Microservices and Docker Swarm

Who am I?

Alberto Guimarães Viana

Goiano, atualmente moro em Hamburg/Alemanha

E-mail: albertogviana@gmail.com

Twitter: @albertogviana

Github: https://github.com/albertogviana

Project

- https://www.getrevue.co/profile/devops-week-news
- https://twitter.com/devopsweeknews

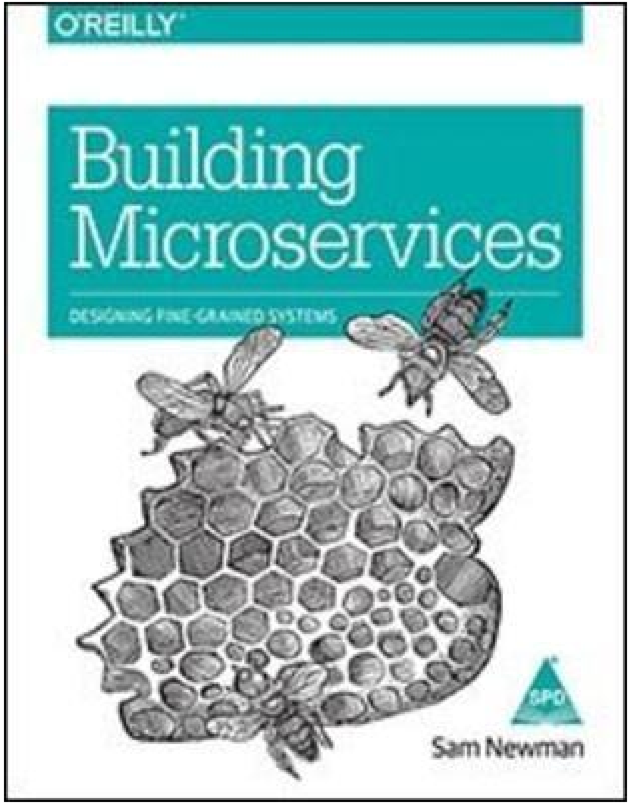
Presentation

https://github.com/albertogviana/docker-swarm-presentation

What is a microservice?

- Small, and Focused on Doing One Thing Well
- Autonomous

Building Microservices by Sam Newman



Docker

Docker ecosystem

- Docker Engine
- Docker Swarm
- Docker Registry
- Docker Machine
- Docker Compose

Docker Swarm

Swarm Features

- Highly-available, distributed store based on Raft (Service discovery)
- Cluster management integrated with Docker Engine
- Declarative service model
- Scaling
- Desired state reconciliation
- Multi-host networking
- Load balancing
- Secure by default
- Rolling updates

Service Discovery

- All other orchestration systems use a key/value store
  (k8s→etcd, nomad→consul, mesos→zookeeper, etc.)
- Swarm stores information directly in Raft
- Analogy courtesy of @aluzzardi:

It's like B-Trees and RDBMS. They are different layers, often associated. But you don't need to bring up a full SQL server when all you need is to index some data.

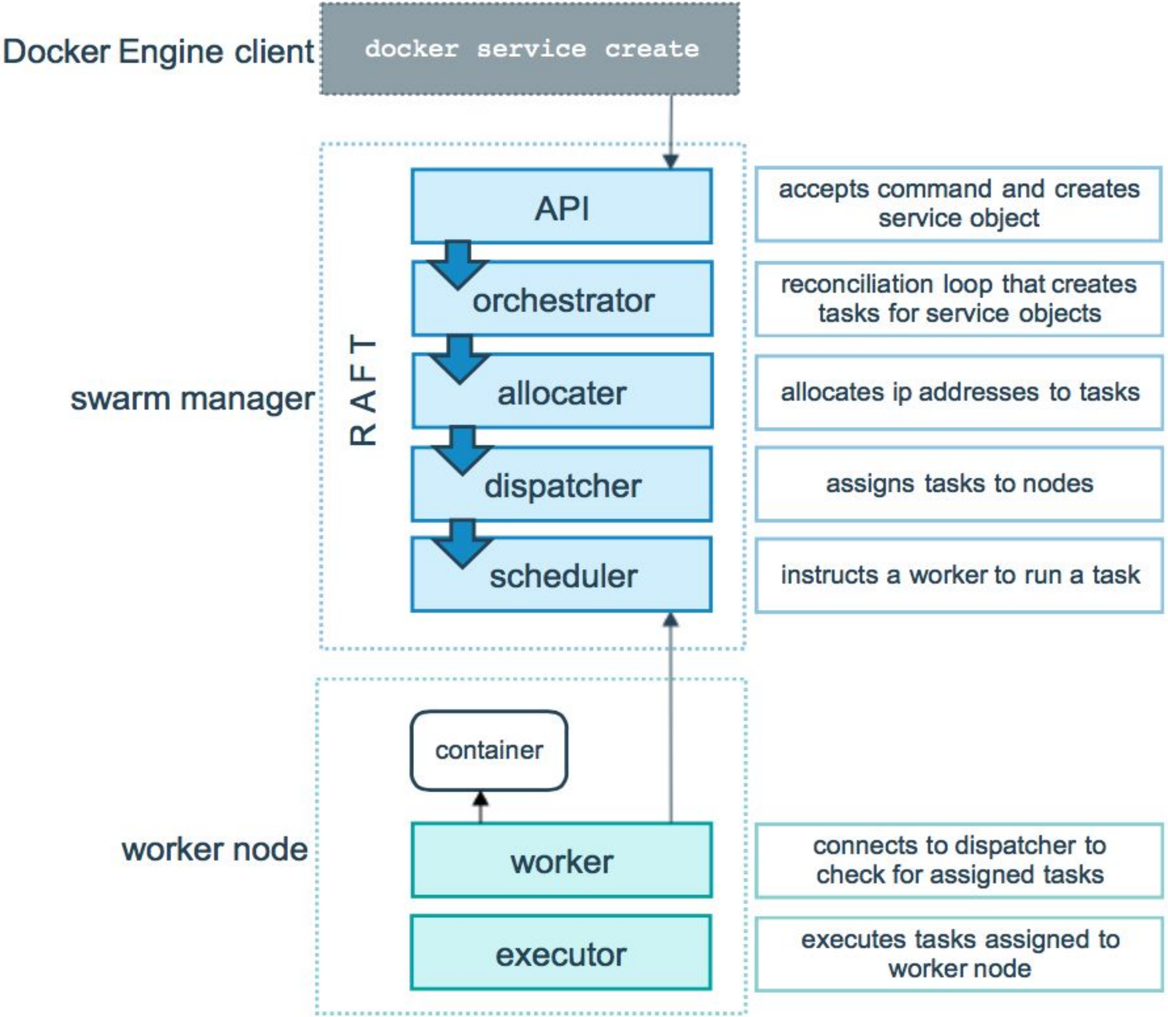
- As a result, the orchestrator has direct access to the data
  (the main copy of the data is stored in the orchestrator's memory)
- Simpler, easier to deploy and operate; also faster

Swarm concept

- A node can be a manager or a worker
- A manager actively takes part in the Raft consensus
- You can talk to a manager using the Swarm API
- One manager is elected as the leader; other managers merely forward requests to it
- Using the API, you can indicate that you want to run a service

Swarm concept

- A service is specified by its desired state: which image, how many instances…
- The leader uses different subsystems to break down services into tasks: orchestrator, scheduler, allocator, dispatcher
- A task corresponds to a specific container, assigned to a specific node
- Nodes know which tasks should be running, and will start or stop containers accordingly (through the Docker Engine API)



accepts command and creates service object

reconciliation loop that creates tasks for service objects

allocates ip addresses to tasks

assigns tasks to nodes

instructs a worker to run a task

connects to dispatcher to check for assigned tasks

executes tasks assigned to worker node

Swarm Mode

- Docker Engine 1.12 features SwarmKit integration
- The Docker CLI features three new commands:
  - *docker swarm (enable Swarm mode; join a Swarm; adjust cluster parameters)*
  - *docker node (view nodes; promote/demote managers; manage nodes)*
  - *docker service (create and manage services)*

Build swarm cluster

Visualizing the cluster

Adding nodes to the cluster

Security

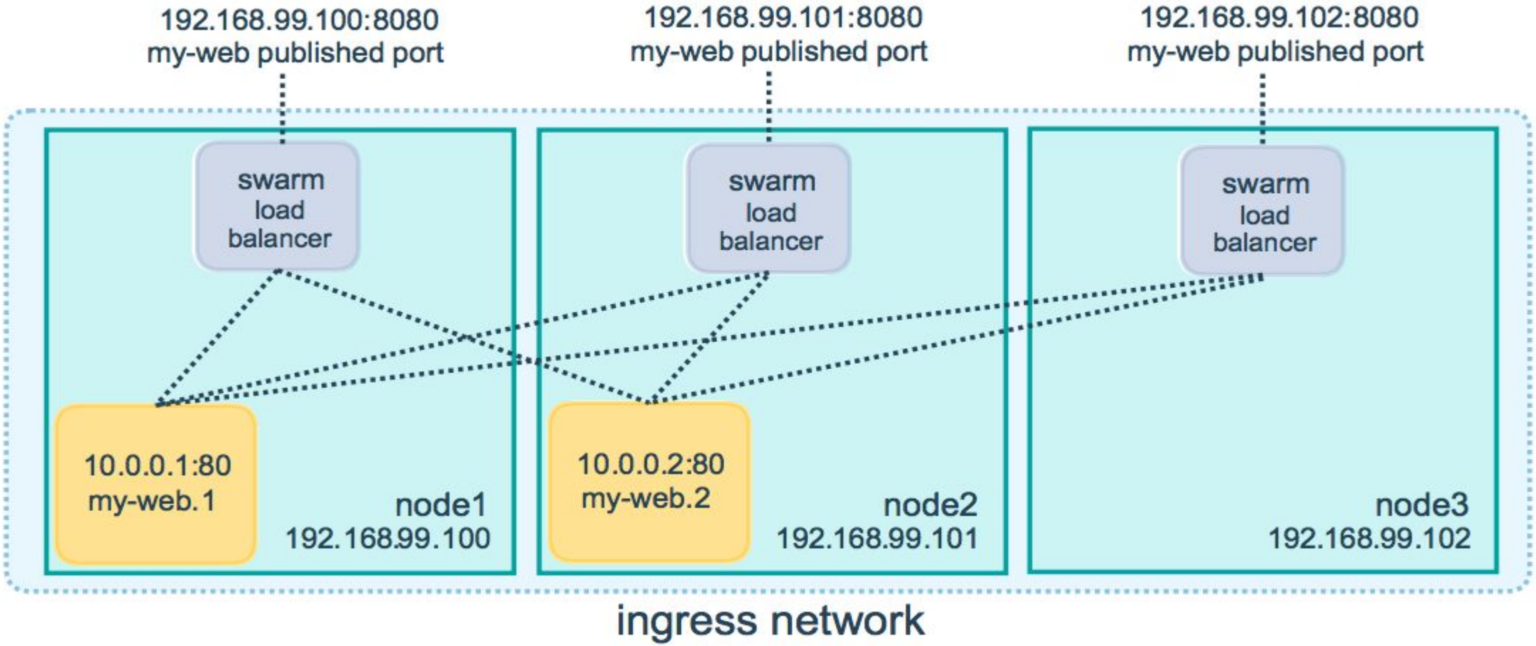
Under the hood

- When we do docker swarm init, a TLS root CA is created. Then a keypair is issued for the first node, and signed by the root CA.
- When further nodes join the Swarm, they are issued their own keypair, signed by the root CA, and they also receive the root CA public key and certificate.
- All communication is encrypted over TLS.
- The node keys and certificates are automatically renewed on regular intervals (by default, 90 days; this is tunable with docker swarm update).

Docker Service

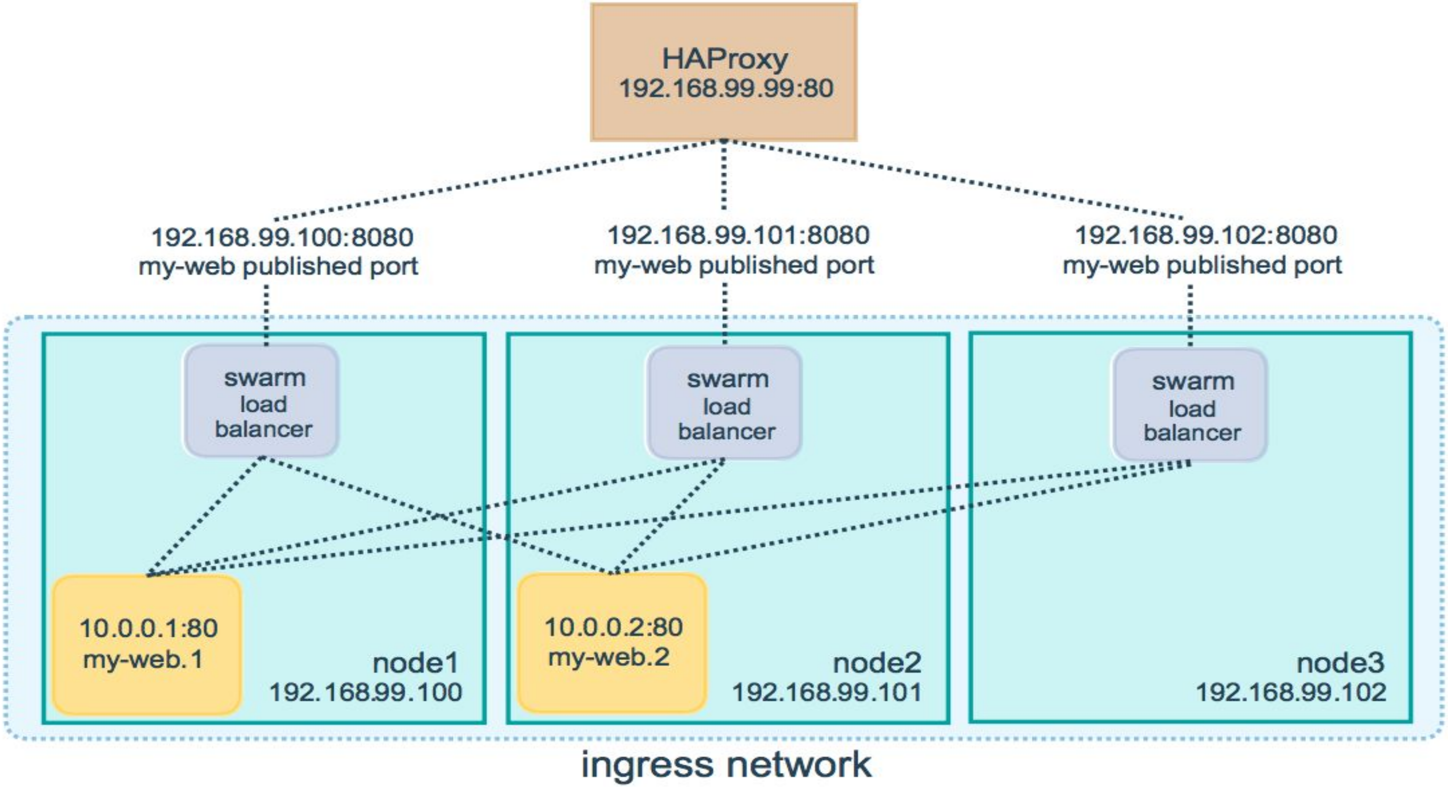
Routing Mesh

Routing Mesh



Scaling a service

Reverse Proxy



Dockerfile

Rolling Updates

Docker secret

Docker Node

Docker system?

What now?

Metrics

- Prometheus
- cAdvisor
- InfluxDB

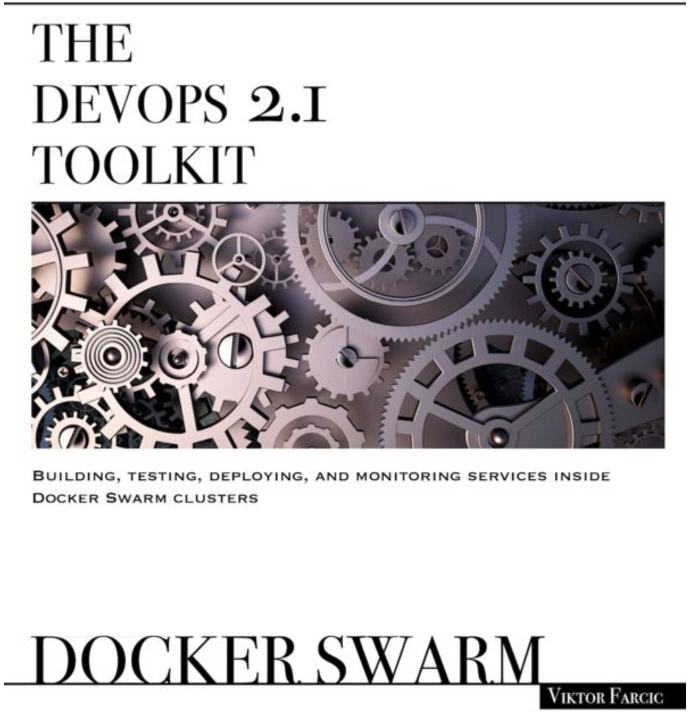
Logging

- ElasticSearch, Kibana and Logstash (ELK)
- FluentD
- Loggly
- Graylog
- gliderlabs/logspout

Dashboards

- Grafana (https://grafana.net/dashboards/609)

Book



THE
DEVOPS 2.1
TOOLKIT

BUILDING, TESTING, DEPLOYING, AND MONITORING SERVICES INSIDE
DOCKER SWARM CLUSTERS

DOCKER SWARM

VIKTOR FARCIC

Questions?

E-mail: albertogviana@gmail.com

Thank you! Obrigado!