SRE

Take the climb to MultiCloud

by Marcel Birkner
Bio

Marcel works as a Staff Site Reliability Engineer at Instana, an Application Performance Monitoring (APM) solution. He has long experience in software engineering and software automation. Currently he focuses on improving the current Kubernetes stack, reducing overall system complexity and installing Instana SaaS infrastructure in IBM Cloud.

@MarcelBirkner

github.com/marcelbirkner
Abstract

For Instana MultiCloud is not just a buzzword, but an opportunity to grow our customer base. We initially offered our SaaS solution in AWS Cloud. Last year we opened new SaaS regions in Google Cloud and this year we are adding SaaS regions in IBM Cloud.

We knew that the platform and infrastructure that got us through the first five years needed an overhaul to prepare us for more growth. Our customers have strict requirements regarding compliance, security and data governance. That is when we decided to update our infrastructure to be able to open new SaaS regions with other cloud providers.

I will present the challenges we faced during the last two years. Running the old stack, not breaking existing customers and designing and implementing our new infrastructure that will serve us the next five years.
Who We Are
SRE Team

- 3 Time zones
- 24 / 7 / 365 support
- On-call rotation
- Team members have operations and software engineering background
Stats

- 10 different datastore clusters per region
- 8K+ Containers Running in SaaS

SaaS stats from 2020
Our MultiCloud Journey
Where we were 2018

SAAS:
- Single Cloud Provider
- 2 x AWS regions
- HashiCorp (Nomad/Consul)
- Ansible playbooks

On-Premises:
- package based
- Chef cookbooks

Challenges
- Growth
- Compliance/Security/Data governance
2021

SaaS:
- Multi Cloud Strategy
- 2 x AWS regions
- 2 x GCP regions
- first IBM region (internal customers only atm)
- Kubernetes

On-Premises:
- Docker
- Kubernetes
Identify Challenges
Identify challenges

- What is **working well** in the current infrastructure?
- What needs to be **improved**?
- How can we save **daily toil**?
- How do we want to run SaaS product in the **future**?

Focus on the big picture

- try not to solve all problems at once
- some requirements will change

Our "Big Picture"

- Kubernetes
- Shared configuration / code for SaaS and On-Premises
- Reduce complexity / toil
Goal #1: Single datastore migration codebase (SaaS / On-Premises)

up to 2019

Challenges: Each datastore had its own migration tool. Duplicate scripts for SaaS and OnPrem.
- Cassandra (cassandra-migrator)
- ClickHouse (golang-migrate)
- Elasticsearch (http-client)
- Kafka (kafka-cli)
- MongoDB (mongo migrator)
  - replaced by CockroachDB
- PostgreSQL (flyway db)
  - replaced by CockroachDB

Runtimes: Ruby/Python/Java

2020

instanactl
- GoLang CLI
  - cobra library
  - golang-migrate library
- codebase used by SaaS and On-Premises
- single place for database migration scripts

Runtimes: GoLang Binary
Goal #2: Shared configuration & codebase (SaaS / On-Premises)

up to 2019

Challenges:
- separate component configuration
- separate packaging
  - SaaS: Docker
  - OnPrem: RPM / DEB
- separate delivery
  - SaaS: Ansible
  - OnPrem: Chef

2020

- shared component configuration
- shared OCI container images
- shared migration tool
- K8s deployments via instanactl

Runtimes: GoLang Binary

Runtimes: Python / Ruby

Supported Operating Systems

Ubuntu, Debian, RedHat, CentOS, Amazon Linux
Goal #3: Infra. config versioned with product (SaaS / On-Premises)

up to 2019

Challenges:
- SaaS and OnPrem had separate repositories for datastore migrations and component configuration
- no common versioning with product source code
- hard to coordinate releases and hotfixes

2020

- Mono-Repo for product source code, component configuration and datastore migration scripts
  - release branches (release-199, release-200, ...)
- releases are easily rolled out from release branches
- easy coordination of SaaS and OnPrem releases and hotfixes
Shared Infrastructure Modules (SaaS / On-Premises)
Migration process
Infrastructure & Code Changes

- new architecture (global vs regional components)
- migrate MongoDB + PostgreSQL to CockroachDB
- component refactoring
"GLOBAL" environment GoLive Steps

Go Live Steps (Tuesday 9am)

Last known: Monday

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test impact before test start.</td>
<td>n/a</td>
</tr>
<tr>
<td>Test to UI to validate in session.</td>
<td>n/a</td>
</tr>
<tr>
<td>Validate not to impact production server, try.</td>
<td>n/a</td>
</tr>
<tr>
<td>Start production for service, try.</td>
<td>n/a</td>
</tr>
<tr>
<td>Validate UI to users in session.</td>
<td>n/a</td>
</tr>
<tr>
<td>Test to UI to validate in session.</td>
<td>n/a</td>
</tr>
<tr>
<td>Validate not to impact production server, try.</td>
<td>n/a</td>
</tr>
<tr>
<td>Start production for service, try.</td>
<td>n/a</td>
</tr>
<tr>
<td>Validate UI to users in session.</td>
<td>n/a</td>
</tr>
</tbody>
</table>

GoLive Environment GoLive Steps

GoNoGo Testing

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
</table>
| T1 | Hadoop
| T2 | Log/Authentication (tunnel)
| T3 | Google Single Sign On (SSO)
| T4 | SPA
| T5 | Buffer: Tenant (buffer&instana.user/database)
| T6 | Check root disk
| T7 | Inventory disk
| T8 | Check root disk
| T9 | Check root disk
| T10 | Check root disk
| T11 | Check root disk
| T12 | Check root disk
| T13 | Check root disk
| T14 | Check root disk
| T15 | Check root disk

Post GoLive Cleanup

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
</table>
| T1 | Advertise about new server & application using email
| T2 | Advertise about new server & application using email
| T3 | Advertise about new server & application using email
| T4 | Advertise about new server & application using email
| T5 | Advertise about new server & application using email
| T6 | Advertise about new server & application using email
| T7 | Advertise about new server & application using email
| T8 | Advertise about new server & application using email
| T9 | Advertise about new server & application using email
| T10 | Advertise about new server & application using email
| T11 | Advertise about new server & application using email
| T12 | Advertise about new server & application using email
| T13 | Advertise about new server & application using email
| T14 | Advertise about new server & application using email
| T15 | Advertise about new server & application using email
Open first GCP Region

- spin up new K8s based GCP region
Migrate Nomad to K8s

- Open two more GCP regions
- Migrate Nomad regions to K8s
Nomad to Kubernetes Migration Steps

Preparation

<table>
<thead>
<tr>
<th>ID</th>
<th>Task</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Migrate Node to Kubernetes</td>
<td>Migrate nodes to Kubernetes cluster</td>
<td>In Progress</td>
</tr>
<tr>
<td>2</td>
<td>Install Helm</td>
<td>Install Helm on Kubernetes cluster</td>
<td>Complete</td>
</tr>
<tr>
<td>3</td>
<td>Deploy Helm chart</td>
<td>Deploy Instana Helm chart</td>
<td>Complete</td>
</tr>
<tr>
<td>4</td>
<td>Update DNS records</td>
<td>Update DNS records to point to Instana Kubernetes service</td>
<td>In Progress</td>
</tr>
</tbody>
</table>

Migrate Plan

We will do the following steps to migrate from Nomad to Kubernetes.

1. Install Helm on the Kubernetes cluster.
2. Deploy Instana Helm chart on the Kubernetes cluster.
3. Update DNS records to point to the Instana Kubernetes service.
4. Verify the service is up and running.

Confidential and Proprietary Information for Instana, Inc.
GoLive blueprint

- create plan for infrastructure migration and document all steps (i.e. Miro & Google Docs)
  - **Infrastructure preparation**
    - ID, Task, Responsibility, Status
  - **GoLive steps**
  - **Go/NoGo steps**
  - **Rollback strategy**

- test all steps mentioned above in production-like environment
  - account for DNS timeouts, loadbalancer changes, Elastic IPs (communicate changes early to customers so they can prepare their network egress configuration)
  - stay away from big-bang migrations
  - automate infrastructure tests, so you can verify that new infrastructure works
  - test from various continents (servers in EU and US)

- communicate GoLive plan and gather engineers and QA that help during GoLive
- coordinate rollout with regular releases (bi-weekly @ Instana)
- Do It!
Project aftermath
What infrastructure migrations can look like - best part of 2020 so far

#sre

- **source code**: +0, -3,678
- **deployment automation**: +0, -18,361
- **loadbalancer configuration**: +3, -2,115
Infrastructure improvements

- test coverage for **instanactl**
- flexible deployments across SaaS and On-Premises releases
- networking infrastructure has been simplified
- spinning up new SaaS regions across cloud providers only takes a few days
  - before this was impossible due to VPC paring, shared datastores across regions, complex security groups, ...

Unplanned benefits of K8s migration

- Managed K8s in all regions (GCP GKE, AWS EKS, IBM OpenShift)
  - great community and tooling around K8s
  - cluster auto scaler, certificate manager, ...
Meet me in the chat lounge for Q&A