What is Cloud Native and Why You Should Care

Dan Kohn
Executive Director, CNCF
- 4,550 unique authors
- 23,347 unique issue commenters
- 2nd fastest development velocity (behind Linux)

See devstats.cncf.io
Kubernetes in Search Trends

Google Trends

Kubernetes      OpenStack      Mesos      Docker Swarm      Cloud Foundry

July-16    Jan-17    Jul-17    Jan-18    Jul-18

WeChat

Kubernetes      OpenStack

July-18

© 2018 Cloud Native Computing Foundation
61 Companies in CNCF’s End User Community

Adidas
ATLASIAN
Bloomberg
box
CapitalOne
COMCAST
CRUISE
DENSO
DiDi
eBay
FORM3
GitHub
Goldman Sachs
indeed
Intuit
JD.COM
J.P.Morgan
Kuelap
Layer
Morgan Stanley
NAIC
National Association of Insurance Commissioners
Nasdaq
NCsoft
The New York Times
NIPR
Oklark
Pinterest
PUSHER
Reddit
ricardo.ch
salesforce
SAP Concur
Shopify
Showmax
Spotify
Spreedfast
Squarespace
Steelhouse
Stix
Textkernel
ThredUp
Ticketmaster
Twilio
twitter
Two Sigma
Werkspot
Wikimedia Foundation
Woorank
workday
WP Engine
Yahoo
zalando
Zendesk

Plus 7 non-public members
61 Kubernetes Certified Service Providers
71 Certified Kubernetes Partners
KubeCon + CloudNativeCon Attendees

- San Francisco (Nov 2015)
- London (Mar 2016)
- Seattle (Nov 2016)
- Berlin (Mar 2017)
- Austin (Dec 2017)
- Copenhagen (May 2017)
ORCHESTRATION.
CONTAINERIZATION.
MICROSERVICES.
Cloud Native Definition (1 of 3)

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as **public, private, and hybrid clouds**. Containers, **service meshes**, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

[cncf.io/d](https://cncf.io/d)
Cloud Native Definition (2 of 3)

These techniques enable *loosely coupled systems* that are resilient, manageable, and observable. Combined with *robust automation*, they allow engineers to make *high-impact changes* frequently and predictably with minimal toil.

cncf.io/d
Cloud Native Definition (3 of 3)

The Cloud Native Computing Foundation seeks to **drive adoption** of this paradigm by fostering and sustaining an **ecosystem of open source, vendor-neutral projects**. We democratize **state-of-the-art patterns** to make these innovations accessible for everyone.

[cncf.io/d](cncf.io/d)
Network Architecture 1.0
Network Architecture Evolution 1.0

1.0: Separate physical boxes for each component (e.g., routers, switches, firewalls)
Network Architecture 2.0
Network Architecture Evolution 2.0

2.0: Physical boxes converted to virtual machines called Virtual Network Functions (VNFs) running on VMware or OpenStack
Network Architecture 3.0
(hardware is the same as 2.0)
Network Architecture Evolution 3.0

3.0: Cloud-native Network Functions (CNFs) run on Kubernetes on public, private, or hybrid clouds
Evolving from VNFs to CNFs (Past)

- VNFs
- ONAP Orchestrator
- OpenStack or VMware
- Bare Metal
- Azure or Rackspace
Evolving from VNFs to CNFs (Present)

Present

VNFs
OpenStack
Bare Metal

CNFs
Kubernetes

ONAP Orchestrator

Any Cloud
Evolving from VNFs to CNFs (Future)

Future

- CNFs
- VNFs
  - KubeVirt/Virtlet
- ONAP Orchestrator

Other layers:
- Kubernetes
- Bare Metal
- Any Cloud
Three Major Benefits

1. Cost savings
Three Major Benefits

2. Improved resiliency (to failures of individual CNFs, machines, and even data centers)
Three Major Benefits

3. Higher development velocity
The challenge of transitioning VNFs to CNFs

• Moving network functionality from *physical* hardware to encapsulating the software in a *virtual* machine (P2V) is generally easier than *containerizing* the software (P2C or V2C)
• Many VNFs rely on kernel hacks or otherwise do not restrict themselves to just the *Linux userspace ABI*
Cloud Native Computing Foundation

• Non-profit, part of the Linux Foundation; founded Dec 2015

Graduated
- kubernetes
  Orchestration
- Prometheus
  Monitoring
- Software Update Spec
- Security
- Storage
- Service Discovery
- Messaging
- Package Management
- Storage
- Security
- Identity Spec
- Tooling
- Registry
- Metrics Spec
- Distributed K/V
- Monitoring

Incubating
- OPENTRACING
  Distributed Tracing API
- fluentd
  Logging
- GRPC
  Remote Procedure Call
- containerd
  Container Runtime
- rkt
  Container Runtime
- CNI
  Networking API
- envoy
  Service Mesh
- JAEGER
  Distributed Tracing
- spiffe
  Identity Spec
- SPIRE
  Identity
- Open Policy Agent
  Policy
- cloudevents
  Serverless

Sandbox
- Software Update Spec
- Container Runtime
- Networking API
- Service Mesh
- Package Management
- Distributed Tracing API
- Container Runtime
- Messaging
- Distributed Tracing
- Container Runtime
- Metrics Spec
- Distributed K/V
- Monitoring

• Platinum members:
Cloud Native Trail Map

Trail Map: cncf.io

1. CONTAINERIZATION
• Commonly done with Docker containers
• Any size applications and dependencies (even POP-11 code running on an emulator) can be containerized
• Over time, you should explore containerizing stateful services

2. CI/CD
• Multiple 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 rollback, rollbacks and testing

3. ORCHESTRATION & APPLICATION DEFINITION
• Kubernetes is the market-leading orchestration solution
• You should select a Certified Kubernetes Distribution, Hosted Platform, or installer such as:
• Stack Overflow helps you install and upgrade even the most complex Kubernetes applications

4. OBSERVABILITY & ANALYSIS
• Prom solutions for monitoring, logging and tracing
• Grafana helps visualize your data
• VictorOps helps you with monitoring
• Fluentd for logging and Logstash for Tracing
• For tracing, look for an open-tracing-compatible implementation like Jaeger

5. SERVICE MESH AND DISCOVERY
• CoreDNS is a fast and flexible tool that is useful for service discovery
• envoy and Linkerd are multi-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 Verrina Net

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

8. MESSAGING
When you need high performance than JSON-RPC, consider using gRPC-Web as 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 OMI

10. SOFTWARE DISTRIBUTION
If you are not familiar with software distribution, evaluate Notary, an implementation of The Update Framework.
PLEASE TRY THE INTERACTIVE LANDSCAPE NOW:

l.cncf.io
KubeCon + CloudNativeCon

- China
  - **Shanghai**: November 13-15, 2018
  - Sponsorships **open**
- North America
  - **Seattle**: December 10-13, 2018
  - Sponsorships **open**
- 2019
  - **Barcelona**: May 20-23, 2019
  - **Shanghai**: June 26-28, 2019
  - **San Diego**: November 18-21, 2019