Shopify’s Path to Kubernetes

Karan Thukral
@k_thukral
Overview

- Introduction about me and Shopify
- History of Development at Shopify
- Why cloud and kubernetes?
- Guiding principles for the platform
- What we built?
- What we learned?
- Where we are headed?
Excuse me, but who are you?
What is Shopify?
Shopify in 2018

3000+ Employees

$55B+ Sales on Shopify

600,000+ Merchants
History of development @ Shopify
Development at Shopify in 2011
Development at Shopify in 2018

Shopify in 2018

- Number of Services
- Number of Developers
- Number of Deploys (Core)
What happened with this scale?
Production Environments

DC

Shopify

AWS

Heroku
Production Environments: Data Centers

- DC
- Shopify
- AWS
- Heroku
Production Environments: AWS

- DC
- Shopify
- AWS
- Heroku
Production Environments: Heroku

DC

Shopify

AWS

Heroku
Why Cloud?

- Cloud promises to be easier to scale and build PoPs around the world
Why Kubernetes?

- Existing experience with Docker
- Large community support
- Shared resources
- Platform agnostic
Why Kubernetes?

- Existing experience with Docker
- Large community support
- Shared resources
- Platform agnostic
Why Kubernetes?

- Existing experience with Docker
- Large community support
- Shared resources
- Platform agnostic
Why Kubernetes?

- Existing experience with Docker
- Large community support
- Shared resources
- Platform agnostic
Why Kubernetes?

- Existing experience with Docker
- Large community support
- Shared resources
- Platform agnostic
What principles did we build around?
Hiding Complexity
What did we build?
User Domain
(Cluster External)
srecon-eu-18-karan / staging

Owners  Edit
@Shopify/cloudplatform — Director: Camilo Lopez

Runtime
Runtime not detected 🔄

Production
Development
GitHub

Dependencies

Number of Gem dependencies: 97
Number of missing licenses: 6
Stack

- Rails (default)
- Other

Cloud Platform
- **kube-shell**: This will give you under-the-hood access to help debug failed deploys, figure out what is actually running, and get you a shell. (Automatically included for all non tier1 apps)

- **elasticsearch**: Configure an Elasticsearch instance.

- **web**: This is so that you can actually run a web server.

- **db-migrate**: So that you can run your migrations.

- **asset-uploading**: So that you can have fast assets on a CDN. Rails apps need to add the 'shopify-cloud' gem to use this.

- **jobs**: So you can run background job workers.

- **redis**: Configure a redis instance.

- **cloudsql**: So that you have a database for your models.

- **memcached**: Configure a memcached instance.

[Build runtime] [Cancel]
# srcon-eu-18-karan production

**Verifying the running version of Bundler (1.15.4) is newer than the version that created the Gemfile (1.16.0)**. We suggest you upgrade to the latest version of Bundler by running `gem install bundler`.

.Runtime errors

```
% kubectl deploy srcon-eu-18-karan-production
```

Phase 1: Initializing deploy

```
All required parameters and files are present
```

Phase 2: Checking initial resource status

```
- ConfigMap/loki-shell: not found
- ConfigMap/srcon-eu-18-karan-production: not found
- ConfigMap/srcon-eu-18-karan-production-database: not found
- ConfigMap/srcon-eu-18-karan-production-database-secret: not found
- Deployment/web: not found
- Deployment/web-ingress-controller: not found
- Deployment/web: not found
- Ingress/web: not found
- Service/web: not found
- Service/web-ingress-controller: not found
```

Phase 3: Pre-deploying priority resources

```
Deploying ConfigMap/srcon-eu-18-karan-production (timeout: 60s)
```

pong
What did we build?
Cloud Platform
(Cluster Internal)
apiVersion: stable.shopify.io/v1
kind: MySQL
metadata:
  name: "srecon-eu-18-karan-production"
  labels:
    app: "srecon-eu-18-karan"
spec:
  database_version: MYSQL_5_7
  database_flags:
    performance_schema: "on"
  data_disk_size_gb: 25
  databases:
    - name: "srecon-eu-18-karan-production"
You have a platform/product, now what?
Overview

The CloudPlatform runtime is a unified deployment stack for Shopify applications backed by Kubernetes that strives to provide an out-of-the-box experience when possible (see Project Brief).

To use this documentation:

- Browse the categories in the sidebar or enter key terms in the search bar at the top right.
- Read the introduction for a primer if you're just getting started.
- Check out the glossary for clarification on any terms. You may also mouse over terms with a hatched underline to see a quick description.
- If you have read through the docs and still need help (or have an emergency) check out the guidelines on how to get help.

Note

This documentation is intended to help make CloudPlatform self-service, without your needing to know too much about how the platform itself works. For knowledge sharing, and to better understand the platform, however, details about how things work are often provided to demystify some of the magic.

Help improve the documentation

This documentation site is powered by Jekyll, and the source code is on GitHub. If you have suggestions or find an error, then please see the README.
Teach
Challenges during our migration

• Giving up control over the infrastructure
• Learning curve from running mostly in DCs
• Migration of services took significant engineering effort
• Educating developers at Shopify about the platform
Challenges during our migration

• Giving up control over the infrastructure
• Learning curve from running mostly in DCs
• Migration of services took significant engineering effort
• Educating developers at Shopify about the platform
Challenges during our migration

• Giving up control over the infrastructure
• Learning curve from running mostly in DCs
• Migration of services took significant engineering effort
• Educating developers at Shopify about the platform
Challenges during our migration

• Giving up control over the infrastructure
• Learning curve from running mostly in DCs
• Migration of services took significant engineering effort
• Educating developers at Shopify about the platform
TL;DR

- Build for the paved path but support other use cases
- Hide away the complexity
- Build the tool to be as self serve as possible
- Document everything
- Focus on teaching people and providing support
**TL;DR**

- Build for the paved path but support other use cases
- Hide away the complexity
- Build the tool to be as self serve as possible
- Document everything
- Focus on teaching people and providing support
• Build for the paved path but support other use cases
• Hide away the complexity
• Build the tool to be as self serve as possible
• Document everything
• Focus on teaching people and providing support
TL;DR

• Build for the paved path but support other use cases
• Hide away the complexity
• Build the tool to be as self serve as possible
• Document everything
• Focus on teaching people and providing support
TL;DR

• Build for the paved path but support other use cases
• Hide away the complexity
• Build the tool to be as self serve as possible
• Document everything
• Focus on teaching people and providing support
TL;DR

• Build for the paved path but support other use cases
• Hide away the complexity
• Build the tool to be as self serve as possible
• Document everything
• Focus on teaching people and providing support
Where are we headed?
Thanks!