Resource Measurement and Control
- Container Security
- Container Security
- Open Container Initiative
- Docker Ecosystem
- Docker Ecosystem (cont.)
- Lab Tasks:
  - Container Concepts runC
  - Container Concepts Systemd

INSTALLING DOCKER
- Installing Docker
- Docker Architecture
- Starting the Docker Daemon
- Docker Daemon Configuration
- Docker Control Socket
- Enabling TLS for Docker
- Validating Docker Install
- Lab Tasks:
  - Docker Basics
  - Install Docker via Docker Machine
  - Configure a docker container to start at boot.

MANAGING CONTAINERS
- Creating a New Container
- Listing Containers
- Viewing Container Operational Details
- Running Commands in an Existing Container
- Interacting with a Running Container
- Stopping, Starting, and Removing Containers
- Copying files in/out of Containers
- Inspecting and Updating Containers
- Lab Tasks:
  - Docker Images
  - Docker Platform Images

MANAGING IMAGES
- Docker Images
- Listing and Removing Images
○ Searching for Images
○ Downloading Images
○ Committing Changes
○ Uploading Images
○ Export/Import Images
○ Save/Load Images
○ Lab Tasks:
  ○ Docker images
  ○ Docker Platform images

○ CREATING IMAGES WITH DOCKERFILE
  ○ Dockerfile
  ○ Caching
  ○ docker image build
  ○ Dockerfile Instructions
  ○ ENV and WORKDIR
  ○ Running Commands
  ○ Getting Files into the Image
  ○ Defining Container Executable
  ○ Best Practices
  ○ Lab Tasks:
    ○ Dockerfile Fundamentals

PART II. KBS-103 Kubernetes Admin with CKA & CKAD exam.prep.

○ Kubernetes introduction
  ○ Cloud computing in general
  ○ Cloud types
  ○ Cloud native computing
  ○ Container orchestration
  ○ Kubernetes
  ○ Kubernetes concepts
  ○ Kubernetes objects categories
  ○ Custom resource definitions
  ○ Kubernetes architecture
  ○ Kubernetes master
  ○ Kubernetes node
  ○ Kubernetes Lab: Health check
- Accessing Kubernetes
  - Accessing the Kubernetes cluster
  - Controlling access to the API
  - Authorization
  - Role Based Access Control
  - Roles and ClusterRoles
  - Role bindings
  - Admission control
  - Kubernetes Lab: Accessing API

- Kubernetes Workloads
  - The pod
  - Our first Pod
  - Operations on pods
  - Pod Status and Lifecycle Pod Status and Lifecycle (cont)
  - Pod probe examples
  - RestartPolicy examples
  - InitContainers Pod resource management
  - Pod security context
  - Patterns for Composite Containers
  - ReplicationController and ReplicaSet
  - Working with ReplicationController, ReplicaSet
  - Deployments
  - Working with Deployments
  - Kubernetes Lab: Workloads

- Scheduling and node management
  - The Kubernetes Scheduler
  - Pod priorities and preemption
  - Assigning Pods to Nodes
  - Assigning Pods to Nodes - Node affinities Assigning Pods to Nodes -
  - Taints and tolerations
  - Managing nodes
  - Kubernetes Lab: Scheduling

- Accessing the applications
  - Services
  - Service types
  - Working with Services
- Working with Services
- Ingress
- Ingress definition
- Working with Ingress
- Network Policies
- Network Policy example
- Kubernetes Lab: Accessing Applications

- Persistent storage in Kubernetes
  - Volumes: Volume example Volume types
  - Persistent Volumes
  - Persistent Volume example
  - Dynamic PVC provisioning
  - Secrets
  - Using Secrets as environmental variables
  - Using Secrets as volumes
  - ConfigMaps
  - Kubernetes Lab: Persistent Storage

- Kubernetes Special Workloads
  - StatefulSets - Limitations
  - StatefulSet example
  - StatefulSet example with PVC
  - Jobs, CronJobs
  - Jobs example
  - CronJobs example
  - DaemonSets
  - Kubernetes Lab: Special workloads

- Logging, monitoring and troubleshooting
  - Logging architecture
  - Monitoring
  - Troubleshooting
  - Kubernetes Lab: Logging and Monitoring

- Installing and upgrading Kubernetes
  - Picking the right solution
  - One node Kubernetes install
  - Kubernetes universal installer
  - Install using kubeadm on CentOS
Upgrading Kubernetes
Kubernetes Networking Kubernetes
Lab: Upgrading Kubernetes
  Appendix: Application containers
  Application containers
  Containers on Linux
  Container runtime

PART III. Helm Package Manager

Introduction to Helm
  What is Helm?
  Helm concepts
  Helm v2 (legacy) components
  Helm v3 components
  Installing Helm
  Helm Lab: Installing Helm

Using Helm
  Generic options and help
  Working with repositories
  Finding charts
  Installing a release
  List releases
  Upgrade/rollback releases
  Uninstalling releases
  Helm Lab: Using Helm

Helm charts
  Introduction to charts
  The structure The Chart.yaml File
  The components of a Chart
  Chart dependencies
  Chart dependencies (cont.)
  Managing Charts with helm
  Helm Lab: Working with charts

Chart Templates
  Writing Templates Templates and Values
  Dependencies and values Dependencies and values
- Chart lifecycle hooks Functions and pipelines
- Flow control
- Variables
- Named templates
- Helm Lab: Writing templates

- Helm plugins
  - Building plugins
  - Helm Lab: Helm plugins

**REQUIREMENTS:**

Proficiency with the Linux CLI. A broad understanding of Linux system administration.

**Difficulty level**

[ ] [ ] [ ] [ ] [ ]

**CERTIFICATE:**

After completing the course, participants receive a certificate signed by Component Soft.

**TRAINER:**

Authorized Component Soft Trainer.