Canarying Well

Lessons Learned from Canarying Large Populations
(and small ones, too)

Štěpán “Steve” Davidovič, SRE at Google NYC, stepand@google.com
Twitter: @StepanDavidovic
Canarying: What is that?

- “Canary in a coal mine”
  - John Scott Haldane recommended use of canaries in coal mines to detect dangerous gases.
  - Canary breathes faster and is smaller, getting affected faster than a human.
Canarying: What is that?

- “Canary in a coal mine”
  - John Scott Haldane recommended use of canaries in coal mines to detect dangerous gases
  - Canary breathes faster and is smaller, getting affected faster than a human

- Let’s abstract that!
  - We have something large which we don’t want to harm
  - We have something small which we are more okay losing
  - Small thing detects danger, and we’re going into the unknown
Canarying: What is that?

● “Canary in a coal mine”
  ○ John Scott Haldane recommended use of canaries in coal mines to detect dangerous gases
  ○ Canary breathes faster and is smaller, getting affected faster than a human

● Let’s abstract that!
  ○ We have something large which we don’t want to harm
  ○ We have something small which we are more okay losing
  ○ Small thing detects danger, and we’re going into the unknown

● And apply to production systems…
  ○ We have a large service we want to sustain
  ○ We are okay losing a small chunk of it
  ○ We deploy production change with unknown impact to that small chunk, to detect danger
What we’re going to talk about

● Canarying as an A/B test of production systems
  ○ Specifically automated A/B test
  ○ No humans required to make decision

● Lots of illustrative examples

● Simple rules of thumb to get canarying quickly
What we’re not going to talk about

- Any particular technology to use for canarying (or monitoring, or deployment)
  - Your organization’s technology stack will likely dictate requirements anyway

- Statistics, accurate numbers, accurate math
  - There’s more knowledge on that in the audience :-)
  - Everything is illustration!

- Cover all important aspects of canarying
  - That’d take too long
To Canarying!
Conflicting Incentives
Conflicting Incentives

- Canarying time
  - You want fast rollout, for better development velocity
  - You want slow rollout, to have more data from the canary process
Conflicting Incentives

- Canarying time
  - You want fast rollout, for better development velocity
  - You want slow rollout, to have more data from the canary process

- Population size
  - You want small population, to minimize the impact on your service
  - You want large population, to maximize statistical significance of findings
Conflicting Incentives

- **Canarying time**
  - You want fast rollout, for better development velocity
  - You want slow rollout, to have more data from the canary process

- **Population size**
  - You want small population, to minimize the impact on your service
  - You want large population, to maximize statistical significance of findings

- **Metric selection**
  - You want many metrics, to cover as many failure modes as you can think of
  - You want few metrics, to avoid increased risk of random benign failures
  - Extra: Some metrics *need* large population, or longer canarying time, or...
Triangle Of Canarying Priorities

Fast Rollout
(short canary)

Large Metric Coverage

Small Impact Of Bad Canary
Triangle Of Canarying Priorities

- Need to ensure balance between the three goals

- Want fast rollout?
  - Might need to compromise on large coverage, or small blast zone

- Want small impact of a bad canary?
  - Might need to compromise on large coverage or fast rollout

- Want large metric coverage?
  - Might need to compromise on fast rollout or small blast zone to have enough data
Triangle Of Canarying Priorities

- Need to ensure balance between the three goals

- Want fast rollout?
  - Might need to compromise on large coverage, or small blast zone

- Want small impact of a bad canary?
  - Might need to compromise on large coverage or fast rollout

- Want large metric coverage?
  - Might need to compromise on fast rollout or small blast zone to have enough data

- Ignore these → increase risk of flakey canarying
  - As with all rules of thumb, this is not universal truth
  - Importantly, this is not binary! You don’t really “give up” fast rollout, you compromise on it
Canary Population
apples to apples
Control

~20ms

Canary

~120ms

OpenClipart, “world map” by molumen. License information: https://openclipart.org/may-clipart-be-used-comparison
Example: Geographical distribution

![Diagram showing geographical distribution with time on the x-axis and latency on the y-axis. The diagram includes points labeled as $T_0$, $T_{+24h}$, and shows the comparison between Canary and Control groups.](image-url)
Example: Geographical distribution

Look at the latency!
Example: Geographical distribution

- We’ve seen a 100ms increase in latency for our canary!
Example: Geographical distribution

- We’ve seen a 100ms increase in latency for our canary! Hang on a minute…

- We have observed that new release *and different continent* have a significant slowdown
  - Maybe not what we wanted to measure?
Takeaways

● It’s important to compare apples to apples
  ○ Better solution here would be to compare within a continent
  ○ Or to take slice of service in each continent, and compare it within the continent

● Your goal is to judge whether the production change is good
  ○ ...but what other things have you accidentally picked up in the process?
Canary Population

high variance behavior
Example: High variance among replicas
Example: High variance among replicas
Example: High variance among replicas
Example: High variance among replicas

Look at the latency!
Takeaways

- Canary selection process is a big source of potential false positives
  - What’s the probability of selecting enough outliers to sway the canary test?
Takeaways

- Canary selection process is a big source of potential false positives
  - What’s the probability of selecting enough outliers to sway the canary test?

- Selection process need not be random to cause problems
  - Consistently going from first replica to last replica is random in relation to the metrics!
  - Selection process is often oblivious to the metrics
  - Worse: There is often no single good solution to canary selection!
Takeaways

● Canary selection process is a big source of potential false positives
  ○ What’s the probability of selecting enough outliers to sway the canary test?

● Selection process need not be random to cause problems
  ○ Consistently going from first replica to last replica is random in relation to the metrics!
  ○ Selection process is often oblivious to the metrics
  ○ Worse: There is often no single good solution to canary selection!

● Before/after information can help, but it has its own pitfalls
  ○ More on that later
Example: Bimodal distribution
(Two server platforms?)
Example: Two metrics, different outliers
Example: Two metrics, different outliers

Latency

Errors

Canary

Canary
Takeaways #2

● The input data can be almost arbitrarily distributed
  ○ For multimodal data distributions, clustering methods may be very helpful

● Metric distribution depends both on population and metric
  ○ Example: crashes may be the same regardless of platform, latency might vary greatly

● Any two metrics can follow different distributions for the same population
  ○ Decisions on handling data are on \{population pair, metric distribution\} basis
  ○ No silver bullet for the entire population
    ■ Unless you have some domain specific prior knowledge…

● Your canary process might need to drop some metrics tests
  ○ There might not be a single canary selection that avoids outliers for all metrics!
Canary Duration

testing post-startup Behavior
Example: Service With Cache

Canary restarted, now has cold cache
Example: Service With Cache

Is this okay?
Example: Service With Cache

Look at the latency!
Example: Service With Cache

- But can’t I just restart some fraction of control too?
- Then I’m comparing apples to apples...
Example: Service With Cache, restarted

Let's restart both...
Example: Service With Cache, restarted

- Time: $T_0$ to $T_{+1h}$
- Latency
- Looks good!

- $T_{+24h}$
Example: Service With Cache, restarted

Hang on a minute...
Example: Service With Cache, restarted

- Latency

- Looks like regression!
Takeaways

● Restarting fraction of control often doesn’t yield good results

● You are deciding the fate of apples by comparing oranges to oranges
  ○ Apples are how the service operates, oranges how it starts up
  ○ It needs to be connected to the thing you actually meant to decide

● Instead: Identify “point of stable operation” and canary from there
  ○ This will take extra time in your release process, but it’s unavoidable if you want to measure the actual service behavior, rather than its post-startup behavior
Canary Duration

testing long-term trends
Example: Memory leak canary
Example: Memory leak canary

![Graph showing memory leak canary example with time, RAM (GB), limit, and T₀ and T⁺₂₄h milestones.]
Example: Memory leak canary
Example: Memory leak canary

Is this okay?
Example: Memory leak canary

Time

RAM (GB)

limit

$T_0$ $T_{+1h}$ $T_{+24h}$

Maybe?
Takeaways

- Detecting memory leaks with canarying may require long canarying
  - Pushes us away from “fast rollout (short canary)”

- Not all memory-based tests require long canary
  - What if your canary unexpectedly has 2x memory usage than your control right after start?
  - Easy to detect quickly!

- Canarying doesn’t obviate the need to monitor
  - Recall canarying is time-limited!
Canary Duration
before/after test
Before/after test

- With all these problems, maybe I can just quickly deploy?
  - We have fast rollback mechanism, we won’t hurt service terribly poorly...
Example: Before/after test
Example: Before/after test
Example: Before/after test
Example Takeaway

● Before/after tests often have us compare “apples to stale apples”

● Global comparison (e.g. 5% canary in each region) not always helpful
  ○ You may smooth out day cycle, but only partially
  ○ You may have different usage patterns in different regions

● Smaller-scale issues than day cycle include:
  ○ Brief traffic spikes
  ○ Machine deaths

● Not always the wrong choice! But be very cautious.
  ○ Catastrophic canary = loss of entire service until you can roll back!
Increasing Coverage
by adding more tests
Example: Compound probability

- Imagine you have 1% failure rate on each of your canary tests.
  - Might be false positive, might be true positive, doesn’t matter for this example.
  - Some might claim 1% false positive rate would be pretty nice score.
- If a metric fails the canary test, human needs to take a look.
Example: Compound probability

- Imagine you have 1% failure rate on each of your canary tests.
  - Might be false positive, might be true positive, doesn’t matter for this example.
  - Some might claim 1% false positive rate would be pretty nice score.
- If a metric fails the canary test, human needs to take a look.

\[ P(\text{single test failure}) = 1\% \]
Example: Compound probability

- Imagine you have 1% failure rate on each of your canary tests.
  - Might be false positive, might be true positive, doesn’t matter for this example.
  - Some might claim 1% false positive rate would be pretty nice score.
- If a metric fails the canary test, human needs to take a look.
- You have 100 metrics to test.

\[
P(\text{single test failure}) = 1% \\
P(\text{at least one test failure out of 100}) = 1 - P(\text{no test failure}) = 1 - (1 - 1%)^{100} \\
= 1 - (0.99)^{100} \approx 1 - 0.366 \approx 63.4% 
\]
Example: Compound probability

- Imagine you have 1% failure rate on each of your canary tests.
  - Might be false positive, might be true positive, doesn’t matter for this example.
  - Some might claim 1% false positive rate would be pretty nice score.
- If a metric fails the canary test, human needs to take a look.
- You have 100 metrics to test.
- Two out of three releases will need human inspection!
Example: Compound probability

- **This was all a lie!**
  - Many assumptions made which are untrue.
  - Example: If your release is bad, failures are likely correlated, not independent.

- **But it’s a useful lie**
  - "All models are wrong, but some are useful" (George Box, statistician)
  - Maybe you have 1% false positive ratio in your system? Or 0.1%?
  - It’s a simplified analysis of achievability of your goals
Example: Compound probability

- You are likely able to increase accuracy of your test by giving it more data
  - Which means more time and/or larger population
Beware Meta Analysis

● It’s easy to bring meta analysis in:
  ○ “But what if I require that any 10% checks need to fail before I bring a human?”

● This has flaws, and might be a slippery slope:
  ○ Why 10%? Our canary checks are anchored in expected system behavior, 10% is not
  ○ Checks are not always of equal importance, so next on our slippery slope is adding weights
    ■ How do you decide those?
  ○ Slippery slope leads to tuning magic numbers not anchored in either goals or statistics
Prefer Few Metrics

- Key question:
  “What are the 3 metrics that most clearly indicate service health?”
Prefer Few Metrics

- Key question: “What are the 3 metrics that most clearly indicate service health?”

- Some of them should connect to your service’s SLI!

- Doesn’t mean you need 3 metrics, just that you can test for many issues with just a few
So in conclusion...
Canary In These 3 Simple Steps

1. Choose only few good metrics
   ○ They need to represent measure of problems in your service
   ○ They should be connected to your SLI

2. Ensure representative population
   ○ Random processes can be bad for you, increasing false positives

3. Compare apples to apples
   ○ If metric exhibits post-startup deviant behavior, maybe wait a bit
   ○ If you are deployed globally with mutually different behavior, maybe compare within regions
Canary In These 3-ish Simple Steps

1. Choose only few good metrics
   ○ They need to represent measure of problems in your service
   ○ They should be connected to your SLI

2. Ensure representative population
   ○ Random processes can be bad for you, increasing false positives

3. Compare apples to apples
   ○ If metric exhibits post-startup deviant behavior, maybe wait a bit
   ○ If you are deployed globally with mutually different behavior, maybe compare within regions

4. Multi-stage canarying
   ○ First canary with very high confidence metrics, small population, fast canary -- fail fast
   ○ If that passes, do a larger population, longer duration, allowing you to use "worse" metrics
Takeaways

- Canarying can be easy and approachable.
  - Just follow three (or four) simple steps...
  - ...and then iterate on them to find what’s best for you & your systems!

- Recall triangle of Canarying Priorities

- Statistics foundations are useful

- Some of the questions can be automated away with sufficiently advanced software