THE HOTELS NETWORK

Resource intensive jobs: when AWS lambda performs too well

This is THN: Growth for the direct channel of hotels

THE HOTELS NETWORK

An ecosystem of
growth tools for hotels

We are a distributed platform for the hotel industry

15,000 hotels

Several million daily visits

Over 8PB of data processed monthly

Intel (my team)

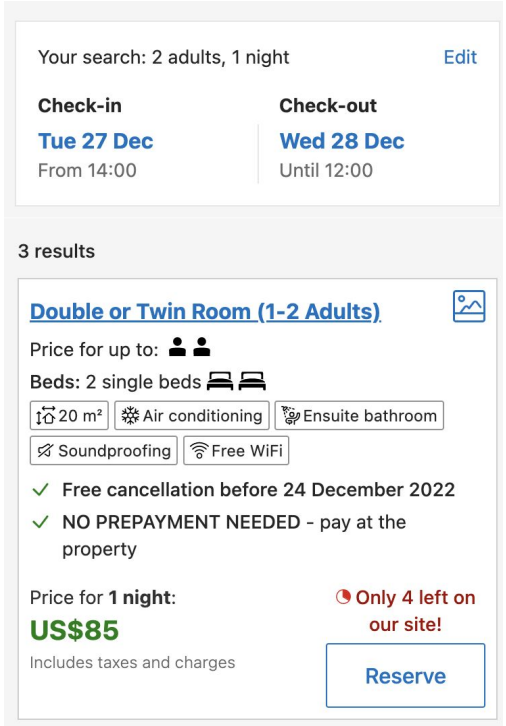
- We enrich data (real time or not) a lot. In order to provide value, we want the richest data possible. This is not always possible with the data we capture from the hotel's website.

- What type of data do we enrich? Room names & description, prices, availability, available discounts…

- We are building a highly concurrent pipeline to grab, process and normalize data, on the top of an event-based set of microservices written in Go and Node.JS backed by AWS.

One need: taking photos of disparities

- Sometimes prices in sources and aggregators do not follow the hotelier guidelines
- In that case we take a photo and save it for later hotelier usage



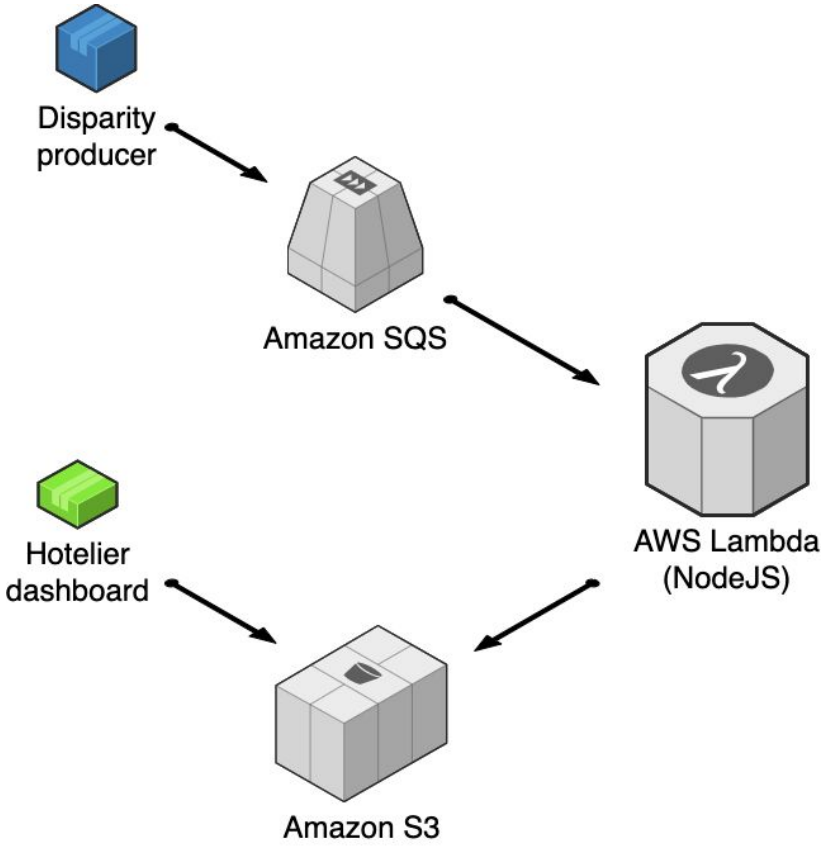
High-level design

1. Another microservice captures the disparity and produces a message with all the information needed.
2. The consumer reads the event and sends it to the internal processor.
3. The processor launches a browser and navigates to the original site.
4. The processor does, if needed, some tweaks into the DOM.
5. The processor takes a screenshot and uploads it into a remote disk.

Numbers

- >30 sources to take screenshots
- ~300K RPD → 250 RPM
- ~16s to process one screenshot
- ~6s delay to process screenshots

Initial approach: AWS lambda to rescue



AWS Lambda

FaaS: serverless computing

Well integrated in AWS ecosystem

Multiple SDKs, including NodeJS

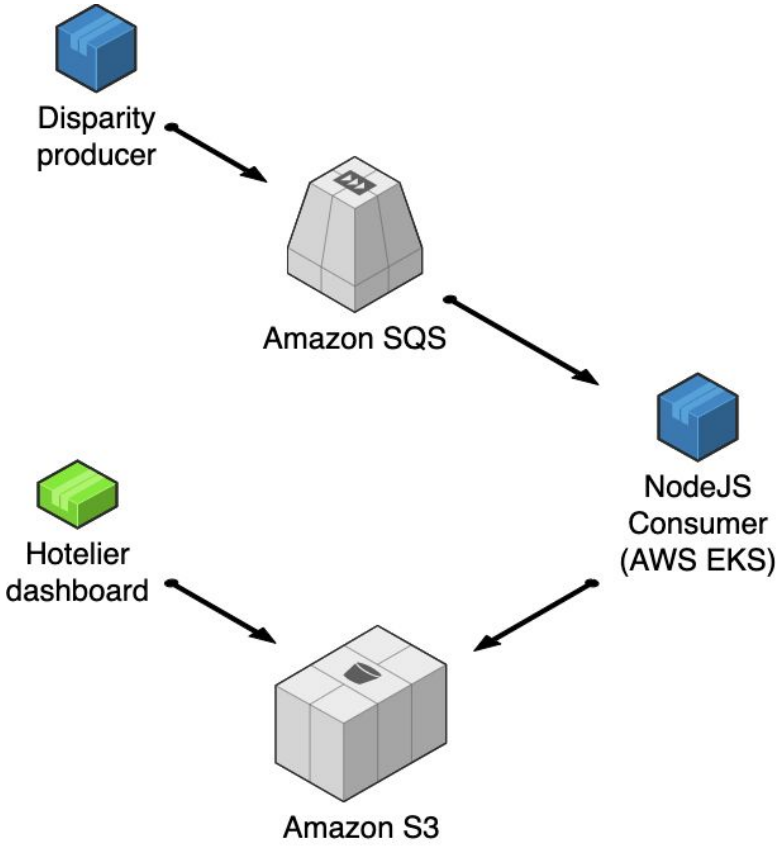
Pricing: per allocation, per usage

Problems found

Analysis

- Tons of requests
- High resources usage:
  - Time running
  - Memory
  - CPU
- Retries

Reengineering with AWS EKS



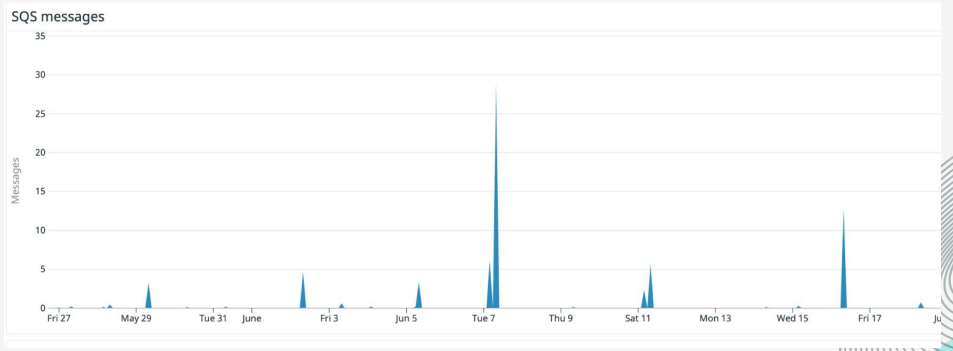
Problems found

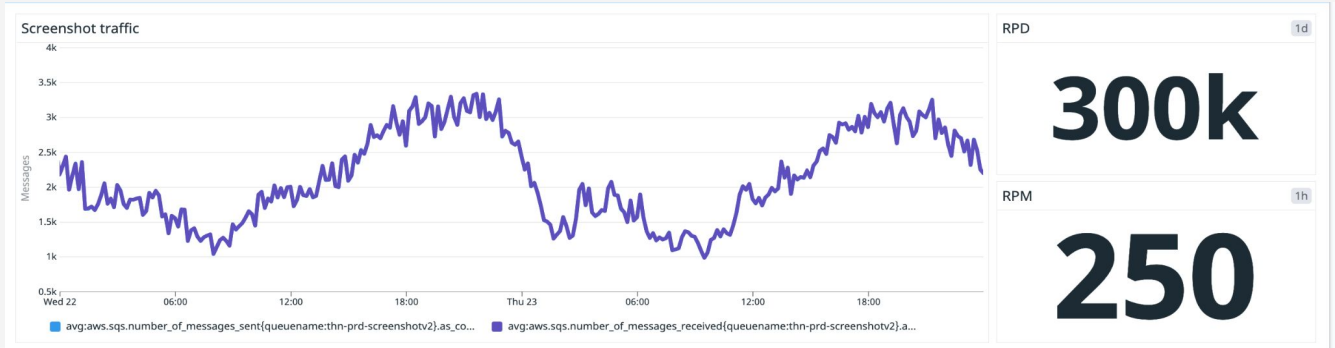
Analysis

- Stability: crashing after just some hours
- Resiliency: any failure lead to crash the POD
- Performance: slow, opening a new tab in every request
- Scalability: did not handle peaks, static POD number
- High cost: needed to oversize the service

Figure out what's going on: instrumentation

- Adding and tagging logs
- Measures: AWS built-in, custom
- Know your service!





Solutions found

- Stability: check the code to release resources, close connections, etc.
- Resiliency: error handlers to catch unexpected errors
- Performance: pool of tabs up & running to handle requests
- Scalability: HPA setup & adding a circuit breaker
- Cost-specific improvements:
  - Chromium fine tuning
  - Screenshot fine tuning
  - AWS spot instances (much cheaper than regular instances)


DEMO
38
BRACE YOURSELVES
imgflip.com

Conclusions

- AWS lambda is powerful…
- …but it is not a swiss knife
- Nice for developing quickly: PoC, MVPs
- If you are running high-intensive workloads consider OpenFaaS (or simply k8s services)

Thank you!

David Torres Garrigós

 dt@thehotelsnetwork.com

 @datoga

 https://github.com/datoga

 https://linkedin.com/datoga

THE HOTELS NETWORK

THE HOTELS NETWORK

www.thehotelsnetwork.com

Headquarters in Barcelona with a team around the world

Athens · Austin · Bangkok · Barcelona · Bogotá · Buenos Aires · Frankfurt · Hoi An · Hong Kong · Istanbul

Jakarta · Los Angeles · Manila · Mexico City · Miami · New York · Paris · San Francisco · Singapore · Vancouver