Building a Planet-Scale Architecture the Easy Way

Sebastian Weigand
@ThatDevopsGuy
Master plan

01 Intro
Who I am, what we're doing here.

02 The Problem
Building block basics.

03 The Solution
It's gotta run somewhere.
Introduction
About me

- “That Devops Guy”
- Sysadmin for a little over 15 years
- Cloud dude for almost a decade
- Solution specialist in NYC
- Has a Tux tattoo
The Problem
Requirements

1. **Distributed.** The system has to run microservices because that’s how you do tough stuff right.

2. **Reliable.** The system should recover if one part of it fails, preferably seamlessly to the users.

3. **Scalable.** We’ve got to support tons of users, or very few, and we want just-the-right-sizing.

4. **Simple.** Don’t want a complex system of systems to enable previous requirements.
Infrastructure
Tenets of a good distributed system

- Automated Scheduling
- Health Checking
- Various Deployment Methods
- Load Balancing
- Centralized Logging
- Centralized Monitoring
- Automated Bin packing
- Responsive Autoscaling
- Managing Secrets
- Managing Application Configuration
- Built-in Service Discovery
- Rolling back Deployments
- Responsive Autoscaling
- Accommodating Different QoS
- Stable Networking/VIPs
- Application Resiliency
- Role Based Access Control
- Primitives for Isolation
- Mounting Storage Volumes
- Long-Running v. Batch Apps
What about data?
Two big types of data

Unstructured / Blob
- Static stuff
- Data lake-type stuff
- Fast stuff

Structured / Schema
- RDBMS stuff
- OLTP stuff
- Fast, updated stuff
Keep the non-devs happy
Boring yet super-important stuff

- Proper authentication and authorization for users, developers, and operators
- Audit logging
- Centralized management of global componentry
- Cost detailing, chargeback, and optimization
- Load testing
- Distributed tracing
- Source control repositories
- Code review
- Automated builds
- Automated deployments
- Pizza delivered to the office

- Programmatic canary deployment mechanisms
- Automated rollbacks
- Beer with the pizza
- Logging and metrics storage and analytics
- Artifact storage and retrieval
- Fine-grained tagging/labeling
- Dessert after the pizza
- Autoscaling based on proper SLIs
- Automatic SLO and error budget computation
- Production debugging capabilities
- Multi-homed connectivity
- Every third Monday off, because Mondays
The Solution
Infrastructure
Tenets of a good distributed system

- Automated Scheduling
- Health Checking
- Various Deployment Methods
- Load Balancing across Containers
- Centralized Logging
- Centralized Monitoring
- Automated Bin packing
- Responsive Autoscaling
- Managing Secrets
- Managing Application Configuration
- Built-in Service Discovery
- Rolling back Deployments
- Responsive Autoscaling
- Accommodating Different QoS
- Stable Networking/VIPs
- Application Resiliency
- Role Based Access Control
- Primitives for Isolation
- Mounting Storage Volumes
- Long-Running v. Batch Apps
Enter Kubernetes

- Automated Scheduling
- Health Checking
- Various Deployment Methods
- Load Balancing across Containers
- Centralized Logging
- Centralized Monitoring
- Automated Bin packing
- Responsive Autoscaling
- Managing Secrets
- Managing Application Configuration

- Built-in Service Discovery
- Rolling back Deployments
- Responsive Autoscaling
- Accommodating Different QoS
- Stable Networking/VIPs
- Container Resiliency
- Role Based Access Control
- Primitives for Isolation
- Mounting Storage Volumes
- Long-Running v. Batch Apps
Specifically GKE...

- Automated Scheduling
- Health Checking
- Various Deployment Methods
- Load Balancing across Containers
- Centralized Logging
- Centralized Monitoring
- Automated Bin packing
- Responsive Autoscaling
- Managing Secrets
- Managing Application Configuration
- Built-in Service Discovery
- Rolling back Deployments
- Load Balancing across Containers
- Centralized Logging
- Centralized Monitoring
- Automated Bin packing
- Responsive Autoscaling
- Managing Secrets
- Managing Application Configuration
- Built-in Service Discovery
- Rolling back Deployments
Google **Kubernetes** Engine is the cheat code for Kubernetes. Thanks to my fellow Googlers I get to focus on using Kubernetes; not managing it.

*Updated, as GKE used to be called “Google Container Engine”*
Specifically GKE + Multi-Cluster Ingress

- Global load balancing
- One anycast IP
- Single A-record and you’re done
... or if you prefer ...
Traffic Director
(for them service meshes)

- Envoy configs made easy
- Centralized health checking
- Traffic-driven autoscaling
Blob Storage
Multi-Regional, 
*CDN-Fronted*
Cloud Storage

- Auto-optimizing direct reads
- Cacheable content at 96 edge locations
- Global routing
Structured Data
<table>
<thead>
<tr>
<th></th>
<th>Cloud Spanner</th>
<th>Traditional Relational</th>
<th>Traditional Non-relational</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>✓ Yes</td>
<td>✓ Yes</td>
<td>✗ No</td>
</tr>
<tr>
<td>SQL</td>
<td>✓ Yes</td>
<td>✓ Yes</td>
<td>✗ No</td>
</tr>
<tr>
<td>Consistency</td>
<td>✓ Strong</td>
<td>✓ Strong</td>
<td>✗ Eventual</td>
</tr>
<tr>
<td>Availability</td>
<td>✓ High</td>
<td>✓</td>
<td>✗ Failover</td>
</tr>
<tr>
<td>Scalability</td>
<td>✓ Horizontal</td>
<td>✓</td>
<td>✗ Vertical</td>
</tr>
<tr>
<td>Replication</td>
<td>✓ Automatic</td>
<td>✓</td>
<td>✨ Configurable</td>
</tr>
</tbody>
</table>
How is this possible?
Google Cloud Platform

134 points of presence and 13 subsea cable investments around the globe

Current regions and number of zones

Future regions and number of zones

Edge points of presence

Network
High-level architecture
Traffic Director: Cross-region failover and overflow
Conclusion

1. **Distributed.** Kubernetes runs my microservices, in multiple clusters around the planet.

2. **Reliable.** Fault-tolerant global load balancing ensures delivery of requests across the globe.

3. **Scalable.** Pods and clusters scale up and down, are right-sized, and support serverless workloads.

4. **Simple.** A single anycast IP, a few clicks in the console, and my planet-scale system is up-and-running, with little management overhead.
Don’t forget process...
How about some SRE with that DevOps?

Come find me after, I’d be happy to chat!
That’s a wrap.