Telling the story of WeissBeerger Cloud Evolution

Eitan Sela - System Architect

WeissBeerger

A part of the ABInBev Family

eitan.sela@weissbeerger.com

$ whoami

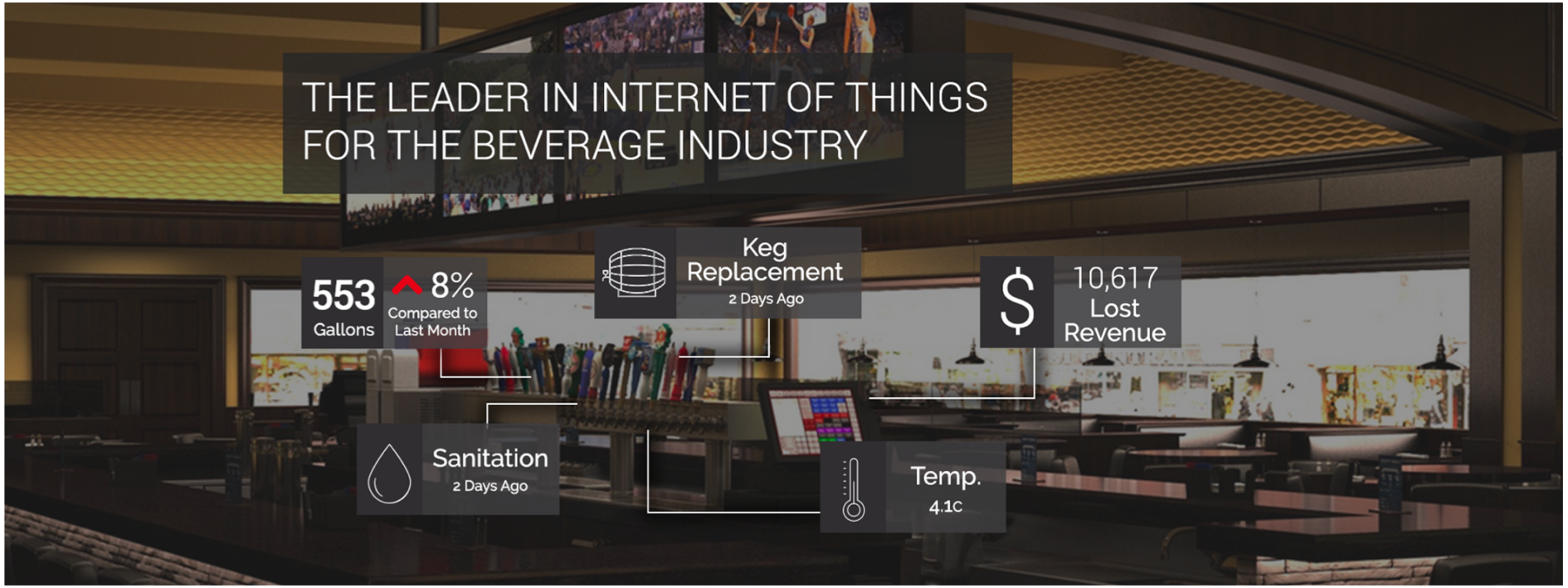
- "Hands-On" system Architect with more than 18 years of experience with billing, banking, information security (DLP) and Cloud IoT/Big Data applications.
- Big Data specialist – Hadoop, Spark, Hive and EMR on AWS.
- Work with vast AWS services, and with serverless projects especially.
- Java development, scalability performance and stabilization expert.
- Alexa skills developer.
- Love to share my experience in lectures and meetups.

What to expect from this session

- WeissBeerger – Company overview & use case.
- Architecture - August 2017.
- What were the problems we faced back then?
- Cloud Evolution – Building the puzzle.
- Architecture - December 2018.
- Future Challenges.

WeissBeerger use case

- WeissBeerger bridges the gap between breweries, bars and customers.



How does it work?

- IoT (Pouring) – Beverage Analytics Hub.
- Point of sales – POS Vendors, via REST API, S3, DB, etc.

WeissBeerger Architecture - August 2017

Web Dashboard and Mobile App Architecture – August 2017

Monolith PHP

Amazon EC2

But it is actually a huge monolith EC2! And all Apps are stateful!

Monolith PHP
Dashboard and Mobile App (stateful)

Monolith Java
Point of sale App (stateful)

Amazon EC2

Data science Algo
(stateful)

MongoDB
For Data science Model

AWS Usage – August 2017

- EC2
- Route 53

What were the problems we faced back then?

Problems with the architecture

- One monolithic EC2 – **No high availability & redundancy, manual security updates.**
- Apps are Monolithic and Stateful – **No reuse & modularity, Can not be scaled!**
  - IoT HTTP Gateway - Developed in C++, stateful and un-maintainable.
  - POS – Developed in pure Java, with custom thread handling, in Spring Boot.
- Apps are updated manually in the server with git pull – **No automation - CI/CD**.

Problems with the architecture – con't

- Security:
  - Development and Production EC2 are on the same account.
  - No dedicated security groups and vpc's.
  - Only one subnet open to the internet.
- No Monitoring:
  - Developers are reactive (vs. proactive).
  - No one knows if something fails in production.

The main problem: Poor performance of Dashboard and Mobile App

Cloud Evolution – Building the puzzle

Re-Architecture project – June 2017

Principle	Concepts
Security	• Network & Infra Security • Secure software implementation practices • Penetration testing • Applicative security using roles and permissions model
Robust, Scalable & Performant	• Ability to scale out to meet demand • APIs, Services & Applications built to perform on Day 1
High Availability & Redundancy	• Multiple Servers in Multiple Physical Locations • Reduce single points of failure • Automated backup and restore • Designed for fast failover
Be Proactive vs Reactive	• Reactive –> Proactive → Predictive • Monitoring – Infra, System & Application • Logging
Automation vs Manual	• Automate for efficiency and less human involvement (human error)
Reuse & Modularity	• Design for multiple use cases • Adopt open-source • Build modules and not a monolith
Simple over complex solutions	• Always opt for straightforward approach • Reduce “moving parts” when possible

Foundation of WB Architecture & DevOps Team

Mike Margolis | Chief Architect

Eitan Sela | Systems Architect

Reuven Kaplan | Systems Architect

Elran Regev | DevOps / DBA

July 2017

Stas Spivak | Systems Architect

Tal Klinger | IoT Architect

Leonid Kopylov | IT & Devops


WINTER IS COMING
@青红造了个白
LET'S START
RE-ARCHITECHTURE PROJECT
imgflip.com

Security

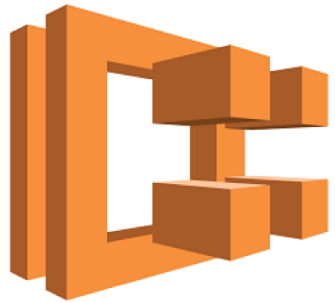
- Added additional auditing in AWS infrastructure (Cloudtrail)
- Removed users that don’t absolutely need access and removing unused keys (AWS & DB)
- Moved to strict AWS IAM user/role management over systems.
- Integrated with SSO provider (OneLogin)
- Moved production environments to separate, isolated AWS account from Dev, Stage environments, and established additional DR account.
- Added additional Security Penetration testing processes to our software development practices
- Require full VPN access (highly restricted) to access any VPC resources.
- All AWS credentials in developers PC are obtained with OneLogin/STS, and expire after 12 hours.
- In later stages – GuardDuty, WAF, AWS Config, and much more…

CI/CD

- We provisioned Jenkins server to trigger automatic builds in Dev, manual in Prod.
- Java jars were now copied to the server and service restarted, without manually developers doing so.
- In later stages – Build and deploy containerized applications to ECS.
- In later stages – Use Jenkins jobs to enable triggering Lambda functions or long running ETLs by “clicking a button”.
- **Today we have more than 150 Jenkins jobs.**

Containerization and micro services

- We started by splitting pos-engine, monolithic app, to three Spring Boot micro-services and deploy them to ECS.
- Move Analytics (data science) code in Python to be containerized.
  - All Linux cron jobs were converted to ECS Scheduled Tasks.
- In the following months, dozens of containerized micro-services will be developed and deployed to ECS.
- Use Fargate - For running POC Monthly reports once a month.
- **Today we have more than 50 Tasks Definitions and more than 100 containers deployed & running in production.**



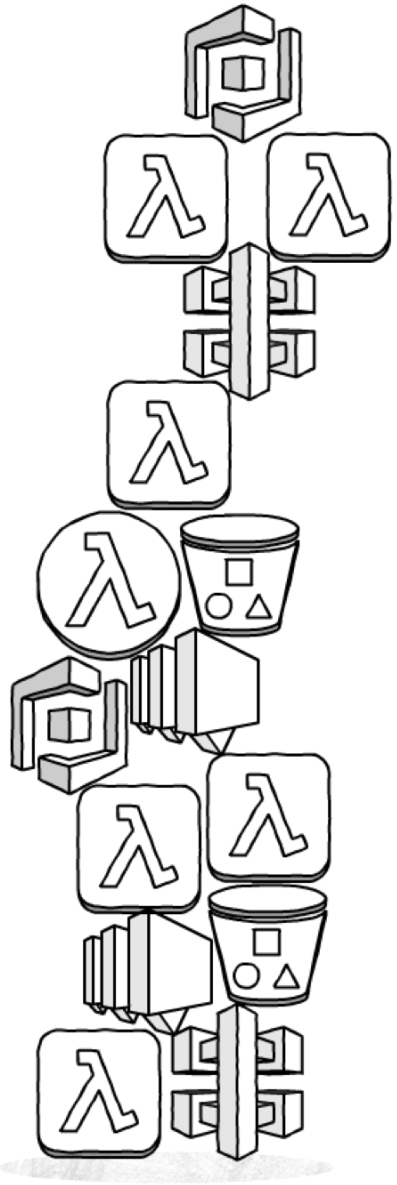
Redshift as Managed DWH and development of ETLs

- We started using Redshift for our DWH:
  - Optimized for Business Intelligence (BI).
  - Columnar Storage for heavy analytics and aggregative queries.
- For our ETLs from MySQL to Redshift – Used PDI (Pentaho data integration) deployed as containerized Spring Boot app.
- **Result: Improved Dashboard and Mobile App query performance by several fold, and make improved user experience!**

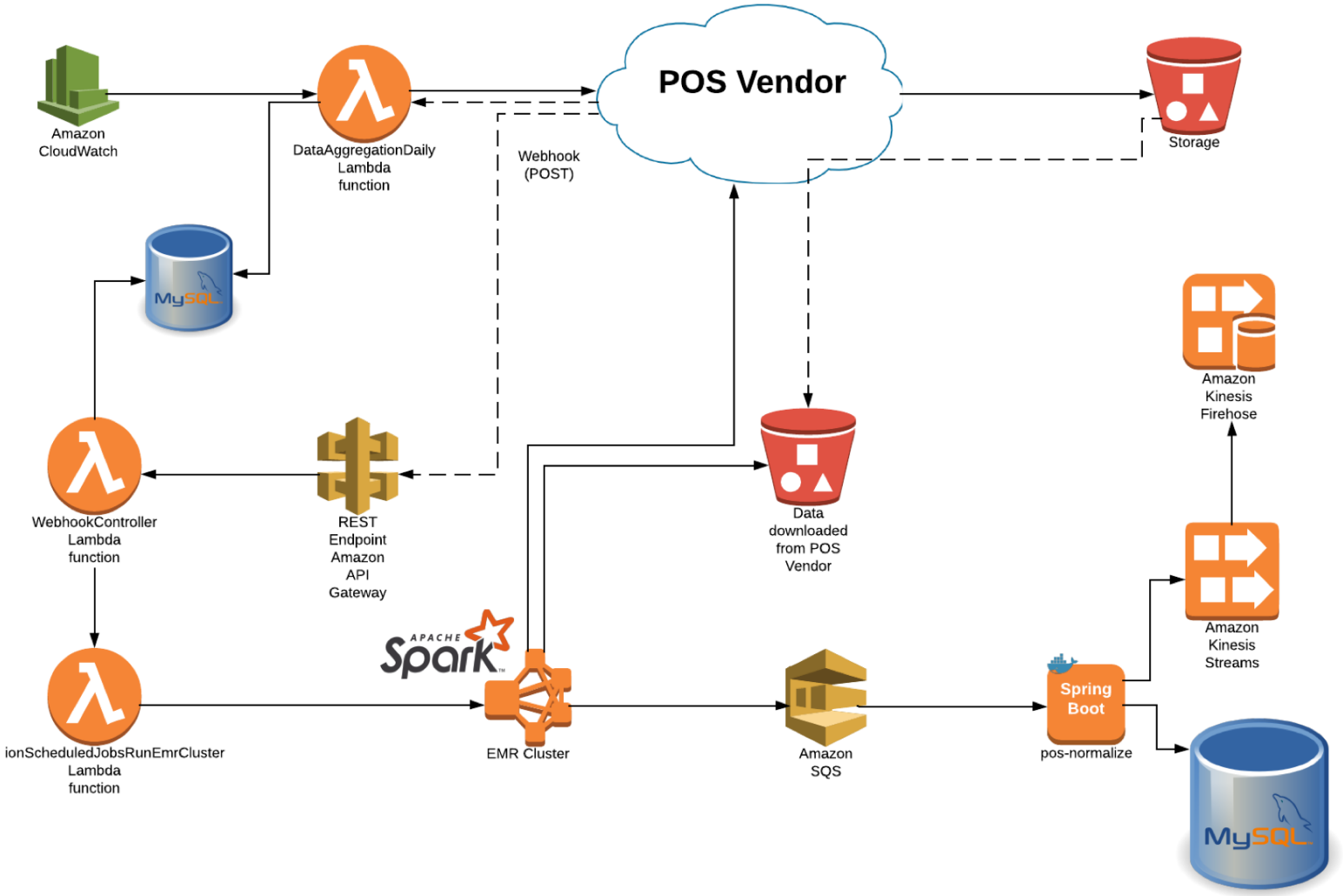
amazon REDSHIFT

Serverless - Start small, grow tall

- We use Lambda Functions extensively.
- Mostly with the CloudWatch Rule as trigger (cron like), but also with other triggers – S3, SQS, Kinesis, API Gateway, Lambda calling other Lambda and more.
- For Lambda CI/CD we use serverless framework.
- Lambdas are very easy to deploy – Just create *serverless.yml* and Jenkins job with few lines of shell script.
- For Lambda tracing, we use AWS X-Ray.
- **Today we have more than 90 Lambda Functions in production**.

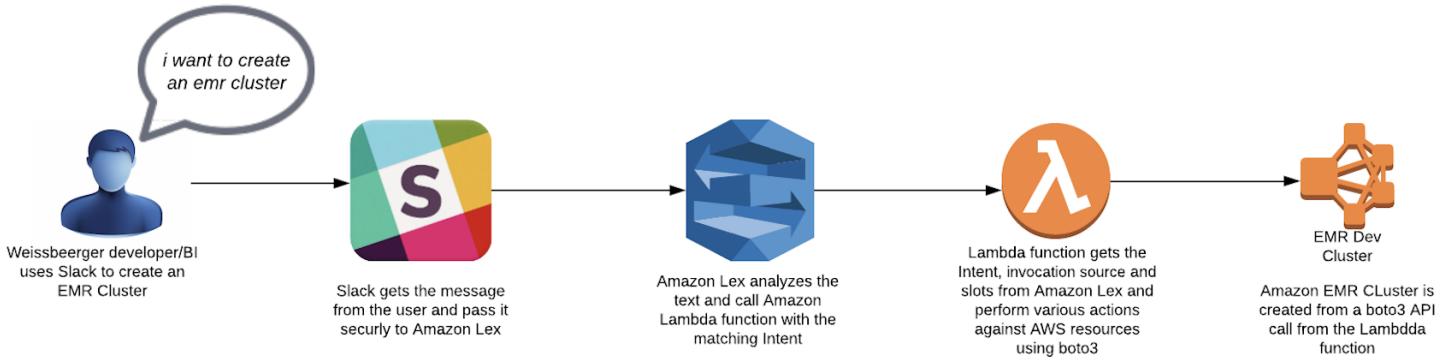


Batch processing pipeline using Lambda Functions, CloudWatch and API Gateway



AI & Machine learning

- From our home made algorithms, trained and deployed on EC2, and stored on MongoDB, to fully managed machine learning service using Amazon SageMaker.
- SageMaker Batch Transform Feature was developed as direct request from WB.
- We save 120 man-hours a month, provisioning EMR Clusters, using our “aws chatbot” built using Slack, Amazon Lex and Lambda functions.



Implementing Big Data pipeline with EMR

- **This effort started only on February 2018.**
- There were two problems:
  - ETLs were fetching data directly from MySQL, which caused 100% CPU, slave lags and eventually, scaling up more and more, to enable normal work, thus exceeding budget.
  - BI devs had to wait sometimes hours, for heavy queries on MySQL to complete.
- After implementing full pipeline with Spark jobs ELTs that fetch data from Hive on S3:
  - ETLs (now ELTs) take minutes to complete. No stress on MySQL.
  - BI query on un-imagined amount of data, and wait usually less than 15 minutes.

Implementing Big Data pipeline with EMR – con't

- We use transient EMR cluster and S3 as our Data Lake – Decoupling of storage from compute and data processing.
- We trained our BI to develop PySpark jobs. Our POS Java developers felt more comfortable developing Spark jobs in Scala, so we have also Scala.
- **Today we have more than 100 PySpark/Scala jobs and more than 40 developers, BI and Data engineers that work with the set of applications that are shipped with EMR.**

EMR Clusters provisioned via WeissBeerger ”aws chatbot” - A typical day

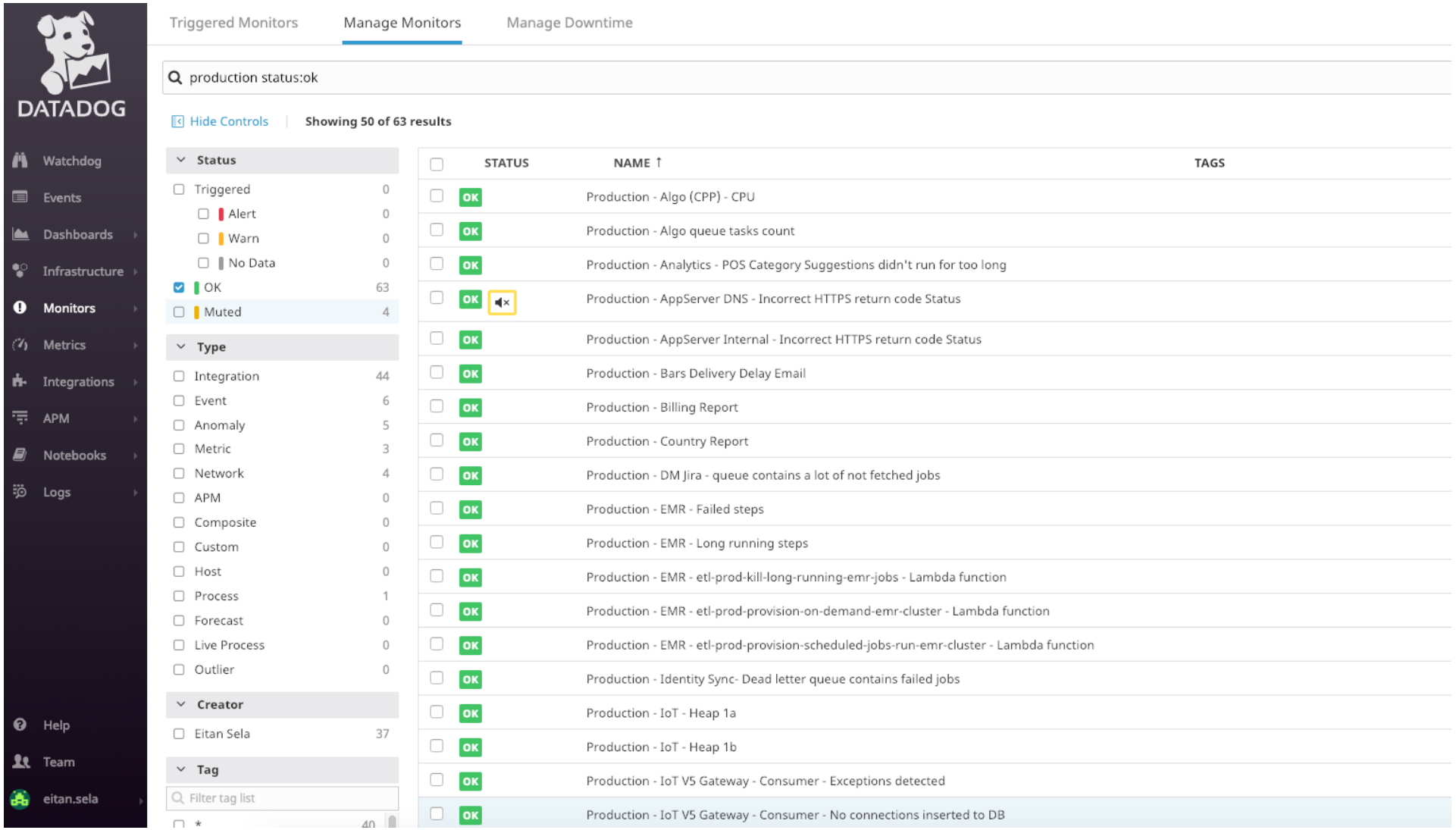
Filter: Active clusters | Filter clusters ... | 17 clusters (all loaded)

Name	ID	Status	Creation time (UTC+2)	Elapsed time
on-demand-dev-emr-pavel-cluster	j-2JI5CZ25KFBH1	Starting	2018-12-19 16:02 (UTC+2)	52 seconds
scheduled-jobs-run-emr-stg-hourly-pos-iot-all-jobs	j-23SVXQ4UYMMZZ	Running	2018-12-19 14:50 (UTC+2)	1 hour, 13 minutes
on-demand-dev-emr-pavel-cluster	j-3UJMT455MW2B9	Terminating User request	2018-12-19 14:43 (UTC+2)	1 hour, 20 minutes
on-demand-dev-emr-matanbiron-cluster	j-20OPRRAINRPAJ	Waiting Cluster ready	2018-12-19 13:35 (UTC+2)	2 hours, 28 minutes
on-demand-dev-emr-lekha.b-cluster	j-16NYBI3D11VFK	Waiting Cluster ready	2018-12-19 13:32 (UTC+2)	2 hours, 30 minutes
on-demand-dev-emr-avishai.ophel-cluster	j-23AC3931M38NI	Waiting Cluster ready	2018-12-19 13:21 (UTC+2)	2 hours, 42 minutes
on-demand-dev-emr-matan.damary-cluster	j-26HBPO30ASV8J	Waiting Cluster ready	2018-12-19 10:51 (UTC+2)	5 hours, 11 minutes
on-demand-hive-prod-replica-dev-emr-assaf.kazakov-cluster	j-MKECDOCMZ8KK	Waiting Cluster ready	2018-12-19 10:17 (UTC+2)	5 hours, 45 minutes
on-demand-hive-prod-replica-dev-emr-tal.barak-cluster	j-14F97HPY6KUFQ	Waiting Cluster ready	2018-12-19 10:15 (UTC+2)	5 hours, 48 minutes
on-demand-hive-prod-replica-dev-emr-gal.geva-cluster	j-3TBMON7B16BGV	Waiting Cluster ready	2018-12-19 09:56 (UTC+2)	6 hours, 6 minutes
on-demand-dev-emr-adi.shemesh-cluster	j-2271AWF4O1L27	Waiting Cluster ready	2018-12-19 09:41 (UTC+2)	6 hours, 22 minutes
on-demand-dev-emr-michal-cluster	j-114LNW1IJQSEE	Waiting Cluster ready	2018-12-19 09:11 (UTC+2)	6 hours, 52 minutes
on-demand-dev-emr-hadar_oren-cluster	j-MOBWF2F7CN6R	Waiting Cluster ready	2018-12-19 08:50 (UTC+2)	7 hours, 13 minutes
on-demand-dev-emr-amit-cluster	j-32GNFMV4GI8HW	Waiting Cluster ready	2018-12-19 08:32 (UTC+2)	7 hours, 31 minutes
on-demand-dev-emr-nevo.ezov-cluster	j-I53QQF2MZASJ	Waiting Cluster ready	2018-12-19 07:52 (UTC+2)	8 hours, 11 minutes
on-demand-dev-emr-eitan.sela-cluster	j-21PFRR0QPL5D0	Waiting Cluster ready	2018-12-19 07:26 (UTC+2)	8 hours, 37 minutes

Monitoring with Datadog

- All our monitoring is done in Datadog.
- We monitor:
  - Resource metrics - Physical components as CPU, memory, disks, and network interfaces.
  - Work metrics – Top-level health of the system - Throughput, Success/Error and Performance.
- Monitoring our containers deployed in ECS is done with dd-agent as side-car.

Monitoring Everything – From heap size of Java apps to failed EMR steps

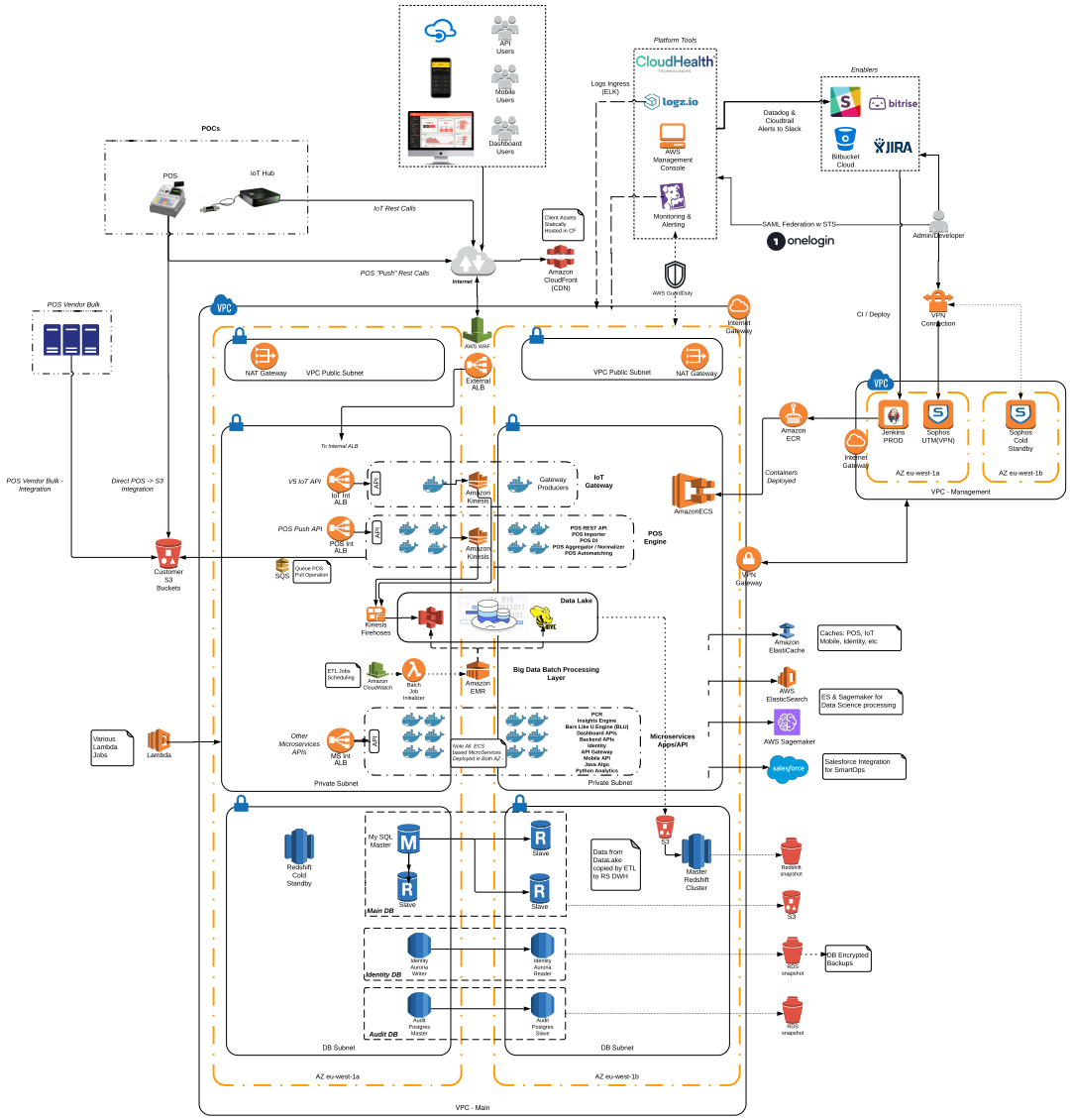


WeissBeerger Architecture - December 2018

WeissBeerger
A part of the AB InBev Family

Weissbeerger AWS Architecture

BEVERAGE ANALYTICS

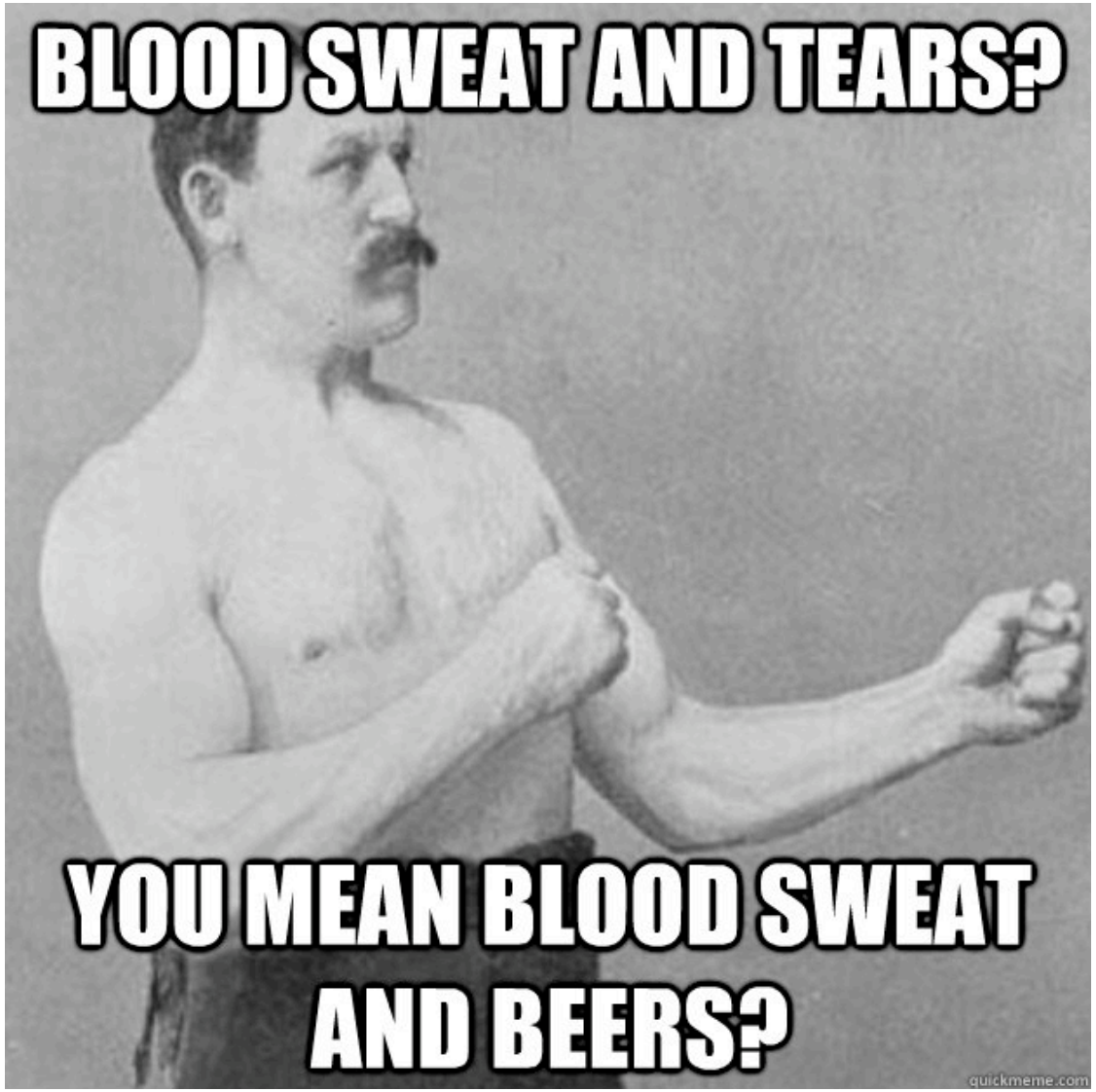


AWS Usage – December 2018

- EC2
- Route 53
- ECS
- Fargate
- RDS
- Aurora
- SQS
- SNS
- Lambda
- CloudWatch
- CloudFront
- EMR
- Kinesis
- IoT Button
- SageMaker
- S3
- Glacier
- API Gateway
- GuardDuty
- WAF
- ALB
- Redshift
- DMS
- ElatstiCache
- Config
- X-Ray
- Lex
- WorkSpaces
- DynamoDB
- Textract – Future
- Timestream – Future
- EKS - Future

How did we achieved all that?

- Commitment and hard work of architecture team.
- We were given mandate to do whatever we thought will help the company, considering costs and budget limits.
- Rapid adaptation of new technologies amongst R&D, DS and BI teams.



The outcome – December 2017

יצרנית הבירה הגדולה בעולם רכשה את וויסבירגר מת"א

חברת הסטארט-אפ נמכרה בסכום מוערך של 80 מיליון דולר ▪ וויסבירגר מספקת מערכות אנליטיקס מבוססות אינטרנט של הדברים ליצרני משקאות, מסעדות וברים

נתי יפת 10:08 ,26/01/2018

א+ א-

תחרות נשיאת בירה למרחק בגרמניה / צילום: Michael Dalder, רויטרס

חברת הסטארט-אפ וויסבירגר מתל-אביב, המספקת מערכות אנליטיקס מבוססות אינטרנט של הדברים ליצרני משקאות, מסעדות וברים, נמכרה בסכום מוערך של 80 מיליון דולר.

הרוכשת היא אנהאוזר-בוש אינבוו (Anheuser-Busch InBev), יצרנית הבירה הגדולה בעולם, המחזיקה כ-400 מותגי בירה, בהם בקס, באדווייזר, סטלה, הוגרדן ולף.

Industry with Heritage

AB InBev 600 years

Carlsberg 169 years

Heineken 152 years

Global beer consumption

187,000,000,000

Liters consumed per year

50,000,000

Outlets serving beverage

ABInBev

WeissBeerger

Future Challenges

- Move to fully IaC - Infrastructure as a code.
- Implement development & ops oriented culture change / Train our developers to gain further knowledge of AWS and how to leverage it.
- Improvements and costs savings on various Big Data services we use – implement AirFlow and start using Athena, automated cost analysis & remediation.
- Develop more for serverless.
- Move all our ML algorithms to SageMaker and develop new algos based on SageMaker pre made ones.
- Development of new self installed IoT hubs that uses AWS IoT and Greengrass.

We Are Hiring!

- Senior Big Data Engineer.
- Senior Full Stack Developer.
- Data QA Engineer.
- Data Management Specialist.
- Head Of Commercial Analytics.
- Data Integrity Team Leader.
- Senior Product Manager.

eitan.sela@weissbeerger.com

https://beverageanalytics.io/about-us/

Q & A