

Cloud Orchestration

What is a cloud?...

==>

- ▶ *distributing the load automatically*
- ▶ *(and selling more than what you have – thin provisioning everywhere...)*
- ▶ not necessarily virtualized – IaaS can also be bare-metal
- ▶ immediate delivery
- ▶ pay-as-you-use vs. unlimited
- ▶ QoS everywhere and **accounting**
- ▶ customers can use APIs

Warning: this lecture is about setting-up your own cloud

- ▶ no AWS
- ▶ no GCP
- ▶ no Azure
- ▶ you're the cloud administrator here, not the *luser*
- ▶ and it fits the sovereignty and privacy laws better, as long as your infrastructure is on the national territory

VMM Orchestrators

Different feature sets

1. just the UI
2. just an orchestrator
3. UI + orchestrator

VMM farm UIs

- ▶ VMware vSphere vCenter
- ▶ Citrix XenCenter
- ▶ RHEV == oVirt (just like RHEL == CentOS)
- ▶ HyperV
- ▶ Proxmox (KVM & LXC)

VMM orchestrators

- ▶ XCP-NG (XEN)
- ▶ Ganeti (XEN & KVM)
- ▶ Apache CloudStack (VMware, KVM, XenServer, XCP, Oracle VM, Hyper-V)
- ▶ OpenNebula (got orchestrator?)
- ▶ DanubeCloud (no orchestrator)
- ▶ OpenStack – *just like k8s... why so much pain?*

LAB // test Apache CloudStack

LAB // find the hot-migration scheduler algorithms in those code bases and discuss/compare techniques. On which hardware resources does it base its decision?

// Questions on VMM UIs and orchestrators?

Anything else than Docker in mind?...

==> Container engines' timeline

- ▶ chroot (1982)
- ▶ FreeBSD jails (2000)
- ▶ Solaris Zones / Containers (2004)
- ▶ AIX WPARs (2007)
- ▶ Virtuozzo / OpenVZ (resp. 2000 / 2005)
- ▶ Ubuntu LXC / LXD (2008)
- ▶ systemd-nspawn (2010)
- ▶ Docker (2013)
- ▶ runc/containerd (Jul 16, 2015)
- ▶ Singularity (2015)

What is a chroot?...

==> the thing you sometimes need for rescuing a system

There are also chroot-capable daemons (`named`, `unbound`, `postfix`)

Anyhow, which engine are good to choose?...

==> stick with the standards

- ▶ (ideally jails but that's for Free-and-DflyBSD)
- ▶ Docker for apps and micro-services
- ▶ LXC for systems

LAB // are there orchestrators for BSD jails?

LAB // study and retro-PoC the sysjail vuln on Net-or-OpenBSD.

Choose your image wisely

musl instead of glibc

- ▶ Alpine Linux
- ▶ also possible with Void Linux and other distros

Weird kinds of system-level virtualization

something in between...

- ▶ User-mode Linux (UML)
- ▶ DragonflyBSD vkernel

Unikernel

something entirely different...

- ▶ binary built-up on purpose
- ▶ libraries within it
- ▶ optimized run-time
- ▶ best-performance and best-security (no libc)

Container orchestrators

- ▶ Docker Swarm – *still-not-deprecated*
- ▶ K8S, Minikube, Play-with-k8s – *too much hype*
- ▶ OpenShift – just another layer on top of K8S
- ▶ Rancher

Better DIY...

- ▶ instances on every node + load-balance health checks
- ▶ instances on every node + one VIP per node (floating vs. CARP/VRRP)

Multi-orchestrators

- ▶ ManageIQ – orchestrate orchestrators (VMM & Containers)

LAB // is it able to schedule guest hot-migrations by itself?

LAB // is it able to schedule instance shuffling by itself?

Containers as guests

- ▶ `docker-machine`
- ▶ **Kata Containers**

Container orchestration history

it's a conspiracy

- ▶ Apache Mesos – *abandoned in favor of K8S*
- ▶ Cattle (Rancher's engine) – *idem*
- ▶ Docker Swarm (2014) – *announced dead but still alive*
- ▶ Kubernetes (7 June 2014) – *the only one remaining*

Redhat is pushing CRI-O (minimal Container Runtime Interface) for K8S

Swarm features

- ▶ overlay network
- ▶ storage=SSD
- ▶ easy to setup IP endpoints for the service to load-balance against

K8S features

- ▶ configuration replication with `etcd`
- ▶ Horizontal Pod Autoscaler (HPA)
- ▶ Ingress – built-in endpoint management and load-balancer
- ▶ Cloud Providers
- ▶ Dynamic Volume Provisioning

Horizontal Pod Autoscaler (HPA)

- ▶ *trashing* issue mitigated by delay setting (default 5 minutes)

Now let's try to plug our load-balancer against K8S

What settings do we have at hand?...

==>

ClusterIP (default)

NodePort

LoadBalancer

ExternalName

ClusterIP (default) - Exposes the Service on an internal IP in the cluster. This type makes the Service only reachable from within the cluster.

NodePort - Exposes the Service on the same port of each selected Node in the cluster using NAT. Makes a Service accessible from outside the cluster using \therefore . Superset of ClusterIP.

LoadBalancer - Creates an external load balancer in the current cloud (if supported) and assigns a fixed, external IP to the Service. Superset of NodePort.

ExternalName - Exposes the Service using an arbitrary name (specified by externalName in the spec) by returning a CNAME record with the name. No proxy is used. This type requires v1.7 or higher of kube-dns

Example parameters...

kube-apiserver: --pod-network-cidr

kube-proxy: --cluster-cidr

kube-controller-manager: --cluster-cidr
--api-advertise-address

// Questions on Docker farm orchestration?

Don't worry be happy

technology intelligence & research lab opportunities

LAB // PoC k8 for serving GPGPUs

LAB // PoC other means of serving GPGPUs

Using GPGPUs with Kubernetes

<https://blog.ubuntu.com/2018/12/10/using-gpgpus-with-kubernetes>

Schedule GPUs

<https://kubernetes.io/docs/tasks/manage-gpus/scheduling-gpus/>

General-purpose computing on graphics processing units

https://en.wikipedia.org/wiki/General-purpose_computing_on_graphics_processing_units

LAB // find out what kernel modules do what exactly for Docker

LAB // and *idem* for the few more modules K8S requires

LAB // search about privesc exploits and techniques against containers.
Discuss the supposed attack vectors.

LAB // K8S was vulnerable as of Dec 5 2018. Retro-PoC that.

Kubernetes Flaw is a “Huge Deal,” Lays Open Cloud Deployments
<https://threatpost.com/kubernetes-flaw-is-a-huge-deal-lays-open-cloud-deployments/139636/>

This is the end