HOW GOOD IS OUR CODE?

Dan Kohn
Executive Director, CNCF
Cloud Native Computing Foundation

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

**Graduated**
- Prometheus Monitoring
- raft
- fluentd
- GRPC
- containerd
- rkt
- CNI
- envoy
- Rook
- SPIRE
- identity

**Incubating**
- JAEGER
- OpenTracing
-可观测性
- Vittess
- CoreDNS
- Linkerd
- Helm
- SPIFE
- cloudevents
- Software Update Spec
- Service Discovery
- Messaging
- Service Mesh
- Package Management
- Open Policy Agent
- Serverless
- Tooling

**Sandbox**
- Alibaba Cloud
- AWS
- Azure
- Cisco
- Dell Technologies
- Docker
- Fujitsu
- Google Cloud
- IBM Cloud
- Intel
- JD.COM
- Mesosphere
- Oracle
- Pivotal
- Red Hat
- Samsung
- SAP
- VMware

- Platinum members:
TODAY THE LINUX FOUNDATION IS MUCH MORE THAN LINUX

Security
We are helping global privacy and security through a program to encrypt the entire internet.

Networking
We are creating ecosystems around networking to improve agility in the evolving software-defined datacenter.

Cloud
We are creating a portability layer for the cloud, driving de facto standards and developing the orchestration layer for all clouds.

Automotive
We are creating the platform for infotainment in the auto industry that can be expanded into instrument clusters and telematics systems.

Blockchain
We are creating a permanent, secure distributed ledger that makes it easier to create cost-efficient, decentralized business networks.

Web
We are providing the application development framework for next generation web, mobile, serverless, and IoT applications.

Let's Encrypt

We are regularly adding projects; for the most up-to-date listing of all projects visit tlfprojects.org
KubeCon + CloudNativeCon

- **China**
  - **Shanghai**: November 14-15, 2018
  - Sponsorships **open**
- **North America**
  - **Seattle**: December 11 - 13, 2018
  - Sponsorships **open**
- **Europe**
  - **Barcelona**: May 21 - 23, 2019
KubeCon + CloudNativeCon Attendees

- San Francisco (Nov 2015)
- London (Mar 2016)
- Seattle (Nov 2016)
- Berlin (Mar 2017)
- Austin (Dec 2017)
- Copenhagen (May 2017)
HOW GOOD IS OUR CODE?
ORCHESTRATION.
CONTAINERIZATION.
MICROSERVICES.
“I haven’t tasted tap water in a long time…..” Doug Evans said. “You have to be agile and tactile, and be available to experiment. Literally, you have to carry bottles of water through the dark.”
Our new software consultancy produces what we call "raw code", guaranteed NOT to have passed through CI or any kind of onerous "testing". The result is a palpably richer and more authentic software experience.

7:08 PM - 4 Jan 2018
OUR SOFTWARE IS NOT AS GOOD AS SQLITE

- Developed mainly by one highly-regarded developer, Richard Hipp
- 100% branch test coverage
- Millions of test cases
- 1,000 times as much test code as product code

https://www.sqlite.org/testing.html
AMERICAN FUZZY LOP

Software fuzzer built by Michał Zalewski that uses genetic algorithms to find bugs
SQLite still has bugs!

When Zalewski ran American Fuzzy Lop against SQLite, he found 22 bugs (!!!) in 30 minutes of work.

Note that SQLite quickly fixed all of the bugs and incorporated AFL into their release process.

But our code is not as good as SQLite’s!
HOW BIG IS OUR APP?
LINUX

17 M SLOCs
DEPLOYMENT PLATFORM

KUBERNETES
35 M SLOCs
3RD PARTY LIBRARIES

3RD PARTY LIBRARIES
2.5 M SLOCs
OUR CODE IS ONLY 40 K SLOCS
OUR APPLICATION SOFTWARE STACK

- Linux: 51.9%
- Landscape: 18.5%
- Node_modules: 25.7%
- Node.js: 3.8%
- Kubernetes:
OUR CODE IS <0.1% OF OUR SOFTWARE STACK
ALL OF THIS CODE IS VULNERABLE

OUR CODE
40 K SLOCs

3RD PARTY LIBRARIES
2.5 M SLOCs

NODE.JS
12.3 M SLOCs

KUBERNETES
35 M SLOCs

LINUX
17 M SLOCs
SPECTRE & MELTDOWN IN LINUX

- **OURS CODE**: 40 K SLOCs
- **3rd PARTY LIBRARIES**: 2.5 M SLOCs
- **NODE.JS**: 12.3 M SLOCs
- **KUBERNETES**: 35 M SLOCs
- **LINUX**: 17 M SLOCs
SPECTRE & MELTDOWN IN LINUX

- **OUR CODE**
  - 40 K SLOCs

- **3RD PARTY LIBRARIES**
  - 2.5 M SLOCs

- **NODE.JS**
  - 12.3 M SLOCs

- **KUBERNETES**
  - 35 M SLOCs

- **LINUX**
  - 17 M SLOCs

- **Heartbleed**
- **DivideConcept**
SPECTRE & MELTDOWN IN LINUX

- OUR CODE
  - 40 K SLOCs

- 3RD PARTY LIBRARIES
  - 2.5 M SLOCs

- NODE.JS
  - 12.3 M SLOCs

- KUBERNETES
  - 35 M SLOCs

- LINUX
  - 17 M SLOCs

- subPath

- Heartbleed

- DivideConcept
SPECTRE & MELTDOWN IN LINUX

- **OUR CODE**
  - 40 K SLOCs
- **3RD PARTY LIBRARIES**
  - 2.5 M SLOCs
- **NODE.JS**
  - 12.3 M SLOCs
- **KUBERNETES**
  - 35 M SLOCs
- **LINUX**
  - 17 M SLOCs
- **Heartbleed**
- **subPath**
- **Spectre and Meltdown**
- **DivideConcept**
The power of **open source** is the ability to leverage **thousands** of other **developers** that are **finding bugs** and **making fixes** to the software we depend on.
But a software patch does not help until we have deployed it into production.
How can we have the **confidence** that that deployment **won’t break** anything?
The power of open source is the ability to leverage thousands of other developers that are finding bugs and making fixes to the software we depend on.

But a software patch does not help until we have deployed it into production.

How can we have the confidence that that deployment won’t break anything?

The Answer is CONTINUOUS INTEGRATION (CI)
WHAT KIND OF TESTS SHOULD CI RUN?

Unit testing of individual portions of our source code in isolation?
WHAT KIND OF TESTS SHOULD CI RUN?

Unit testing of individual portions of our source code in isolation?

Integration testing, where we work with external systems like a database?
WHAT KIND OF TESTS SHOULD CI RUN?

- Unit testing, testing individual portions of our source code in isolation?
- Integration testing, where we work with external systems like a database?
- Regression testing, where we add a test after a failure
WHAT KIND OF TESTS SHOULD CI RUN?

Unit, testing individual portions of our source code in isolation?

Integration testing, where we work with external systems like a database?

Regression testing, where we add a test after a failure?

Smoke testing, also known as build verification testing?
WHAT KIND OF TESTS SHOULD CI RUN?

- Unit testing, testing individual portions of our source code in isolation?
- Integration testing, where we work with external systems like a database?
- Regression testing, where we add a test after a failure?
- Smoke testing, also known as build verification testing?

All of the above.
HOW GOOD IS OUR CODE?
HOW GOOD IS OUR CODE?

NOT GOOD ENOUGH
We need to build in the systems and processes that enable us to continuously improve it
ORCHESTRATION.
CONTAINERIZATION.
MICROSERVICES.
If you don't have a CI system capable of building your application, then Kubernetes is the least of your problems. Focus on CI first.

12:38 PM - 10 Jun 2016
Continuous integration (CI) just means constant testing.

But what is testing?
We have a hypothesis of what we believe our code should do, but we don’t know for sure until we test against objective reality.
Karl Popper defined science as **BEING TESTABLE AND FALSIFIABLE**
What do Continuous Integration, Science, and Entrepreneurship all have in common?
What do Continuous Integration, Science, and Entrepreneurship all have in common?

They each require comparing an idealized conception to the often brutal truth of objective reality.
HOW DOES CONTINUOUS INTEGRATION FIT INTO THE CLOUD NATIVE JOURNEY?
Cloud Native Trail Map

Trail Map: l.cncf.io
Continuous Integration / Continuous Delivery (CI/CD)
PLEASE TRY THE INTERACTIVE LANDSCAPE NOW:

l.cncof.io