

Container, Container-Plattformen, Basics und Konzepte

To make mistakes is human. To automatically deploy all mistakes to all Servers/Container is DevOps.

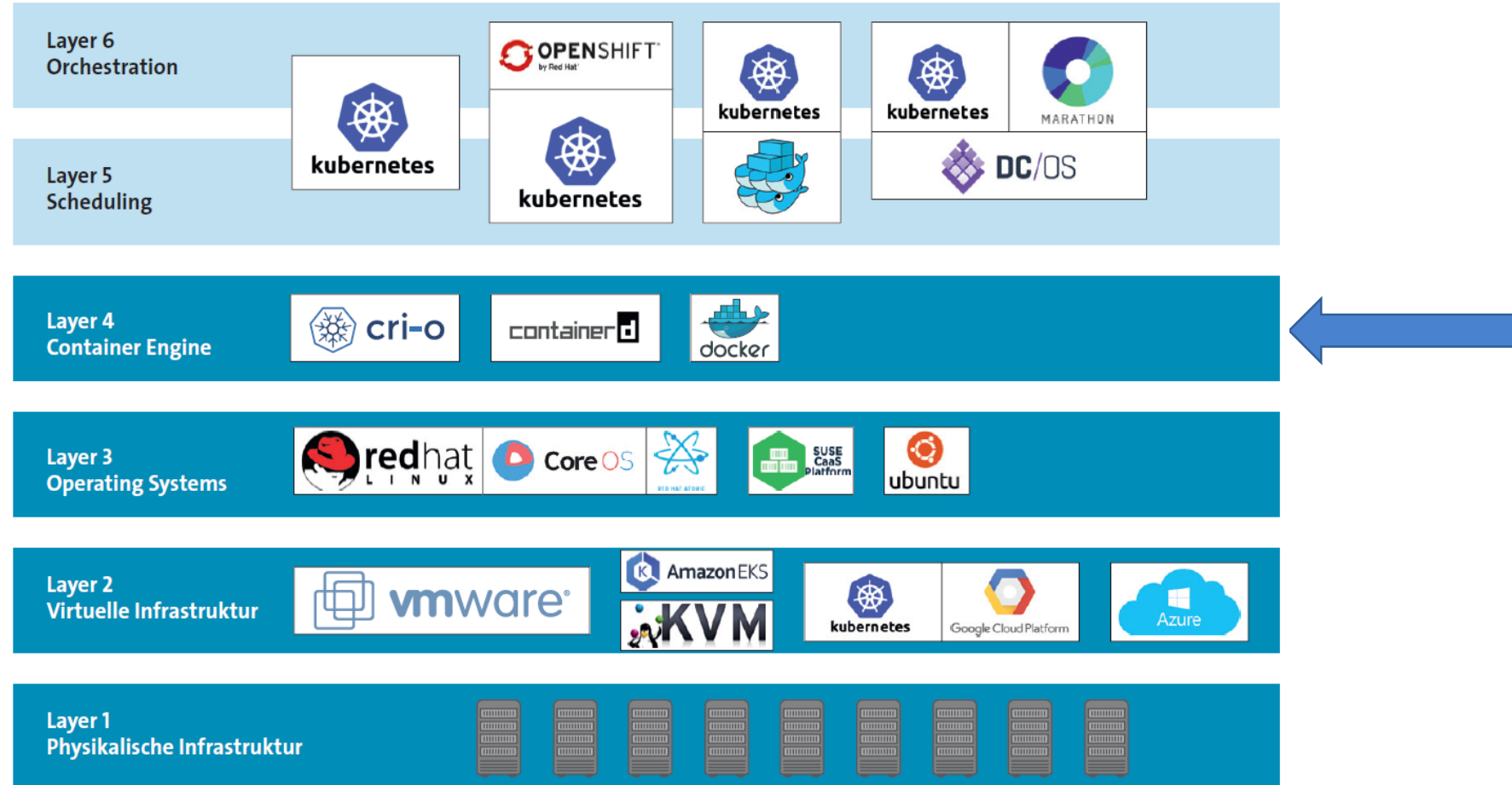
Lernziele

- ★ Sie haben einen Überblick über die Container Basics.
- ★ Sie haben ein ersten Überblick über das Container Ökosystem.

Zeitlicher Ablauf

- ★ Layers
- ★ Namespaces und Container-Konzepte
- ★ Container vs. Virtuelle Maschinen
- ★ Container-Lösungen
- ★ Cloud Native (Docker/Kubernetes) Landscape und Trail Map
- ★ Übung
- ★ Reflexion
- ★ Lernzielkontrolle

Die (vereinfachten) »Layer« der Container-Welt



Container - Geschichte

- ★ Container sind ein altes Konzept. Schon seit Jahrzehnten gibt es in UNIX-Systemen den Befehl chroot, der eine einfache Form der Dateisystem-Isolation bietet.
- ★ Seit 1998 gibt es in FreeBSD das Jail-Tool, welches das chroot-Sandboxing auf Prozesse erweitert.
- ★ **Solaris** Zones boten **2001** eine recht **vollständige Technologie** zum Containerisieren, aber diese war auf Solaris OS beschränkt.
- ★ Ebenfalls 2001 veröffentlichte Parallels Inc. (damals noch SWsoft) die kommerzielle Containertechnologie Virtuozzo für Linux, deren Kern später (im Jahr 2005) als Open Source unter dem Namen OpenVZ bereitgestellt wurde.
- ★ Dann startete **Google** die Entwicklung von CGroups für den Linux-Kernel und begann damit, seine **Infrastruktur in Container zu verlagern**.
- ★ Das Linux Containers Project (LXC) wurde 2008 initiiert, und in ihm wurden (unter anderem) CGroups, Kernel-Namensräume und die chroot-Technologie zusammengeführt, um eine vollständige Containerisierungslösung zu bieten.
- ★ **2013** lieferte **Docker** schließlich die fehlenden Teile für das Containerisierungspuzzle, und die Technologie begann, den Mainstream zu erreichen.

Container basieren auf Linux Konzepten (1)

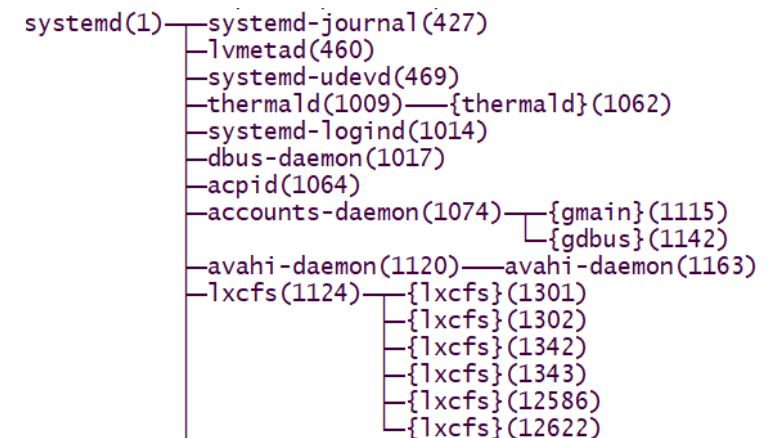
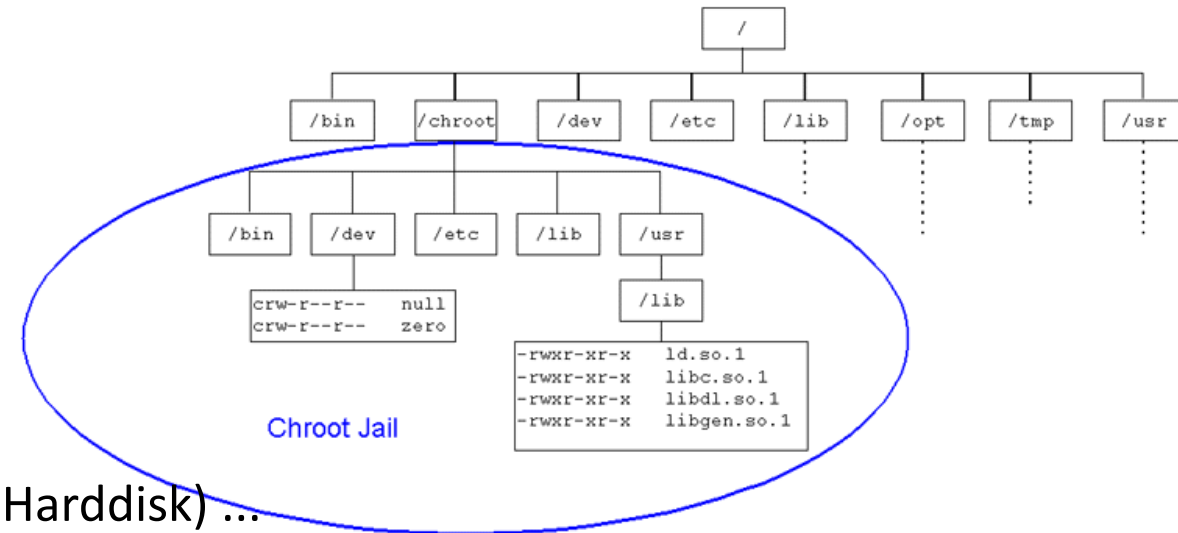
★ Dateisystem (ls -l /)

- Alles ist eine Datei z.B. /proc/cpuinfo, /dev/sdaX (Harddisk) ...
- Es gibt keine Laufwerke alles ist via / (root) erreichbar. Mehrere Roots sind möglich.
- Klein/Grossbuchstaben in Dateinamen werden unterschieden
- Es gibt keine Dateierweiterungen, z.B. .docx. Die ersten 4 Bytes bestimmen den Dateityp.
- Die Unterstützung mehrere Filesysteme (ext4, unionfs, aufs ...) ist die Regel und nicht die Ausnahme.

★ Prozesse (pstree -n -p)

- Prozesse sind hierarchisch angeordnet
- Der erste Prozess hat die ID 1.

★ Die Kommandozeile ist die bevorzugte Umgebung.



Container basieren auf Linux Konzepten (2)

★ Namespaces

- Ressourcen des Kernsystems (in diesem Falle also des Kernels) voneinander zu isolieren

★ Arten von Namespaces

- IPC -Interprozess-Kommunikation
- NET - Netzwerkressourcen
- PID -Prozess-IDs
- USER - Benutzer/Gruppen-Ids
- UTS - Systemidentifikation): Über diesen Namespace kann jeder Container einen eigenen Host- und Domännennamen erhalten.

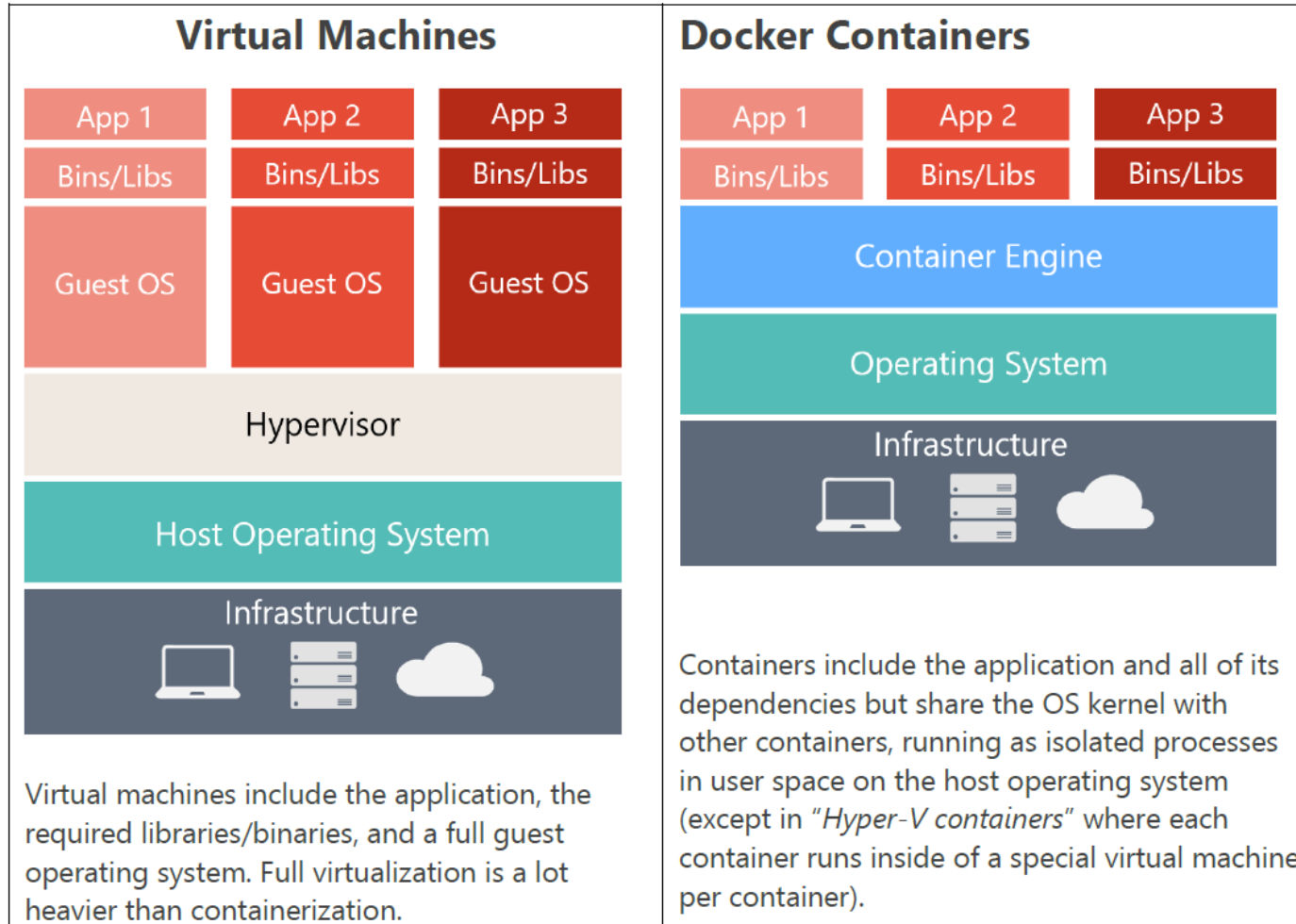
★ Weitere Informationen

- <https://docs.docker.com/engine/security/seccomp/>
- <https://medium.com/faun/the-missing-introduction-to-containerization-de1fbb73efc5>

Container basieren auf Linux Konzepten (Test)

- ★ Kommandozeile starten und in Virtuelle Maschine wechseln
 - `kubeps.bat` oder `kubesh.bat`
 - `vagrant ssh`
- ★ Anzeige Namespaces, aktueller Prozess
 - `ls -al /proc/$$/ns`
- ★ Wechsel in eigener Namespace mit eigenem Netzwerk und Prozess-IDs
 - `sudo unshare -n -p --fork --mount-proc /bin/bash`
- ★ Testen ob Netzwerk und Prozesse isoliert sind:
 - `ping google.com`
 - `ifconfig -a`
 - `ps tree -n -p`

Container vs. Virtuelle Maschinen



★ Virtuelle Maschinen

- Hypervisor
- Guest OS
- Applikation und Abhängigkeiten in VM
- Daten in VM gespeichert
- Schwergewichtig

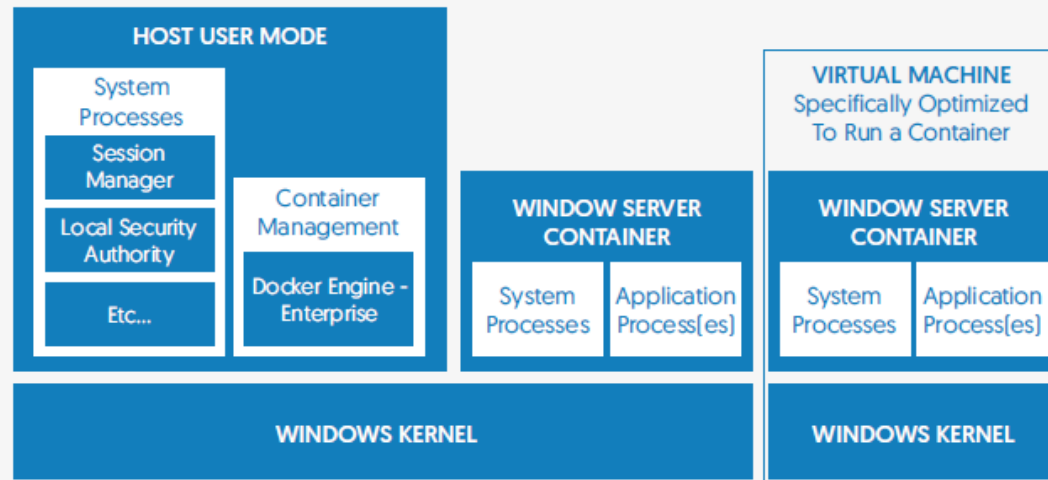
★ Container

- Container Engine
- Namespaces
- Applikation und Abhängigkeiten im Container
- Daten nach Beendigung des Containers verloren
- Leichtgewichtig

Container-Lösungen

- ★ **Lightweight Linux Container/VMs (LXC)** - eher als Application Virtualization Engine für eine dedizierte Applikation zu sehen.
- ★ **Docker** wurde im März 2013 von dotCloud veröffentlicht. Im Oktober 2013 benannte sich dotCloud in Docker Inc. um. **Im 2017 Marktanteil von 90 %.**
 - libcontainer, containerd und runC: Ab Version 1.0, keine Abhängigkeit mehr von LXC. Direkter Zugriff auf Namespaces etc.
 - **runC und containerD:** [runC](#) generische Abstraktionsschicht, die der *OCP(Open Container Plattform)*-Spezifikationen entspricht. **Containerd** fungiert als Standalone-Container-Daemon.
- ★ **CoreOS /Container Linux und Rocket:** CoreOS steht als abgespecktes, Linux-basiertes Container-Host-Betriebssystem.
- ★ **VMware Photon:** klassisches Hardware/VM/Container-Setup.
- ★ **Windows Server 2016:** Unterstützung Windows- (Hyper-V?) und Linux-Container via Linux Subsystem.
- ★ **Cloud:** siehe CNCF Cloud Native Landscape (nächste Folie).

Windows Container



- ★ **Windows Server-Container:** – Bieten Anwendungsisolation mithilfe einer Technologie zum Isolieren von Prozessen und Namespaces
- ★ **Hyper-V-Isolierung:** Erweitert die von Windows Server-Containern bereitgestellte Isolierung, indem jeder Container in einem hochgradig optimierten virtuellen Computer ausgeführt wird.

★ Quelle: <https://docs.microsoft.com/de-ch/virtualization/windowscontainers/about/index#windows-container-types>

CNCF Cloud Native Landscape

Sie sehen 553 Karten mit insgesamt 1.168.992 Sternen, einer Marktkapitalisierung von 6,93 T und einer Finanzierung von 19,3 B \$.



Cloud Native Trail Map (dt.: Wanderkarte)

CLOUD NATIVE COMPUTING FOUNDATION

CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape Landscape has a larger number of options. The Cloud Native Trail Map is a recommended process for leveraging open source cloud-native technology as it guides you to choose a vendor-supported offering or do it yourself, and everything after that is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification
Consider taking either from CNCF or to take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer.

B. Consulting Help
If you want assistance with Kubernetes or the surrounding ecosystem, consider looking at a Kubernetes Certified Service Provider.

C. Join CNCF's End User Community
For companies that don't offer cloud-native services externally.

WHAT IS CLOUD NATIVE?
Cloud-native technologies, such as containers, microservices, or serverless, enable organizations to build and deploy scalable, resilient applications and services in dynamic, distributed environments. By taking into account these characteristics, such systems are designed to be resilient, elastic, and loosely coupled, a managed platform and resilient infrastructure (IaaS) are essential. By taking into account these characteristics, such systems are designed to be resilient, elastic, and loosely coupled, a managed platform and resilient infrastructure (IaaS) are essential. By taking into account these characteristics, such systems are designed to be resilient, elastic, and loosely coupled, a managed platform and resilient infrastructure (IaaS) are essential.

1. CONTAINERIZATION
• Commonly done with Docker containers
• Any software application and dependencies (even PaaS) can be containerized
• Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

2. CI/CD
• Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
• Setup automated rollbacks, roll backs and testing

3. ORCHESTRATION & APPLICATION DEFINITION
• Kubernetes is the market leading orchestration solution
• You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer for ease of use
• Helm Charts help you define, install, and upgrade even the most complex Kubernetes application

4. OBSERVABILITY & ANALYSIS
• Pick solutions for monitoring, logging and tracing
• Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for tracing
• For tracing, look for an OpenTracing compatible implementation like Jaeger

5. SERVICE MESH AND DISCOVERY
• CoreDNS is a fast and flexible tool that is used for service discovery
• Envoy and Linkerd each enable service mesh architectures
• They offer health checking, routing, and load balancing

6. NETWORKING
To enable more flexible networking, use a CNCF-compliant network project like Calico, Flannel, or Weave Net.

7. DISTRIBUTED DATABASE
When you need more resiliency and scalability than you can get from a single database, a distributed database is a good option. For running MySQL at scale through sharding.

8. MESSAGING
When you need higher performance than JSON-RPC, consider using gRPC. NATS is publish/subscribe message oriented middleware.

9. CONTAINER RUNTIME
You can use alternative container runtimes. The most common, all of which are OCI compliant, are containerd, rkt and CRIO.

10. SOFTWARE DISTRIBUTION
If you need to co-secure software distribution, evaluate Notary, an implementation of The Update Framework.

Incubation Status:
CNCF Incubating (blue icon), CNCF Graduated (green icon), CNCF Incubating (red icon)

Logos: Kubernetes, Helm, Prometheus, Fluentd, Jaeger, CoreDNS, Envoy, Linkerd, CoreDNS, ViteSS, NATS, gRPC, Notary, CRIO, rkt.

QR Code: <https://github.com/cncf/landscape/blob/master/README.md>

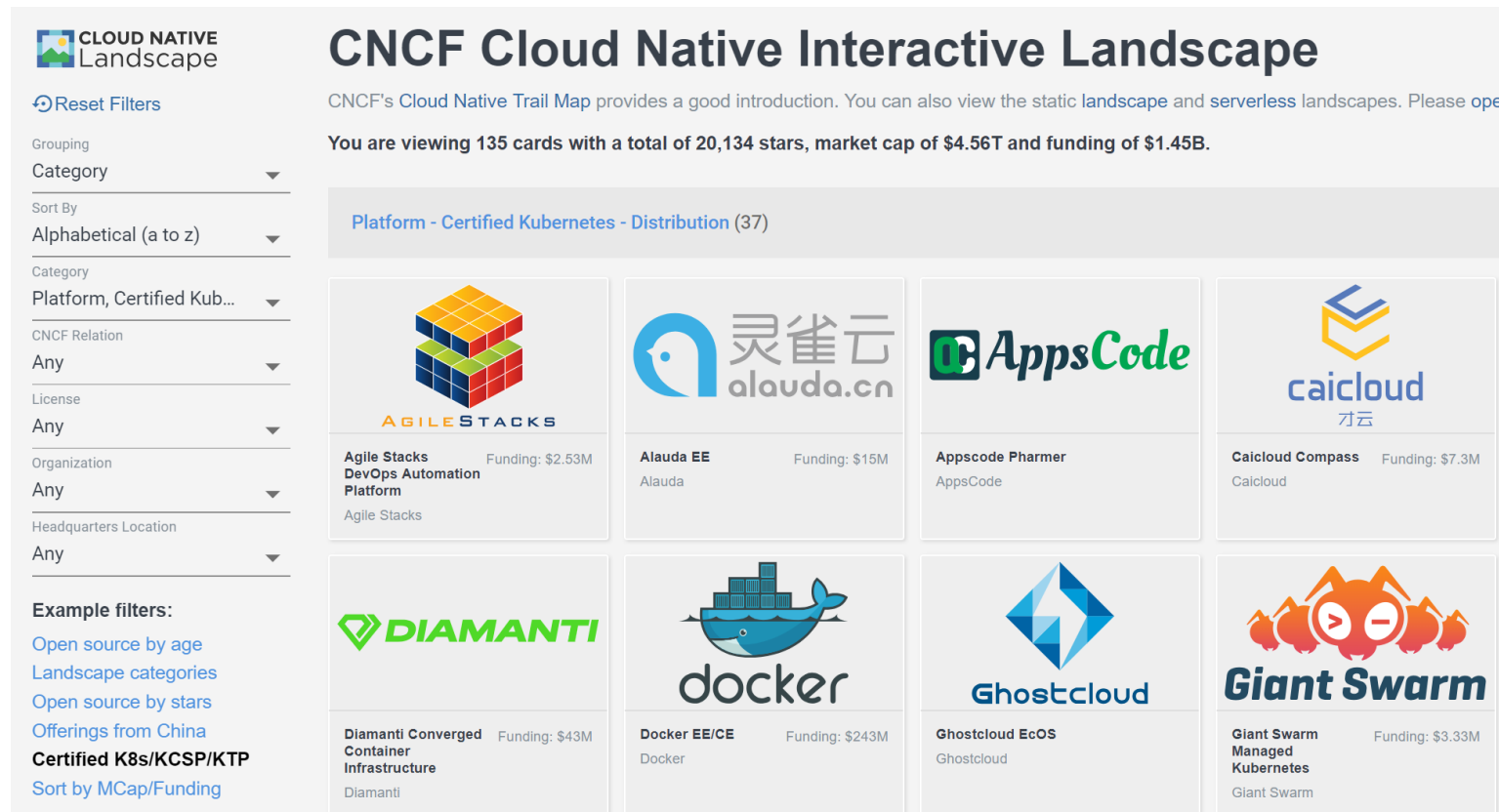
l.cncf.io
v20180604

★ Die Cloud-Native-Trail-Karte bietet einen Überblick für Unternehmen, die ihre Cloud-native (Docker/Kubernetes) Reise starten.

1. Containerization (Docker)
2. CI/CD (dt.: Fortlaufende Integration, Verteilung)
3. Orchestration (Kubernetes)
4. ...
8. Messaging (z.B. für IoT, Big Data, ML)

Übung: CNCF Cloud Native Landscape

- ★ Öffnet die [CNCF Landscape](#) und sucht alle Kubernetes zertifizierten Produkte:











CNCF Cloud Native Interactive Landscape

CNCF's [Cloud Native Trail Map](#) provides a good introduction. You can also view the static [landscape](#) and [serverless](#) landscapes. Please [open](#)

You are viewing 135 cards with a total of 20,134 stars, market cap of \$4.56T and funding of \$1.45B.

Platform - Certified Kubernetes - Distribution (37)

Logo	Product Name	Funding
	Agile Stacks DevOps Automation Platform	Funding: \$2.53M
	Alauda EE	Funding: \$15M
	Appcode Pharmer	AppCode
	Caicloud Compass	Funding: \$7.3M
	Diamanti Converged Container Infrastructure	Funding: \$43M
	Docker EE/CE	Funding: \$243M
	Ghostcloud EcOS	Ghostcloud
	Giant Swarm Managed Kubernetes	Funding: \$3.33M



Reflexion

- ★ Container sind ein altes Konzept, es basiert vereinfacht auf Linux Namespaces.
- ★ Das Container Ökosystem umfasst alle wichtigen Technologiefirmen und wird durch die Open-Source-Software-Stiftung «Cloud Native Computing Foundation» (CNCF) gefördert.

? Lernzielkontrolle

- ★ Sie haben einen Überblick über die Container Basics.
- ★ Sie haben ein ersten Überblick über das Container Ökosystem.