Serverless on Kubernetes
(Thoughts on DB Access and Monitoring)

Santa Clara, California | April 23th – 25th, 2018
Who Are These Guys?

Bill Bauman, Head of Innovation and Strategy, Opsview
Overly passionate technology conversationalist with some (say) crazy and often contrarian ideas. Bicycles, coffee, dark beer and emerging technologies are some of the things Bill loves to discuss.

John Jainschigg, Technical Content Manager, Opsview
Developer, DevOps advocate, and long-time technology journalist, John recently joined Opsview from Mirantis, a provider of OpenStack and Kubernetes technologies, where he worked in Product and Partner Alliance marketing. He was previously Executive Director of the Internet and Community Lab (Ziff Davis Enterprise) and Online Editor in Chief of Dr. Dobb's Journal of Software Development (United Business Media) among other market-leading b2b and b2c technology titles.
Setting the Context
This is Research

• We’re not experts
Overview

• Context and terminology
• A brief history
• Introduction to OpenFaaS
• Diving in – Kubernetes, OpenFaaS, our “lab”
• Show and tell – video demo walk-thrus
  • Building functions
  • Autoscaling
  • Monitoring
• Time permitting – playing with cats on stage
The Words We Use

- **Serverless** – a computing model that abstracts infrastructure from the developer
  - You still have servers, you just don’t worry about them (or a lot less)
- **Functions** – basic building blocks of a service
- **Events** – triggers that activate a function
- **Event-driven** – resources are turned on and off, consumed as needed to service a specific event
- **Functions as a Service (FaaS)** – event-driven, serverless model: on-demand processes (often microservices) that execute a function-specific task
Serverless and FaaS Landscape

Public Cloud:
- Amazon’s AWS Lambda
- Microsoft Azure Functions
- IBM Cloud Functions
- Google Cloud Functions

On-Premises:
- OPENFAAS
- APACHE OpenWhisk™
- fission
- IronFunctions
- Kubeless
From Boxes to Containers

Physical Server ➔ Virtual Server ➔ Container ➔ Serverless
FaaS – PaaS-like or more COE-like?

PaaS-like FaaS
(often fully hosted)

Container Orchestrator-like FaaS
(generally on-premises)
Introducing OpenFaaS
Facts

- Founded by Alex Ellis, Docker Captain (blog.alexellis.io)
- Underwritten by VMware OSTC
- 90+ contributors, 11K stars on GitHub, very active community!
- Open source FaaS framework, leveraging Docker workflow and tooling
- Runs on Docker Engine, Swarm, and Kubernetes (faas-netes)
- Any container can be a function if composed with a tiny shim
- Widely-accepted standard ‘lite’ container runtimes (e.g., Alpine Python)
- Easy to learn: growing library of courses, docs, etc.
- Works really, really well
Benefits of (OpenFaaS-style) Serverless

• OpenFaaS minds itself (really!) and deploys in mere seconds
• Developers focus on code (and are more like DevOps)
  • Function writers use standard Docker workflow, manifests, composition methods
    - Developers treat FaaS as an existential plane of functions
• Operators focus on the substrate (and are more like SREs)
  • Substrate folks use whatever they like, e.g. Ansible (but kubeadm works fine)
    - Ops people treat FaaS as workload
• Exploits (but also hides) evolving container orchestration engines (COE)
  • Provides opinionated and “preferred” services and components
  • Bolsters capabilities to solve for some of orchestration’s current weak points

Makes life easy for function writers, while allowing Docker and/or Kubernetes experts great freedom and flexibility
The Adventure Begins
What we Wanted to Know

• How to install and lifecycle manage it
• What comprises a workload?
• How does a workload execute?
• How are workloads scaled?
• How can the underlying stack be monitored?
  • With Opsview Monitor 6 (Linux, Docker, Kubernetes)
• How can workloads be monitored?
  • With … OpenFaaS’ built-in Prometheus integration
• What would FaaS do to a database?
And then of course, Kubernetes ...

- Battle of the ten-minute Kubernetes recipes!
  - Bonus! This is one is by Alex Ellis!
  - Conjure-up
  - Minikube
  - Kubeadm
- Plus cloud-based “instant K8S” recipes from Hashicorp, StackPoint.io, et. al.
- Plus Kubernetes-as-a-Service
- But: Wanted on-premises
- But: Didn’t want a single-node
- But: Didn’t want to pay
- But: Didn’t want to wipe laptop hard drives
- And: Had to work across Linux and Windows
Physical Infrastructure

- Lenovo T420 – W10
- Lenovo W530 – W10
- HP Envy 740 – Ubuntu 16.04
- Netgear NightHawk r7000
Demo Infrastructure

- IP/PORTS
  - k8s-master:31112/functions
  - k8s-master:31112/ui
  - k8s-master:31112
  - k8s-master:31119
  - k8s-master:8080

- Lenovo T420 – W10
- Lenovo W530 – W10
- HP Envy 740 – Ubuntu 16.04
Show and Tell
Demo: How to Build a Function
HOW TO BUILD A FUNCTION
Demo: Autoscaling a function with OpenFaaS
OpenFaaS Default Scaling

• Simple rule:
  • If a function is invoked more than 5x/second …
    - Set the APIHighInvocationRate alert
  • Else …
    - Reset APIHighInvocationRate
  • If APIHighInvocationRate is set …
    - Launch 5 replicas per minute, to cap of 20 replicas
  • Else …
    - Delete all but one replica, immediately

• Naïve algorithm firewalls at 20 replicas per function when traffic bursts
• Alternative: Use Kubernetes Horizontal Pod Autoscaler with custom, rate-based or other scaling algorithm
HOW TO SCALE A FUNCTION
Show and Tell: Enter Monitoring
How do you Monitor This?

• Phase 1: Monitor the Stack Components
• Phase 2: Monitor the Cluster as a Business Service

• Use IT monitoring that’s comprehensive, simple, standards-based
  • … That can monitor any major app, DB, OS, or substrate (on premises or in the cloud)
  • … Using curated, open source plugins that are actively maintained
  • … Or (as needed) using a range of community-supported methods
    - Nagios NRPE Agent-based
    - Application/platform API-based
    - SNMP-based

• For us, that’s Opsview Monitor 6.0
Opsview Kubernetes Opspack (Beta)

- Configuring the Opsview Kubernetes Opspack
- Simple, just fill in the relevant values for your Kubernetes
• Configuring the Opsview MySQL Opspack

• Simple, just fill in the relevant values for your MySQL

Opsview MySQL Opspack
Demo: Autoscaling a simple function on OpenFaaS with Monitoring
MONITORING A FUNCTION
Hmmmm
(The Opsview part wasn’t very exciting, was it?)

• But wait … that’s a good thing!
• Normal IT operations should not produce alerts for non-alert conditions
• But: We can interrogate the IT monitor at will (data remains available)
• So there are things IT monitoring can tell us:
  • HTTP requests/latency
  • CPU seconds/node
  • Etc.
• We can decide if this is operationally relevant
• And we can use other tools for non-operations-relevant insight
IT Monitoring Provides Ops Insight
Impressed with OpenFaaS & K8S

• Accidents happened in the course of running these demos:
  • Children needed a computer for homework
  • … while watching reruns of Nikita on NETFLIX
  • … then closing the lid when done
  • … and don’t get me started on the cats …
• In IT terms, find servers in downed state, slept state, unknown state
• Reboot and cross fingers
• Wait … (not long – about 20 seconds)
• Linux is up, Kubernetes is up, OpenFaaS is up …
  • And functions are still running! (or, actually, restarted, awaiting new invocations)
  • Excuse me if I don’t demo this for you, but I could!
• Bonus: Opsview Monitor 6 and MySQL also come back!
  • But you wouldn’t want to try this with OpenStack in most deployments
Now Let’s Blow Things Up Look at Cats

• Function 1: Randstore. With each invocation …
  • Reads a random cat picture off a Russian cat-picture aggregation site …
    - Who knew this was a thing?
  • Converts it laboriously to escaped base64 (unnecessary, but we need the exercise)
  • Writes them into longblobs in a table in (Percona) MySQL
    - Who ever does this? Nobody, right? Discuss …

• Function 2: Randshow. With each invocation …
  • Does a select of the whole DB, ordered by random, selecting one image only
    - Seriously, who does this?
  • Unescapes the base64 and serves it as an image tag

• SystemBreaker 1.0. A javascript thing, running in a browser that …
  • Throws a little traffic at these functions in any ratio you dial in
Demo: Breaking Things (Playing with Cats)
Tweaking MySQL to avoid Redlining

- Shorten connection persistence
- Up number of connections
- Turn off the queue cache
  - Chance of cache hits approaches zero as DB size increases, so no benefit
  - Cost of cache prunes increases with cache size, so why risk incurring them?
- Implement connection-pooling? Probably not needed here (too small)
- Bonus insight: Scale functions faster or increase replicas!
- Needless to say:
  - Don’t run micro-instances of MySQL on under-resourced VMs under load
Conclusions

• Serverless/FaaS is going to grow & evolve quickly
• FaaS infrastructure is readily monitored by Opsview Monitor and potentially other monitoring tools
  • OM6 monitors the performance of the system in response to invocations; system impact, business impact
• Prometheus and other purpose-specific monitoring tools are necessary for getting deeper into specific metrics for a specific purpose, but require programming and customization to make the data interesting
  • e.g. wrote a PromQL to monitor a trigger that monitors invocations per seconds

• We’re just getting started here…

• Read our eBooks
  • The DevOps Manifesto
  • Containers as a Business Service
  • Serverless Computing
• Join our Community
• Download Opsview Monitor (free!)

Immediately database-friendly infrastructure and monitoring
Thank you!

Questions?

Bill.Bauman@Opsview.com
John.Jainschigg@Opsview.com
Rate My Session

TAP THE SESSION

Serverless Kubernetes Infrastructure Writing Data to a Percona DB: Overview and Simple Demonstration

April 24, 2:20 PM - 3:10 PM
Room M1

Tap a star to rate

Review (optional)

Submit
Thank You!