openSUSE Kubic
exploring the containerized frontiers

Thorsten Kukuk
Distinguished Engineer
Future Technologies
kukuk@suse.com
Agenda

• What is Kubic?
• openSUSE MicroOS
• openSUSE Kubic
• Transactional Update
• Health Checker
• Container Runtime
• Kubernetes
• Demo Time!
• How we build openSUSE Kubic
Kubic Project

Founded in May 2017

Sub-Project of the openSUSE Project

Focused on Container and Core Operating System Technologies

Project: https://kubic.opensuse.org
Kubic Project

Based on openSUSE Tumbleweed
  • openSUSE MicroOS
  • openSUSE Kubic

Targeting the latest upstream container technologie
  • Simple Installer
  • Transactional Operating System Updates
  • Podman and CRI-O as container runtime
  • Kubeadm cluster bootstrapper
  • Whatever our community want to explore
openSUSE MicroOS
What is openSUSE MicroOS?

openSUSE MicroOS:
Modern, Single Service Linux Operating System, optimized for containers and large deployments.

openSUSE MicroOS inherits the openSUSE Tumbleweed knowledge while redefining the operating system into a small, efficient and reliable distribution.
Single Service:

I get all the required support from the OS

I’m small but powerful to run as many containers as needed
Multi Purpose:

I'm designed to carry more than what fit's into me. Too heavy, too expensive.
openSUSE MicroOS – Highlights

• The “Perfect” Container Host
• Read-only root filesystem
• Fully automated Transactional-Updates
• Automatic initial configuration during first boot (Account, Network)
• Podman as default container runtime
• Rolling Release (openSUSE Tumbleweed based)
Why MicroOS as OS development platform?

- **Limited number of packages**
  - But: full access to all openSUSE Tumbleweed packages
- **Limited scope**
  - SELinux on SLES: lot of functionality to cover
  - SELinux on MicroOS: very limited functionality, easy to keep running and current *(if you have somebody with the knowledge to get it running first!)*
- **Quick rollback**
- **Very flexible with system roles and patterns**
- **Single Service → no breakage of other functionality**
What are we developing with it?

- **Transactional updates**
- **Security:**
  - Apparmor – read-only root filesystem
  - SELinux – Framework, Policy, integration in transactional-update
  - IMA/EVM
- **Rolling out of Kubernetes via kubeadm**
- **Cluster Management**
  - Installation
  - Update
- **IoT**
  - https://kubic.opensuse.org/blog/2019-01-30-kubiconaarch64/
openSUSE Kubic
Pets or Cattle?

“In the old way of doing things, we treat our servers like pets, for example Bob the mail server. If Bob goes down, it’s all hands on deck. The CEO can’t get his email and it’s the end of the world. In the new way, servers are numbered, like cattle in a herd. For example, www001 to www100. When one server goes down, it’s taken out back, shot, and replaced on the line.”

(http://cloudscaling.com/blog/cloud-computing/the-history-of-pets-vs-cattle/)
openSUSE Kubic

• Kubernetes Cluster using
  • openSUSE MicroOS as OS
  • CRI-O
• “Cattle”
• System Roles:
  • MicroOS with OCI Container
  • kubeadm (master)
  • kubeadm (salt minion)
  • …
• Contains everything to install, run and maintain a kubernetes cluster
openSUSE Tumbleweed

• Multipurpose OS
• “Pet”
• Useable for everything
  • Universal
  • With the overhead to be useable for everything
  • With the limitations to be useable for everything
Common Mistakes

Treat a container like a virtual machine
• This will reduce most of the advantages of the container model
  (https://cloud.google.com/solutions/best-practices-for-building-containers)
• Look at Native Cloud Applications
  • Auto-scaling on any managed infrastructure

Treat your cattle's like a PET
• May work with 10-30 node beginner cluster, but average is > 100 Nodes

Hardware consolidation
• Workload for 4.5 Nodes → Cluster size of 5 Nodes
  => How do you want to schedule an update or replacement of a Node?
Transactional Update
“I NEVER want to touch a running system”
- Every SysAdmin, ever
Upgrading a Running System Is DANGEROUS

• Services are running
• Users are doing things
• Software changes things (sometimes on purpose!)
• Not all packages are packaged properly (sorry!)
Reality does not make things easy

openSUSE Tumbleweed – rolling release since many years
→ everything must be fine, or?

Rolling Releases bring larger challenges
• Intrusive Updates (SysV init → systemd)
• Major version updates of complex stacks (GNOME, KDE)
• What should I do if the update breaks my system?
Critical Mission Systems are even worse

- **Update should not interrupt services**
  Service interruption more expensive than regular reboots

- **Updates need to be fully applied perfectly, or not changes made**
  System state should never be undefined or questionable
  RPM post-installation scripting can lead to an undefined state
What is a Transactional Update

An update that

• **Is atomic**
  – Either fully applied, or not applied at all
  – Update does not influence the running system

• **Can be rolled back**
  – A failed or incompatible update can be quickly discarded to restore the previous system condition
Updates with transactional-update

1. Snapshot of current system
2. Create new target snapshot
3. Update system and set as default for next boot

Current root file system is not modified!
Learn More about Transactional Updates

Transactional Updates in SLE15 SP1 [TUT1110]

Wednesday, 10:00am – 11:00 am, Fisk 1
Health Checker
Health Checker

Checks for errors during boot phase

- Error with new snapshot:
  - Rollback to last known working snapshot
- Error with already successful booted snapshot:
  - Try reboot
  - Shutdown services, inform admin
- Needs access to harddisk
CRI-O

• Kubernetes focused container runtime
• Official Kubernetes Project with aligned releases
• Open Project with broad contributor base including Red Hat, SUSE, Intel, Google, IBM, Alibaba, etc
• No large daemon
• Lightweight, more easily securable with apparmor, seccomp, cgroups, etc
Podman

• CLI tool for running OCI-containers in Pods
• Shares core concepts and libraries with CRI-O
  • Roadmap includes integration with CRI-O, replacing current container management layer
• Capable tool for standalone container hosts

• Able to pull & run containers from any OCI registry (includes Docker Hub)
• Syntax very similar to Docker Engine
  • `podman run` <=> `docker run`
  • `podman pull` <=> `docker pull`
Podman ‘pods’

• Additional functionality for managing k8s-like ‘pods’
  • podman pod create|rm|start|stop|ps
• Bulk manage containers and images
  • podman rm --all
  • podman rmi --all
• Supports systemd socket activation
  • Only start a container when a user tries to connect to the service
Buildah

- Builds containers
- From scratch, based on images, or using a Dockerfile
- OCI or Docker Image Format
- Mount/unmount live containers root filesystem
- Create new container layers from filesystem changes
Skopeo

Manages registries
• Copying containers to and from registries
• Inspecting a remote image showing its properties and layers without pulling
• Deleting an image from an image repository.
• Authentication supported
Kubernetes/kubeadm
kubeadm

• Upstream Kubernetes cluster bootstrapping tool
• Under Active Development with expanding feature set
• Released as GA since v1.13
• Decoupled from kubelet (Kubernetes Node Agent)
  • Allows upgrade of OS + kubelet independently of control plane
  • Control plane upgrades require user interaction eg. “kubeadm upgrade v1.13.0”
Cluster Setup with kubeadm

- Check https://en.opensuse.org/Kubic:kubeadm for updates to this procedure
- Install 2+ (recommended 3+) systems with openSUSE Kubic
- Choose kubeadm Node system role

- Start install with defaults, or customise network, software, etc as you wish
- Once installed, ssh / login to system that will be Master Node

- Run the following
  - `kubeadm init --cri-socket=/var/run/crio/crio.sock --pod-network-cidr=10.244.0.0/16`
  - Copy/record the the “kubeadm join” command provided by the successful “kubeadm init”
openSUSE Kubic and kubeadm

Certified Kubernetes 1.13
Demo Time!
What you should have seen

• Boot three images Kubic-1, Kubic-2 and Kubic-3
• Run “kubeadm init” on first image
• mkdir ${HOME}/.kube; cp “/etc/kubernetes/admin.conf ${HOME}/.kube/config
• Setup flannel
• Deploy kured
• Setup salt master on first machine, accept minions
  • Salt master is salt minion, too
• Run “salt -G 'roles:kube-worker' cmd.run kubeadm join <token>”
• Install MetalLB Loadbalancer
• Deploy “Hello World” on the kubernetes cluster
How we build openSUSE Kubic
Open Source, Unique Challenges

Upstream software projects move very fast

• Linux Kernel – New version every 3 months
• Kubernetes – New version every 3 months
• Salt – New version every 3-6 months
• Podman/skopeo/buildah – New version all the time
Anatomy of a Linux Distribution

• “How to condense thousands of packages from thousands of different people into something which people can use?”
• Must be coherent, consistent, and operational
• Linux Kernel + Toolchain + 1000’s of additional software packages
DevOps OS Development

code
plan
release
build
test
monitor
deploy
operate
Building Linux Better

https://openbuildservice.org/

- Open Build Service started in February 2006
- Used to build the openSUSE® & SUSE® distributions
- Can also build packages for other distributions (Fedora/Red Hat, Ubuntu, Debian, Arch, etc)
- Also used by ownCloud, Linux Foundation, VideoLAN (VLC), Dell, Cray, Intel and more.
- Builds containers, and publishes in a notarised OCI registry:
  - registry.opensuse.org
Testing Linux Better

https://open.qa/

• openQA started in November 2009
• Able to fully test Linux distributions from install to user applications
• Integral part of the openSUSE® Kubic, Tumbleweed & Leap process
• Used by SUSE® to test SUSE Linux Enterprise
• Also used by Red Hat to test Fedora
“Fancy, but I don’t want to wait for all that build & test nonsense”
Working with Upstream Projects

• Kubernetes 1.13 – Released Dec 3rd, shipped Dec 10th

• CRI-O 1.13 – Released Jan 7th, shipped Jan 10th

• Podman 1.0.0 – Released Jan 16th, shipped Jan 15th (oops)
Questions?

Thank you.

See you on https://kubic.opensuse.org/!