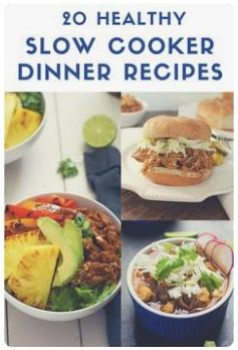
Invisible OS and Platform Upgrades

Adam McKenna, Site Reliability Engineer, Pinterest
@deathtocss
/in/admckenna
Pronouns: He, his

Healthy | 4 Ingredients | Chicken | Mornings | Crockpot Meals | Cheap | Vegetarian | Beef | Simple | For Kids | Families | Hamburger | Dump | For Two | Dinners

20 Cozy Dump Dinners That Basically Cook...



The Inspired Home

20 Healthy Slow Cooker Dinner Recipes

Promoted by The Inspired Home

Crock Pot Lemon Garlic Butter Chicken

Slow Cooker Beef Chili Recipe

Kraft Natural Cheese

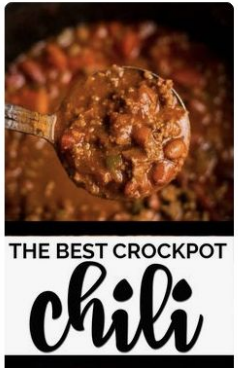
Deluxe Mini Mac & Cheese

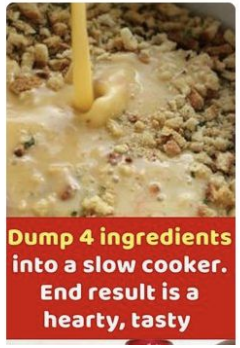
Promoted by Kraft Natural Cheese

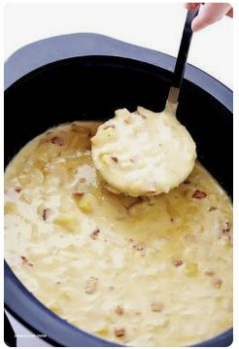
Philadelphia Cream Cheese

30-Minute Spinach Artichoke Fettuccine Alfredo

Slow Cooker Buttery Bacon Green Beans







★★★★★116

Slow Cooker Potato Soup

4.5 hours

Our mission

To bring everyone the inspiration to create a life they love.

Pinterest Scale

- 30 Million monthly active users
- ~80,000 hosts under management
- Dockerized micro-service architecture

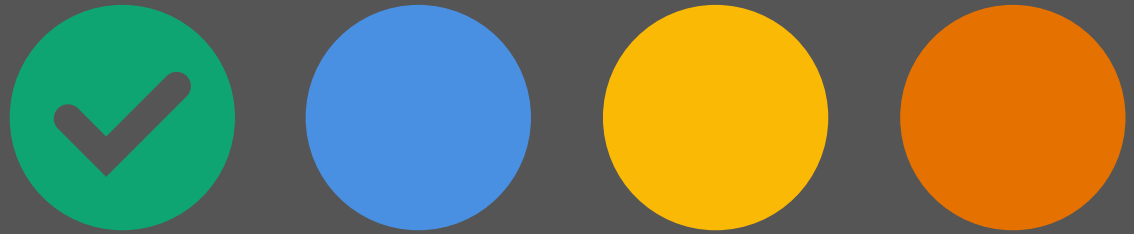
Core SRE @ Pinterest

- Overall Pinterest Uptime
- Internal Services

About Me:

- Linux user since 1993, Sysadmin since 1998
- DevOps practitioner since early 2000's

About this talk



Target Audience

- DevOps/SRE/Sysadmin/Release Engineer
- Don’t like having prod issues
- Looking for ways to improve reliability of deploys

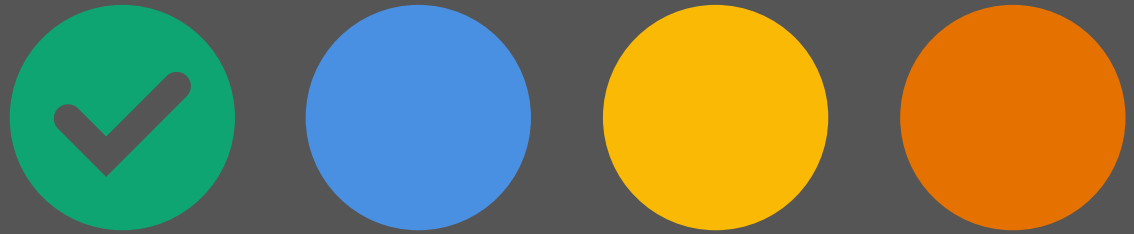
Key Takeaways

- What Canary Analysis is and isn’t
- What it’s useful for
- Practical Considerations
- Example System / Components

Prerequisites

- Your org already uses CI/CD for deploys
- Critical service metrics stored in a time series DB
- Organizational buy-in

Definitions



“Enabling Invisible Infrastructure Upgrades with Automated Canary Analysis”

“Enabling Invisible Infrastructure Upgrades with Automated Canary Analysis”

Invisible

Does not require a significant time investment by the service owner(s)

Infrastructure

“the underlying foundation or basic framework (as of a system or organization)” - Merriam-Webster

Infrastructure

- **Physical Hardware**
  - Cloud provider Instance Type/CPU Generation
  - Storage Technology
  - Networking stack
- **Operating System**
  - Kernel
  - User space
- **Language**
  - Compilers (C++/Go/etc.)
  - Runtime (JVM/Python Interpreter/etc.)
  - Dependencies (versions of vendored libraries/modules/etc)

Canary Analysis

Image credits: Michael Sonnabend (flickr)
https://pxhere.com/en/photo/1412043

Infrastructure has an expiration date

Time's up for the Trusty Tahr

Ubuntu 14.04 Reaches End of Life on April 30

End of Public Updates of Java SE 8

Java SE 8 has gone through the End of Public Updates process for legacy releases. Oracle will continue to provide free public updates and auto updates of Java SE 8 from Oracle at java.com, until at least the end of December 2020 for Personal, Development and other Users. Developers can find Oracle Java SE updates, including Oracle Java SE 8, 11 and current releases, free for development on OTN. As of the April 16, 2019 quarterly critical patch update, Oracle Customers should access updates to Java SE 8 for commercial use from Oracle through My Oracle Support and via auto update where applicable (Visit My.Oracle Support Note 1439822.1 - All Java SE Downloads on MOS – Requires Support Login).

Python 2.7

0	2	2
Years	Months	Days

Enable Guido Mode Huh?

How to prepare for Windows 7 End of Life

By Matt Hanson March 21, 2019 How To

The end is nigh for Windows 7 – here's what you need to know

Infrastructure has an expiration date

- **OS/Container Runtimes**
  - Ubuntu
    - 12.04, 14.04 EOL
  - Docker
    - Quarterly Updates
  - Windows
    - Windows 7 EOL (Pinterest not affected)

- **Language Runtimes**
  - Python
    - 2.x to 3.x
  - Java
    - Oracle to OpenJDK
    - Java version updates
  - Go
  - C++

Upgrading is not optional

The business needs to:

- **Meet industry compliance requirements**

- **Retain support**
  - Security bug fixes

- **Access to new features**
  - Latest hardware support (CPU, GPU, etc.)
  - Performance improvements
  - New storage technologies

Developers want

- Access to new packages / features
- Platform Efficiency

```
class java::params {
  case $::osfamily {
    'Debian': {
        case $::lsbdistrelease {
            '18.04': {
                $java_runtime_packa
                    'oracle-java8-j
                }
                $java_runtime_defau
            }
            '16.04': {
                $java_runtime_packa
```

But upgrading is hard!

- **Complexity**
  - Hundreds of microservices across tens of thousands of hosts
- **Service owners**
  - Don’t like downtime.
  - Migration work is generally not a preferred task

Canary Analysis

A tool that helps us automate and normalize the most mundane migration tasks

- **Automate**
  - looking at charts, comparing metrics
- **Normalize**
  - The same process can be applied to many similar systems.

What Canary Analysis isn't

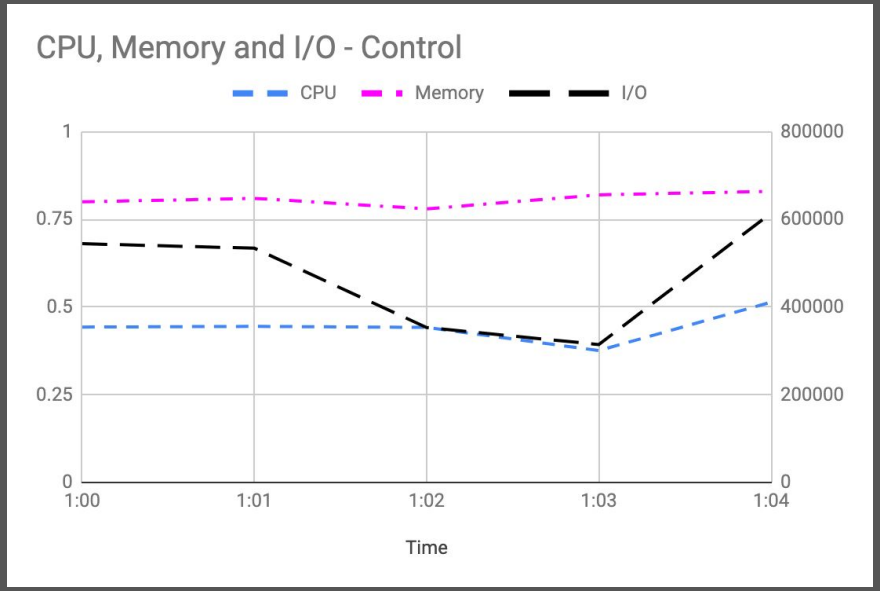
- **A replacement for traditional testing patterns:**
  - Unit and integration tests
  - Service Health Checks
- **Mysterious or Magical**
- **Off the Shelf**
- **Instantaneous**

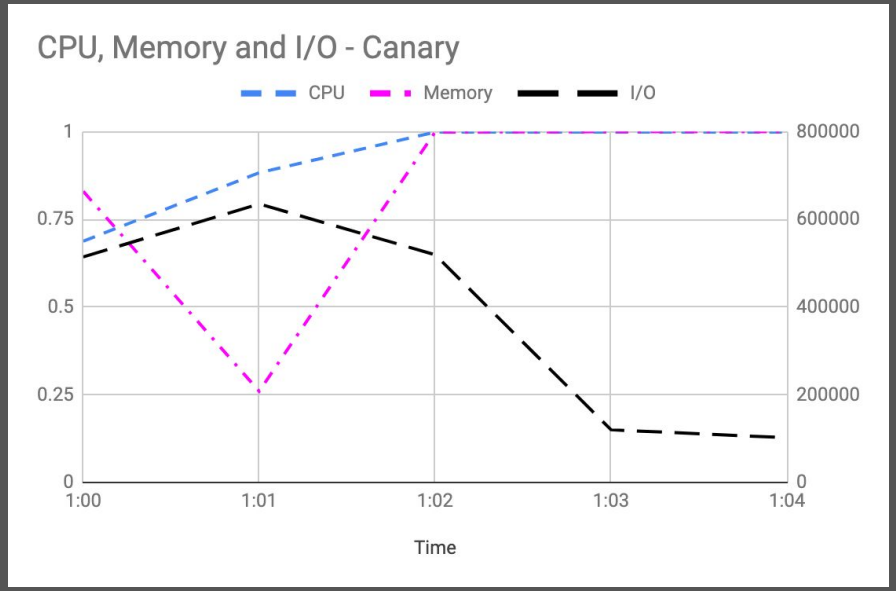
Typical migration ‘toil’ pattern

1. New Image is released
2. Service owner updates Image ID in deploy system
3. Operator triggers rolling cluster upgrade
4. Humans watch charts and cross fingers
5. Did things go OK? Good!
6. Is the site down/degraded? Bad! Rollback!
   Outage?

“Looks good to me!”

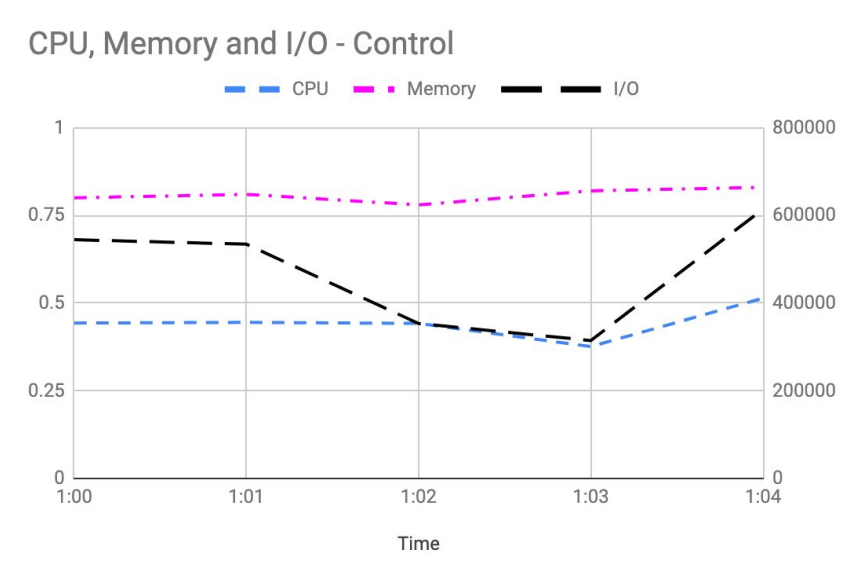
Can you make a Go / No-Go decision based on these two charts?

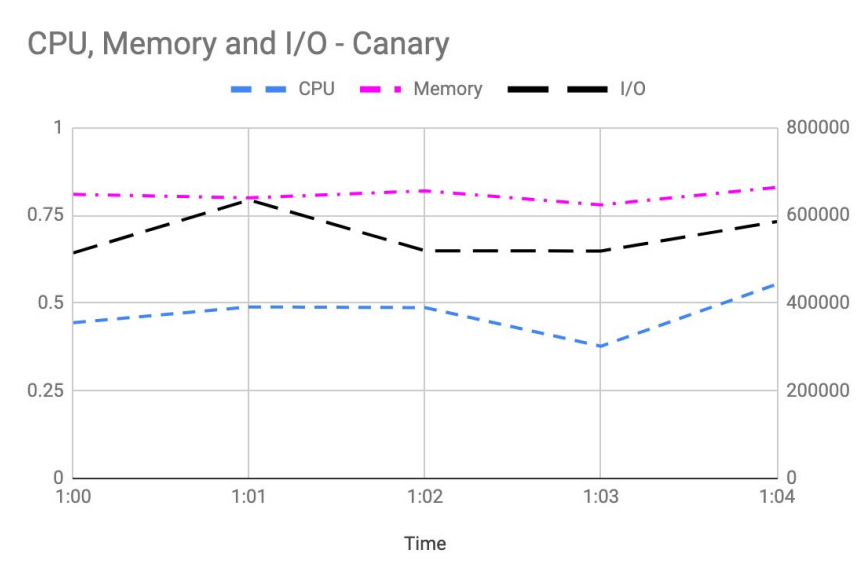




Sometimes humans disagree

How about these?





The human factor

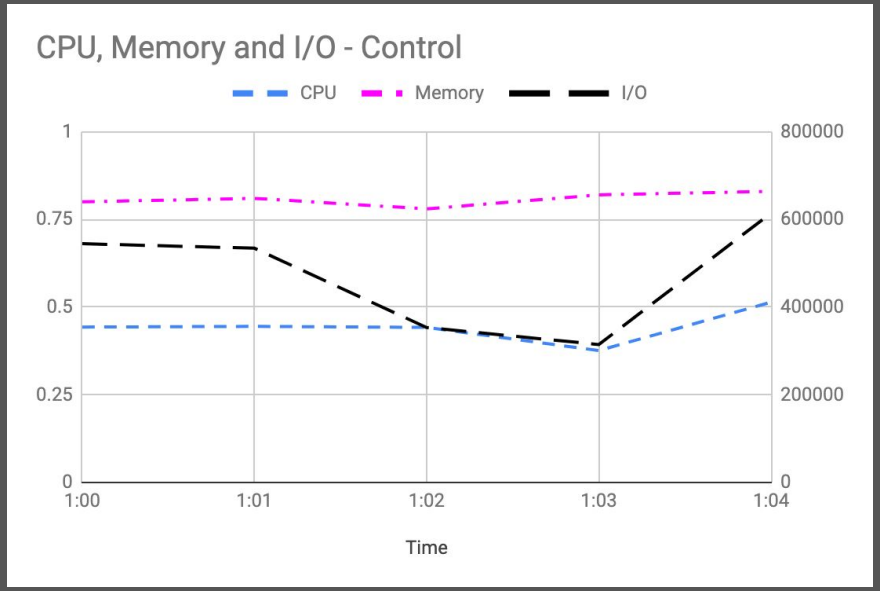
What might these Service Owners think?

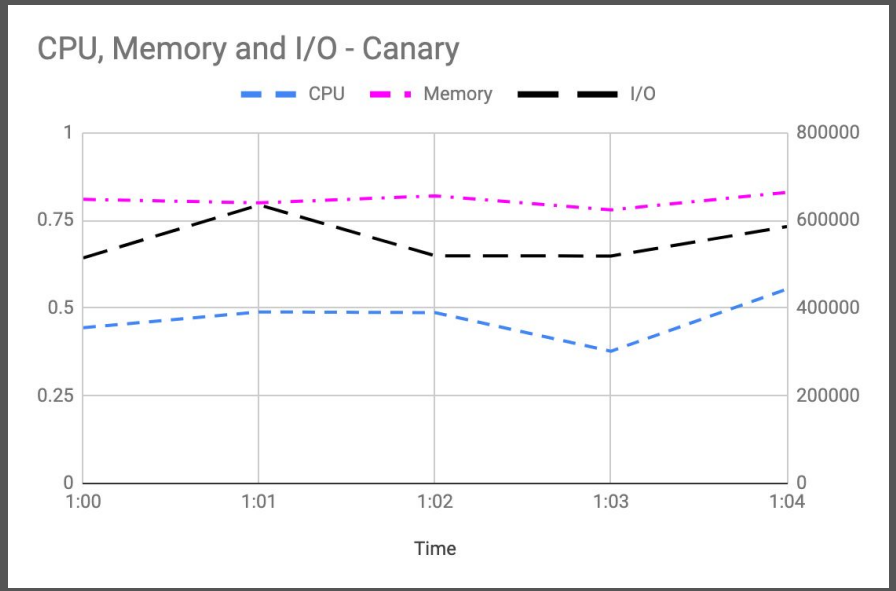
Gene

- His service is his “baby”
- Extremely Risk Averse
- Attains 99.999999% uptime, but runs on Ubuntu 10.04

Gene's call

"No way. Look at that I/O line! Let's meet on Monday and figure out what's going on."



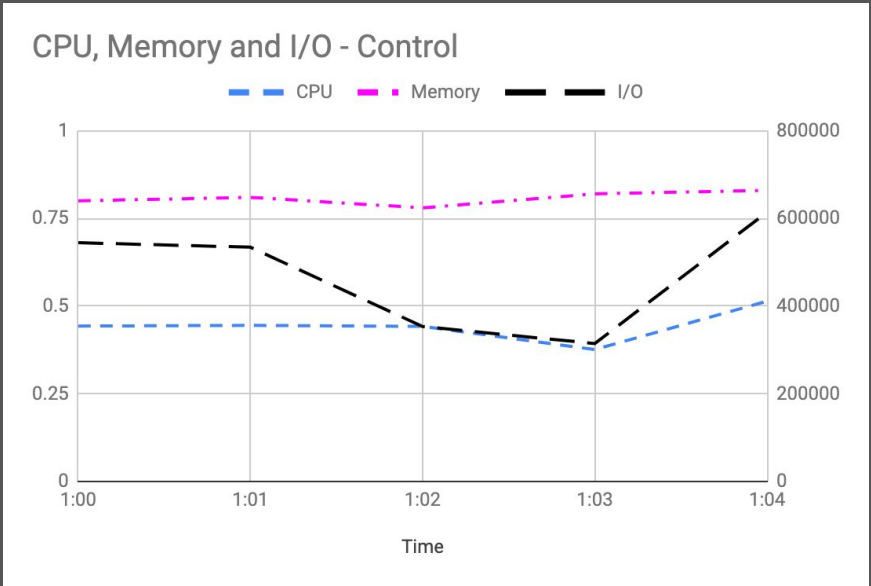


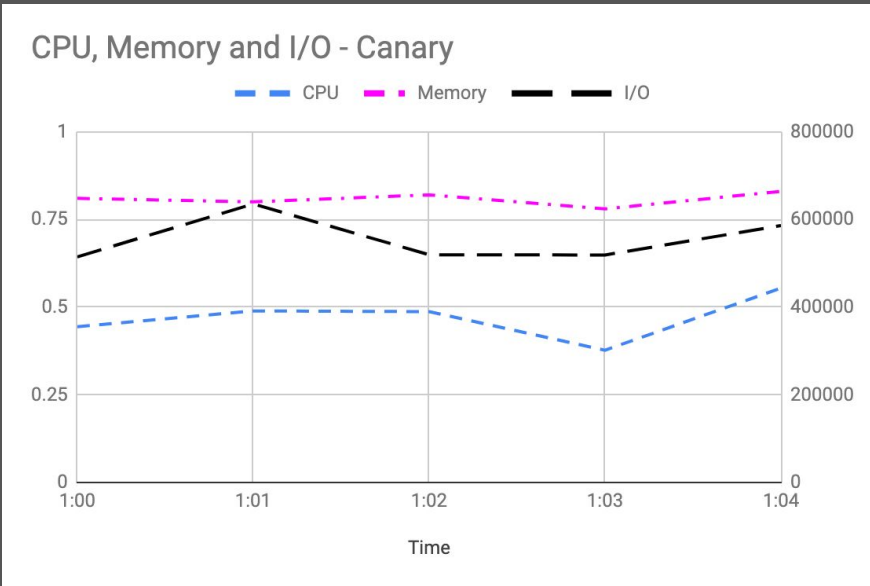
Angus

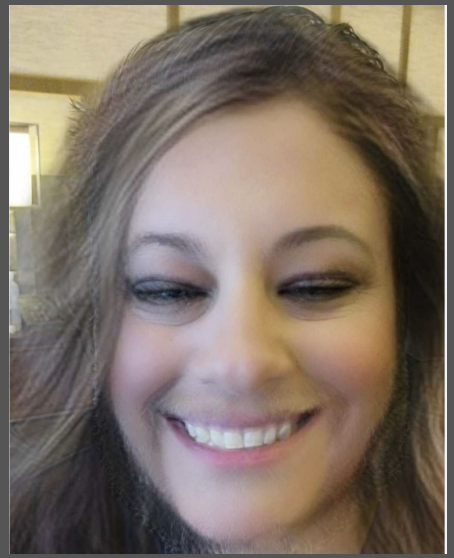
- Last job was at a Bitcoin startup
- Likes to “Move fast and break stuff”
- Yells “TONIGHT WE TEST IN PROD” at least once a day

Angus's call

"LGTM. SHIP IT!"





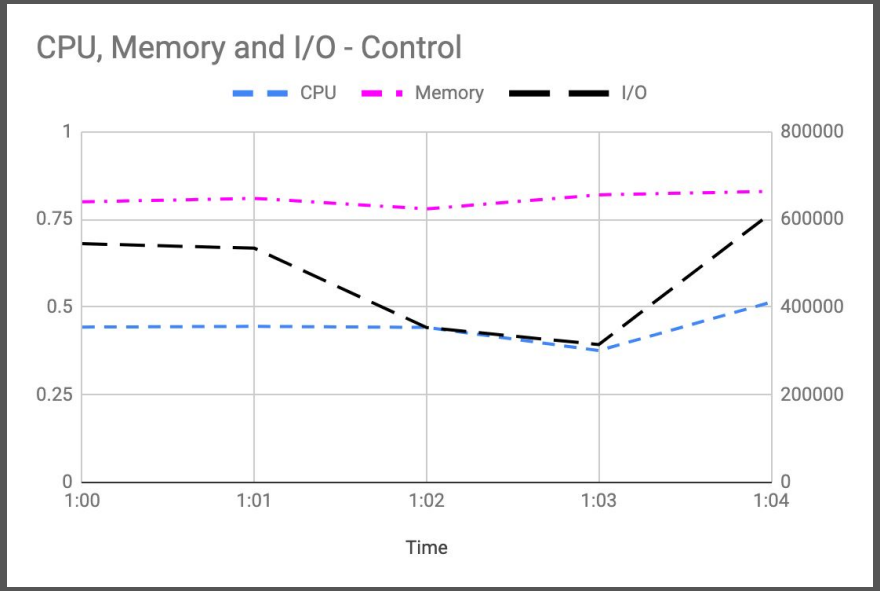


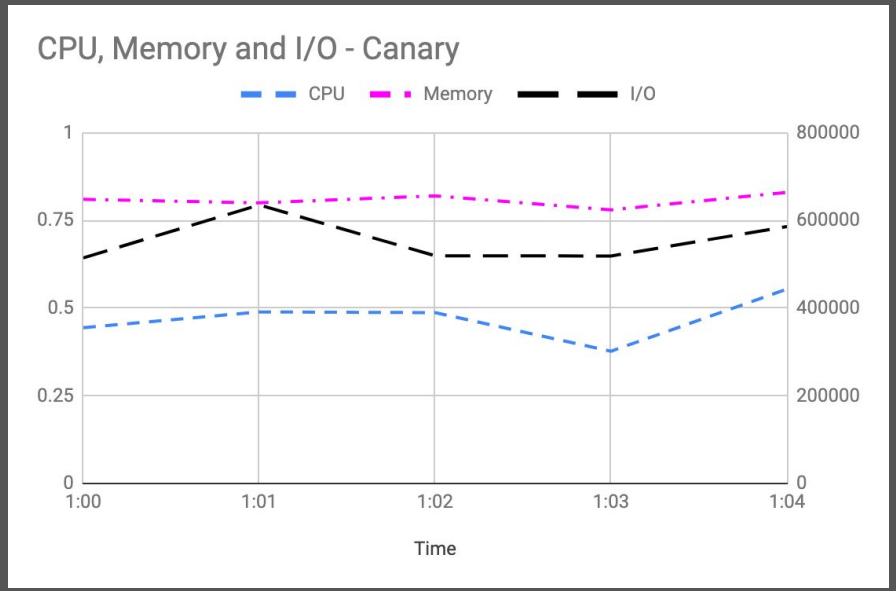
Carol

- Has 100% test coverage on all her code
- Is researching statistical models to improve fault detection

Carol's call

"I'll need a couple hours to run some additional tests against the new hosts. Stand by."





You are viewing the archives of a deactivated account

Close

Image credit: svgsilh.com

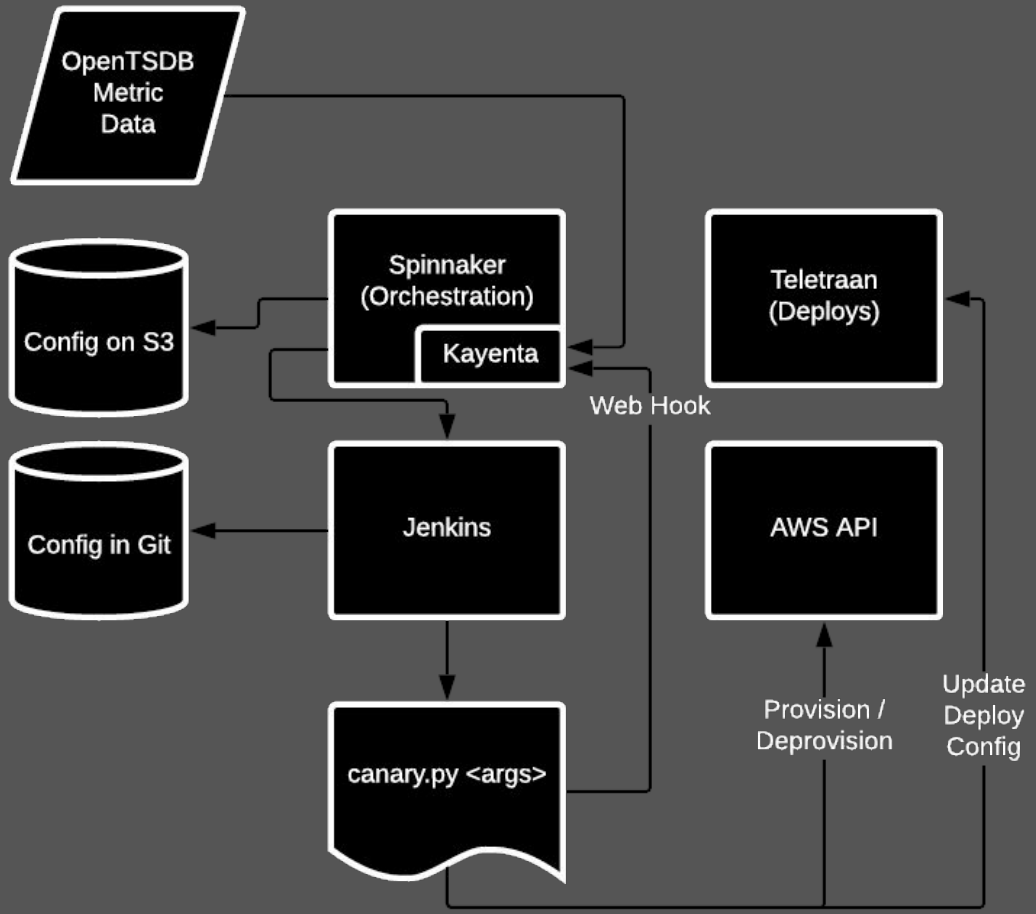
`<deleted>`

- The employee who left the company
- All of the knowledge they didn't write down about their service left with them!

Practical Considerations

Components

- CI/CD Pipeline
  - Workflow orchestration
    - https://www.spinnaker.io/
- Time-Series Metrics Database
- Execution environment for custom code
- Canary Judge software



Canary Best Practices

- Canary and Control should be similarly sized, in order to have a good basis for comparison.
- Both clusters should be serving a meaningful amount of production traffic.
- A minimum of 50 data points is required for a reliable Canary Analysis score
- You need to have metrics that reflect your application's health. (Golden Signals: latency, traffic, errors, and saturation.)*

*https://landing.google.com/sre/sre-book/chapters/monitoring-distributed-systems/#xref_monitoring_golden-signals

Overall Effort

- About one year of effort for 2-3 FTE
- Custom Python CLI application
- Kayenta: No native support for OpenTSDB, so we wrote our own

Lessons Learned

- UX is important
- Don’t pilot ACA using Gene’s app.
- Pilot ACA using Gene’s app.
- Have good and bad versions of your app to test with.

Additional Resources

- **Kayenta:** https://cloud.google.com/blog/products/gcp/introducing-kayenta-an-open-automated-canary-analysis-tool-from-google-and-netflix
- **Spinnaker Canary Analysis doc:** https://www.spinnaker.io/setup/canary/
- **Waze presentation on Canary Analysis:** https://cloud.google.com/blog/products/devops-sre/canary-analysis-lessons-learned-and-best-practices-from-google-and-waze
- **Netflix presentation:** https://medium.com/netflix-techblog/automated-canary-analysis-at-netflix-with-kayenta-3260bc7acc69

We’re hiring! Come work with us!

hiring-srecon@pinterest.com

Questions?