ANTONIS KALIPETIS - @AKALIPETIS

I CREATED MY SWARM NOW WHAT?

@AKALIPETIS

- Docker Captain and early user
- Python lover and developer
- Technology lead at SourceLair / stolos.io
- Docker training and consulting

I love automating stuff and sharing knowledge around all things containers, DevOps and optimizing developer workflows.

Photo by rawpixel.com on Unsplash

Photo by Gabriel Matula on Unsplash

WHAT THE HECK IS HAPPENING?

UNDERSTAND YOUR SWARM

- Find your way into your Swarm
  - List services, tasks and get a grasp of what is running and where
- See overall usage metrics
  - Host used RAM, CPU, disk, etc
- Learn a bit about RAFT,
  - This will help you in bad situations where not everything goes as planned

MONITORING

- Services run in random places
- High level server monitoring is not enough
- Without metrics, you're blind

MONITORING

- Prometheus is the cool kid in the neighbourhood
  - ...but there is a huge amount of alternative monitoring tools
- stefanprodan/swarmprom will take you up and running quickly
  - ...but you need to customize it to your needs later on
- When basic metrics are good to go, it's time for your own /metrics
  - ...to start adding custom metrics for your services

Photo by Jehyun Sung on Unsplash

AUTOMATE
ALL THE THINGS!

CONTINUOUS INTEGRATION

- Containerize your whole CI pipelines
  - ...to run your tests in the exact same environment
- Create a branching strategy
  - ...and hook deployments with merge events
- Make sure your deployments are triggered by your CI system
  - ...since software is better than humans, ¯_(ツ)_/¯

KEEP DEPLOYMENTS LOGGED

- Most CI systems have a way of marking deployments, use it!
- Easily rollback to previous versions, through the CI
- Deploy small changes often

Photo by Michał Parzuchowski on Unsplash

PLAN AHEAD!

SIZING

- Sizing your containers is one of the most important part of the process
  - ...luckily, we have metrics!
- Find out the base resource usage
  - ...and reserve it
- Observe possible peaks
  - ...and limit them

CLUSTER SIZING

- Find a machine size the fits your containers nicely
- Rule of thumb: if you're up to 7 servers, make them all managers/workers
  - ...then switch to 3 managers + the number of workers needed

Photo by Rick Mason on Unsplash

GET ALL THE PIECES RIGHT

NETWORK SEPARATION

- Make sure that critical services are in separate networks
- Start with a frontend network for user-facing services
- Create networks for each stack
- Use purpose-driven networks, like monitoring

SECURITY

- Use secrets for everything
- Use --autolock to encrypt manager data at rest
- In bigger clusters, don't run payload in managers

HEALTH CHECKS

- Make sure your application are healthy, not just running
- Avoid warm-up time failures
- Make rolling update to actually roll

Photo by Stanford Smith on Unsplash

SERVICE YOUR USERS

HTTP(S) ENTRYPOINT

- Add an external TCP load-balancer, from 80, 443 to 80, 443 of every worker
- Add a proxy and TLS termination service at the edge of your cluster
  - sourcelair/ceryx is a great choice, with LE certs, dynamic routes, based on NGINX
- Use Swarm's internal load-balancing to reach your services

SCALING

- Use your metrics to scale your services
  - ...unfortunately, there's no one solution that works for everyone here
- Use cluster capacity to scale the cluster
- Always keep a safe amount capacity available, to easily and quickly scale services

ANTONIS KALIPETIS - @AKALIPETIS

THANKS