Argo: Real Enterprise-scale with Kubernetes

Al Kemner, Principal Engineer & Architect

Caleb Troughton, Product Manager

Daniel Jimbel, Staff Engineer

Safe Harbor

This presentation and the information herein (including any information that may be incorporated by reference) is provided for informational purposes only and should not be construed as an offer, commitment, promise or obligation on behalf of New Relic, Inc. ("New Relic") to sell securities or deliver any product, material, code, functionality, or other feature. Any information provided hereby is proprietary to New Relic and may not be replicated or disclosed without New Relic's express written permission.

Such information may contain forward-looking statements within the meaning of federal securities laws. Any statement that is not a historical fact or refers to expectations, projections, future plans, objectives, estimates, goals, or other characterizations of future events is a forward-looking statement. These forward-looking statements can often be identified as such because the context of the statement will include words such as "believes," "anticipates," "expects" or words of similar import.

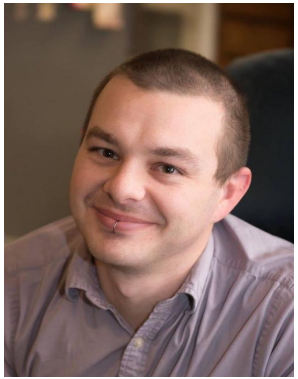
Actual results may differ materially from those expressed in these forward-looking statements, which speak only as of the date hereof, and are subject to change at any time without notice. Existing and prospective investors, customers and other third parties transacting business with New Relic are cautioned not to place undue reliance on this forward-looking information. The achievement or success of the matters covered by such forward-looking statements are based on New Relic's current assumptions, expectations, and beliefs and are subject to substantial risks, uncertainties, assumptions, and changes in circumstances that may cause the actual results, performance, or achievements to differ materially from those expressed or implied in any forward-looking statement. Further information on factors that could affect such forward-looking statements is included in the filings New Relic makes with the SEC from time to time. Copies of these documents may be obtained by visiting New Relic's Investor Relations website at ir.newrelic.com or the SEC's website at www.sec.gov.

New Relic assumes no obligation and does not intend to update these forward-looking statements, except as required by law. New Relic makes no warranties, expressed or implied, in this presentation or otherwise, with respect to the information provided.

Who We Are

Al Kemner

Principal Engineer & Architect



Caleb Troughton

Product Manager

Daniel Jimbel

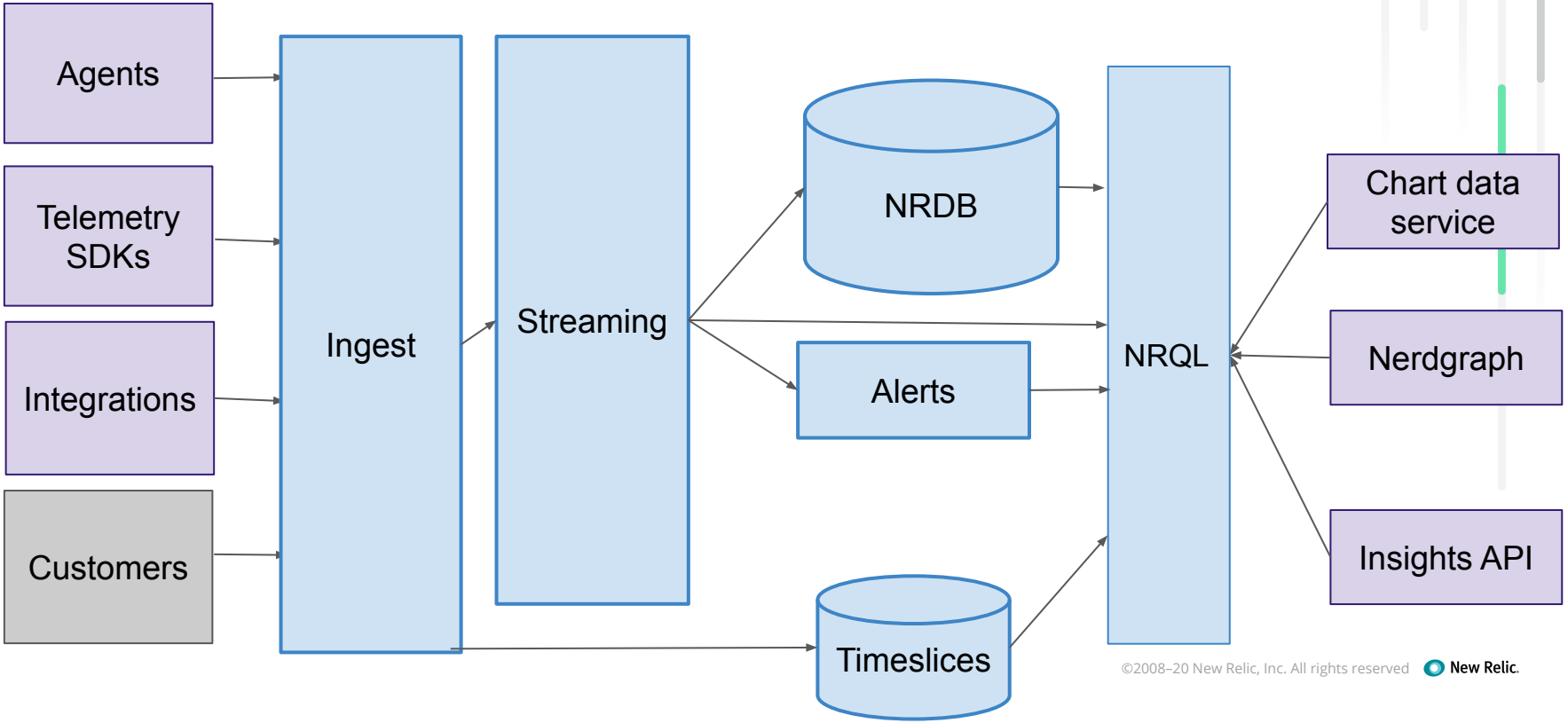
Staff Engineer

Agenda

1. New Relic's Architecture Overview
2. Argo CD
3. Argo Rollouts (Demo)
4. Orchestration at Scale
5. Argo Workflows (Demo)

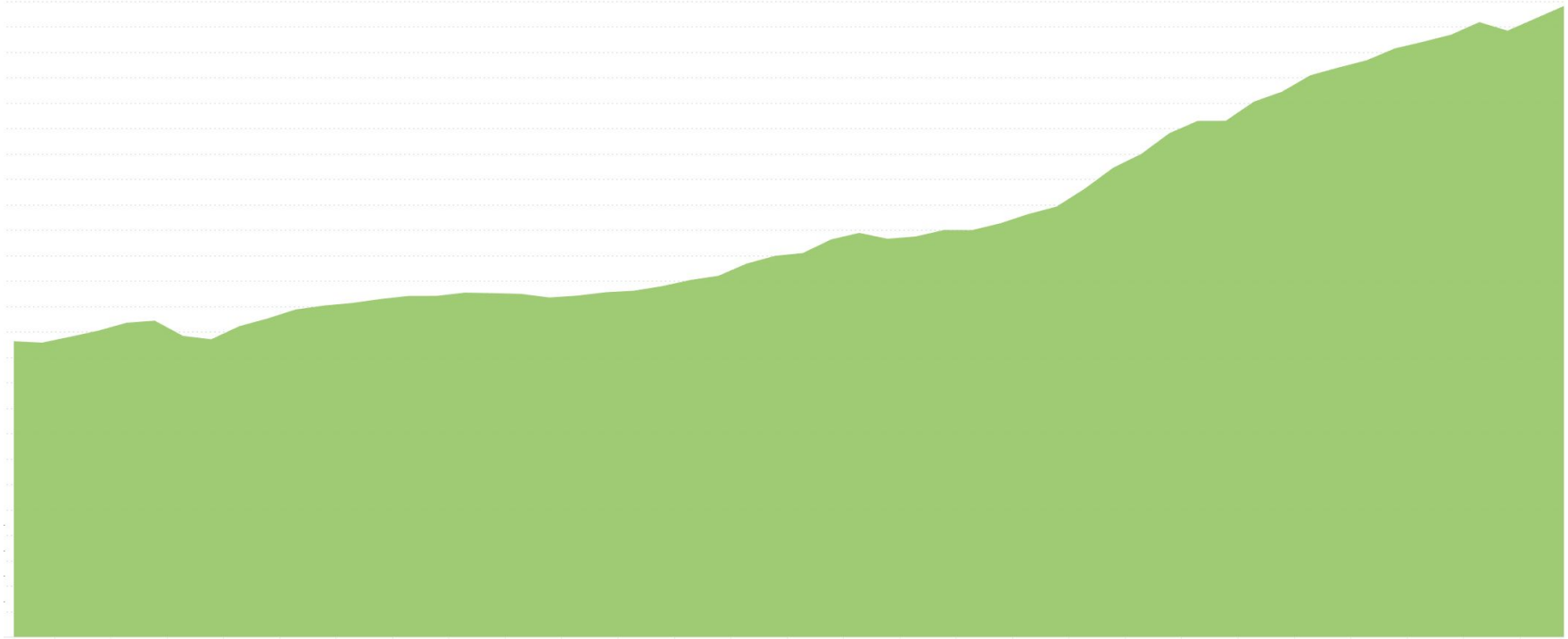
New Relic's Architecture Overview

New Relic's Architecture Overview



New Relic.

Data Growth



Argo CD

Why Argo?

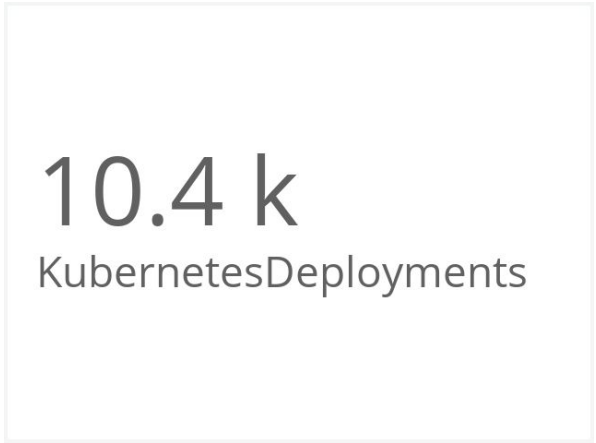
- Argo = Get stuff done with Kubernetes
- New Relic runs on Kubernetes

Argo CD

- Declarative, GitOps continuous delivery tool for Kubernetes.
- Argo CD Features
  - Automated deployment of applications to specified target environments
  - Support for multiple config management/templating tools (Kustomize, Helm, Ksonnet, Jsonnet, plain-YAML)
  - Ability to manage and deploy to multiple clusters
  - SSO Integration (OIDC, OAuth2, LDAP, SAML 2.0, GitHub, GitLab, Microsoft, LinkedIn)
  - Multi-tenancy and RBAC policies for authorization
  - Health status analysis of application resources
  - Automated configuration drift detection and visualization
  - Automated or manual syncing of applications to its desired state
  - Web UI which provides real-time view of application activity
  - CLI for automation and CI integration

Multi-Cloud Architecture

- Split between cloud and physical data centers
- Cloud infrastructure shared across many clusters
- Different types of clusters for different workloads
  - As few as ~2, as many as ~20 depending on the type
  - Clusters are BIG
- ~3000 applications in one ArgoCD



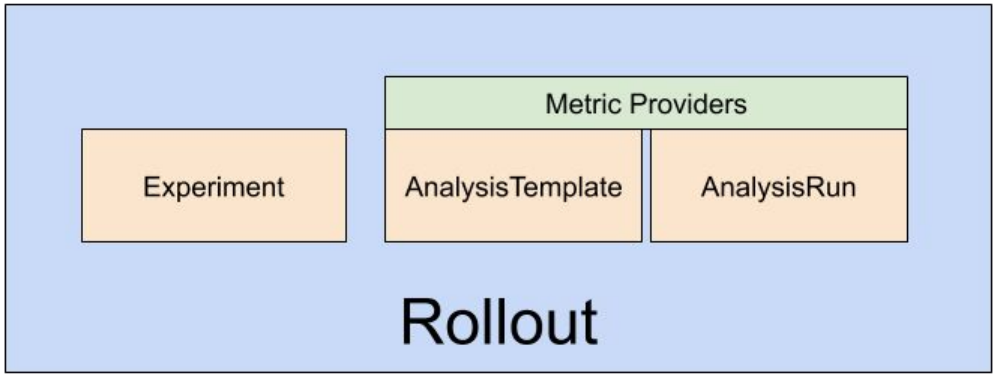
Argo Rollouts

Argo Rollouts: The goals

- Canary deploys for every new change, with a bake time to detect problems in the canary.
- Automate canary analysis so humans don’t need to babysit the deploy.
- Automate rollback in case something **does** go wrong.

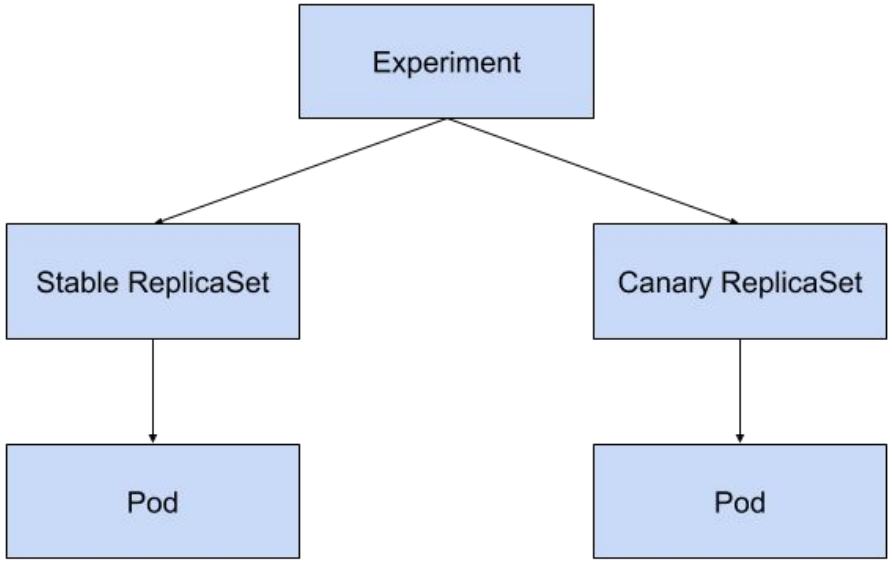
Argo Rollouts: The pieces

- Custom Resources
  - Experiments
  - AnalysisTemplates
  - AnalysisRuns
  - Rollouts
- Metric Providers



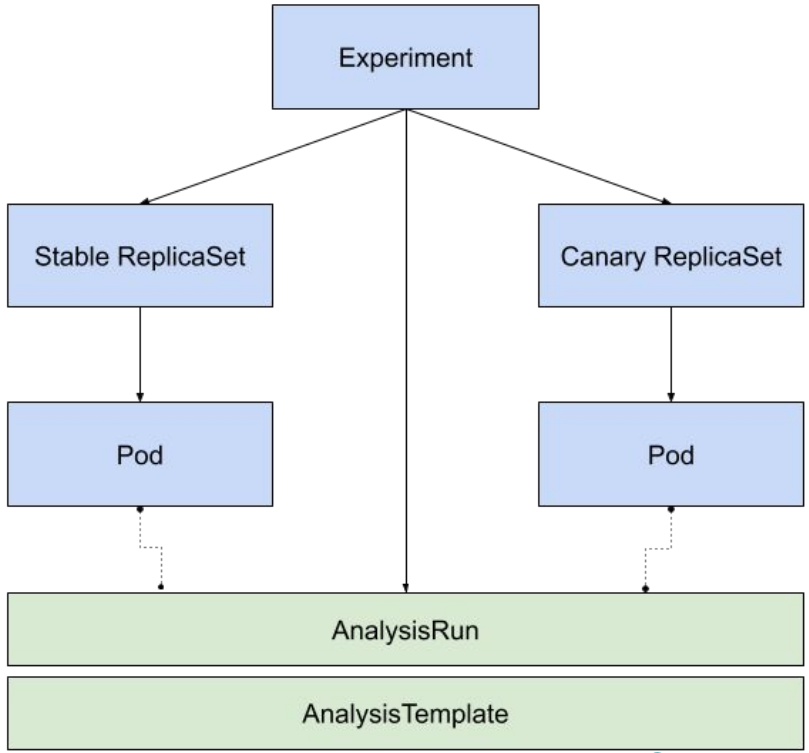
Argo Rollouts: Experiments

- Creates two ReplicaSets with different Pod specs
- Compare metrics across these ReplicaSets using AnalysisTemplates



Argo Rollouts: AnalysisTemplates and AnalysisRuns

- AnalysisTemplates
  - Describe which metric providers to query, what to query, and what “success” looks like.
- AnalysisRuns
  - An individual instance of an AnalysisTemplate, run against the pods created by an Experiment.



Argo Rollouts: Rollouts

- Resource that acts as a drop-in replacement for a Deployment.
- Like Deployment, but with specialized deployment strategies for Canary or Blue-Green Deploys.
- Wraps up all of the Experiment and Analysis functionality into deployment steps.

```
apiVersion: argoproj.io/v1alpha1
kind: Rollout
metadata:
  name: guestbook
spec:
…
  strategy:
    canary:
      steps:
      - setWeight: 20
      - pause: {duration: 5m}
      - analysis:
          templates:
          - templateName: success-rate
          args:
          - name: service-name
            value: guestbook-svc.default.svc.cluster.local
```

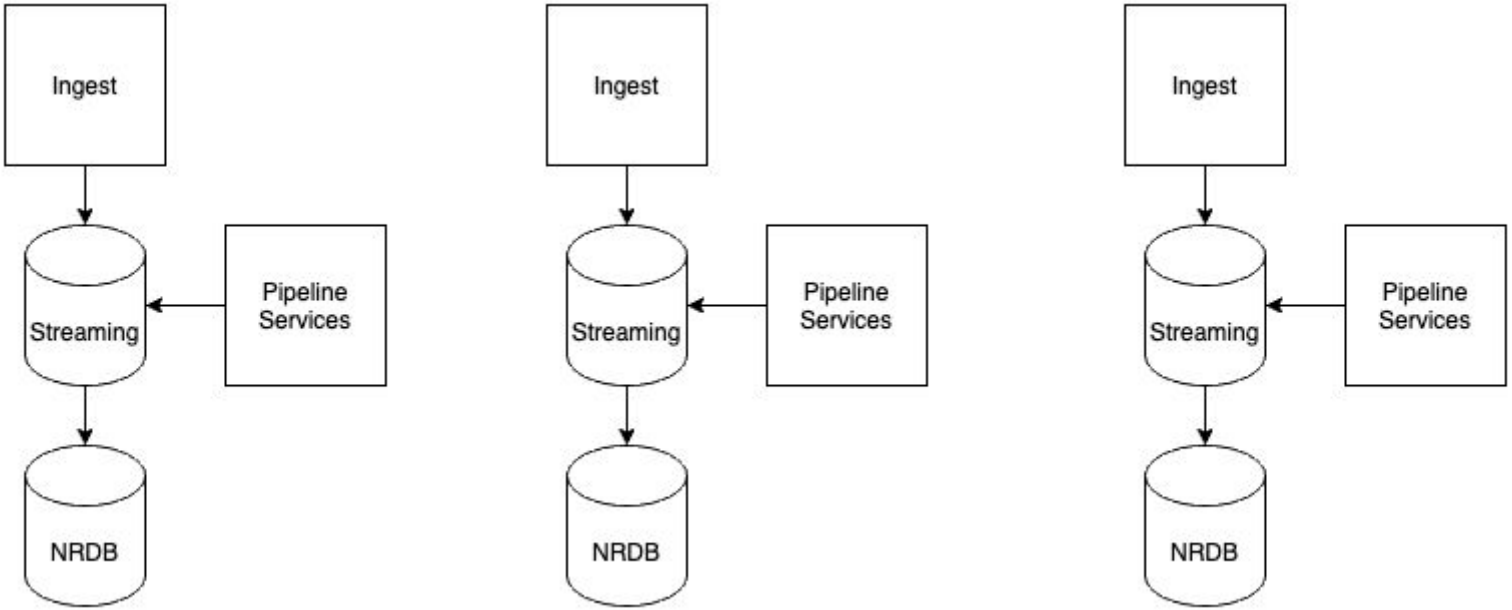
Argo Rollouts: Metric Providers

- Job
  - Run a Kubernetes Job. A non-zero exit indicates failure.
- Web
  - Perform an HTTP request against any JSON endpoint, and inspect the response payload to determine success/failure.
- Kayenta
  - Perform Mann-Whitney analysis. Requires an instance of Kayenta running, which has its own set of metric providers.
- Prometheus
  - Execute PromQL queries to fetch metrics.
- Commercial Providers
  - New Relic, Datadog, Wavefront

Demo Time

Orchestration at Scale

Orchestration at Scale



High Scalability post

http://highscalability.com/blog/2012/5/9/cell-architectures.html

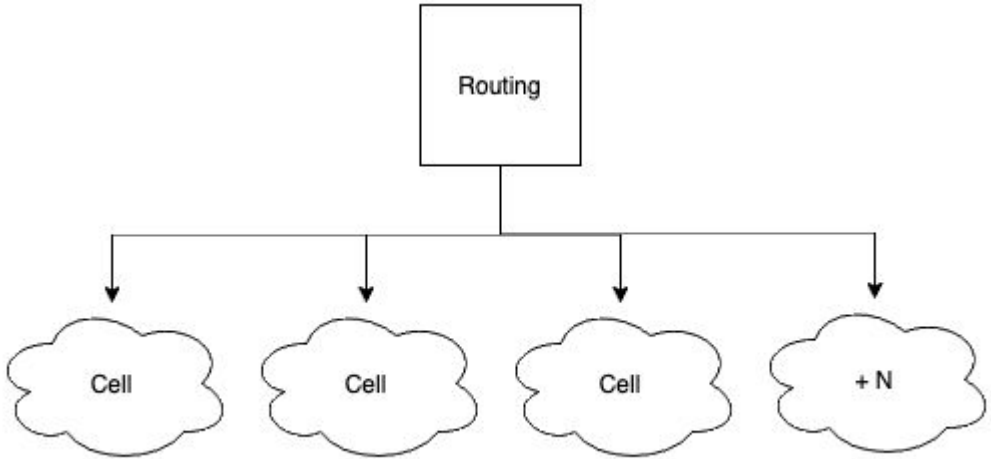
> A Cell Architecture is based on the idea that massive scale requires parallelization and parallelization requires components be isolated from each other. These islands of isolation are called cells. A cell is a self-contained installation that can satisfy all the operations for a shard. A shard is a subset of a much larger dataset, typically a range of users, for example.

High Scalability post

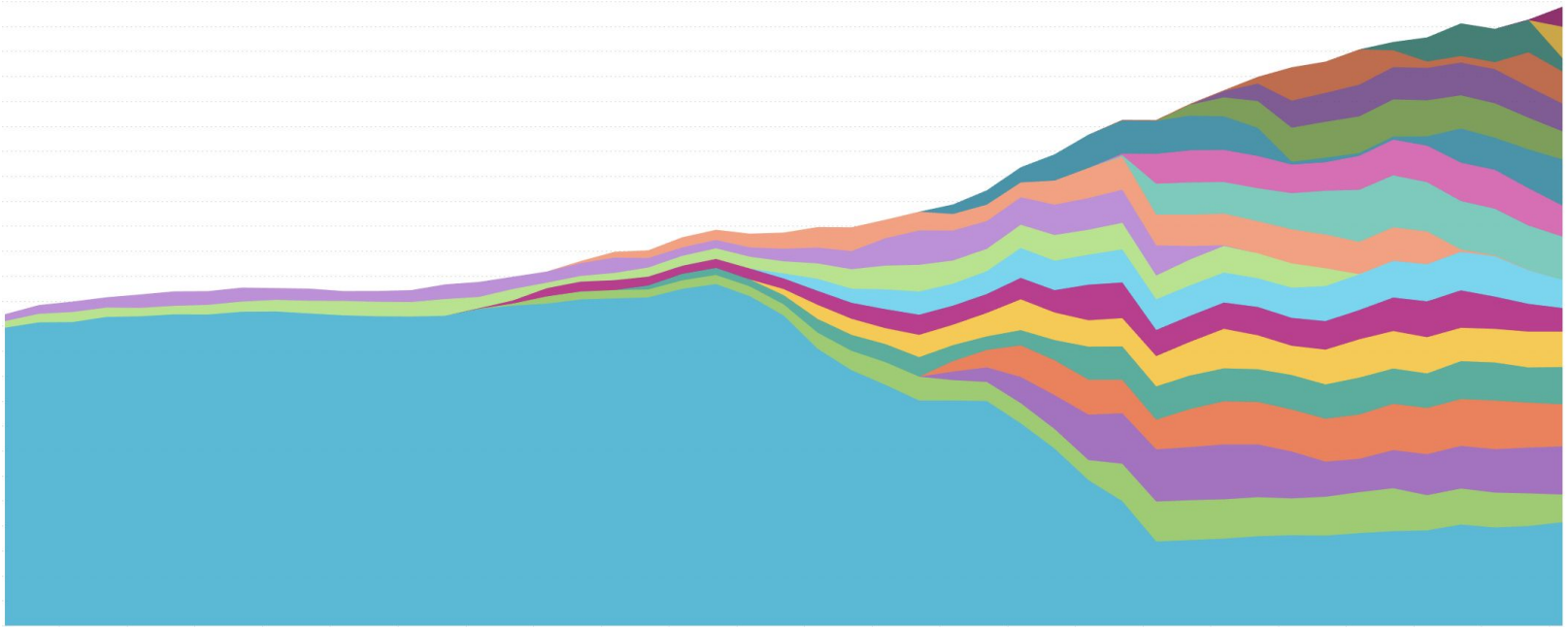
http://highscalability.com/blog/2012/5/9/cell-architectures.html

- **Cells are added in an incremental fashion as more capacity is required.**
- **Cells isolate failures. One cell failure does not impact other cells.**
- Cells provide isolation as the storage and application horsepower to process requests is independent of other cells.
- Cells enable nice capabilities like the ability to test upgrades, implement rolling upgrades, and test different versions of software.
- Cells can fail, be upgraded, and **distributed across datacenters** independent of other cells.

Orchestration at Scale



Orchestration at Scale



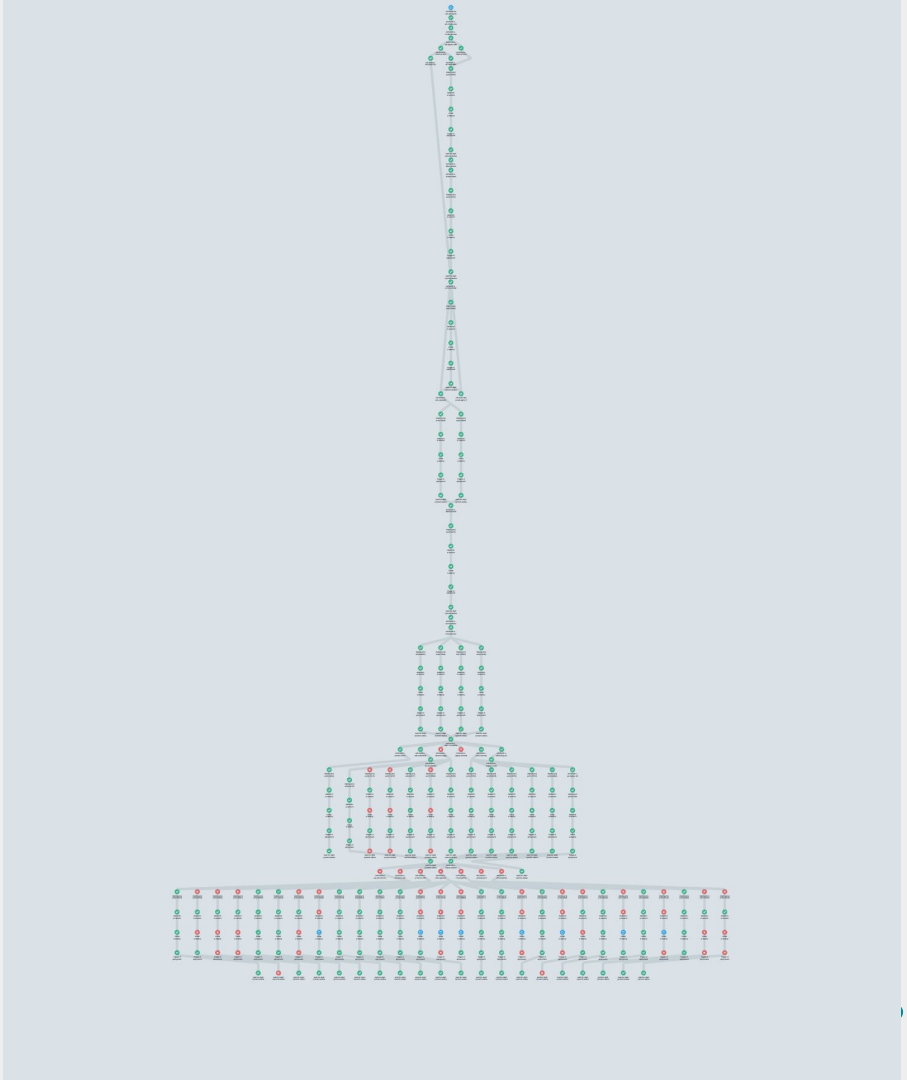
Argo Workflows

Argo Workflows

- container-native workflow engine
- Cloud agnostic and can run on any Kubernetes cluster
- a step in a workflow is a container

Cell Build Workflow

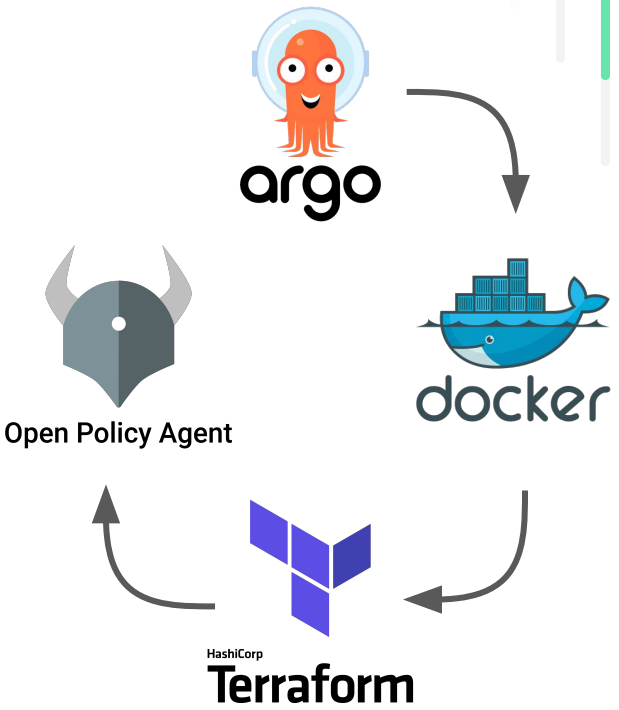
- 2+ times per week
- Retries and restarts biggest pain point



Argo Terraform Workflow

Project goals

- Deploy new cells using Terraform and Argo.
- Every step has to be idempotent.
- The solution has to be generic for all teams.
- Make sure that is secure to apply it.



How we have done it

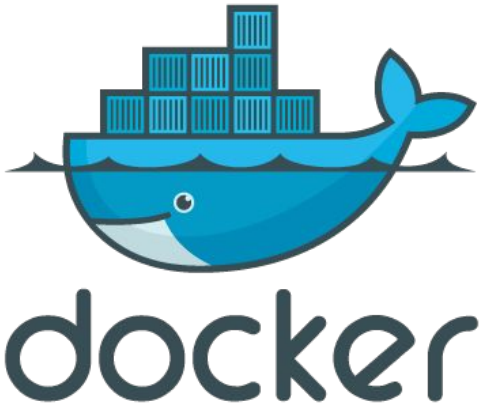
Argo Workflows

- A Workflow can be triggered from another Workflow.
- We can make use of the artifacts to clone the github repository with the Terraform code.
- It gives us the opportunity to be triggered with different parameters, i.e: cell name.

How we have done it

Docker

- We created a custom Docker image with Terraform and Open Policy Agent to be able to run our code on an automatic and safety way.
- Docker give us the ability to create a custom container and adapt it to our needs.



How we have done it

Terraform

- Is able to create all our Infrastructure using IaC.
- Using Terraform Workspaces we can run the same code for every cell without changing it.
- We can use tfvars files with the name of the cell to pass it to terraform if the file is present.
- We create the terraform workspace if it doesn't exists when creating new cells.

How we have done it

Open Policy Agent

- It helps us detecting if undesirable changes are going to be applied and cancel the job if needed.
- This allow us to apply changes in an automatic way without any kind of human supervision.

Open Policy Agent

Demo Time

Q & A

We’re always hiring!
https://newrelic.com/about/careers

Thank You

New Relic.