Andrea Medda

github.com/andream16

Email : me@andreamedda.com

Mobile : +447366115386

Experience

- **Cloudflare** — London, United Kingdom
  Systems Engineer — *Oct 2020 - Current*
  - **Logpush for Audit Logs**: Big customers pay a lot of attention to their audit logs. Unfortunately they were forced to build and manage jobs to poll this information from our audit logs APIs because there was not better way for them to ingest their own audit logs data. This led to issues like network calls failures, rate limiting and inconsistencies. We then decided to leverage Logpush for audit logs to seamlessly push this data in our customers' cloud storage service. I designed, implemented and deployed this integration with 0 down time. We decided to leverage audit logs events being emitted to *Kafka* that are usually used to ingest these events to: understand if these were valid and correctly stored, fetch & transform them and finally push them to our Logpush integration. There were some challenges as our SLO is 15 minutes but we were consistently lagging behind as the throughput was about 50 writes per second but our we were able to process 10 per second. We then introduced distributed tracing via *opentracing* and profiling with *pprof* to find bottlenecks and increase our consumption rate. After making improvements such as consuming & processing in batches, we were able to reach 10k reads per second.
  - **Audit Logs Ingestion**: Designed Cloudflare's Audit Logs new ingestion and validation system to enhance producers' onboarding experience and audit logs quality with 0 down time. The legacy system ingested audit log events in *JSON* format via *Kafka* and provided a *Golang* SDK to produce such events. Producing these events directly to *Kafka* caused issues where the lack of synchronous validation led to poor audit logs quality. Moreover, producers leveraging other programming languages had to maintain their own SDK. The new system deployed a *GRPC* server in front of the topic which allowed for strong validation checks leveraging *Protobuf* and *Buf* as well as generating ready to go client side code for producers in the programming languages that were needed. Now producers can onboard by just enriching the audit logs request contract and importing the generated code in their service - *Kafka* knowledge is no longer required. This system also enabled for giving producers immediate feedback when a request was invalid as well as metrics that could be checked & is able to easily handle 1 million daily requests and potentially scale further leveraging *Kubernetes' HPA*. Client-Server authentication leverages *mTLS* for a seamless producer experience.
  - **Anomaly Detection System**: Designed and implemented anomaly detection for Firewall Events to notify our customers when they are under attack. When customers are under attack, we collect metrics and store them in *Clickhouse* and then we can later on access this information using ABR. We usually store information such as amount of firewall events registered per website over time. We built a system leveraging *Kafka*, *Golang* and *Groupcache* that is able to schedule anomaly detection jobs for all websites that have this notification enabled. When a job gets scheduled, it pulls & process this raw information. It then uses a bucketed z-score algorithm to understand if there were anomalies in the past detection window (5 minutes). This works well for us but we're currently designing improvements for the detection pipeline to also perform regression checks as well as allowing customers to define their own filters our dataset.
  - **Observability & Performance Tuning**: Introduced several concepts and tools to improve our microservices' observability and performances. Before we could not deeply understand certain bottlenecks in our *Kafka* pipelines. I introduced *opentracing* and *pprof* to diagnose these issues in production by implementing some easy to use SDKs and instrumenting our services. Introduced some other SDKs for *Kubernetes* health checks and connection pooling for *Postgres*. Introduced batch processing concepts in my team and extended our *Kafka* SDK to easily leverage it. Thanks to these improvements, we were able to increase our confidence when releasing as well as our pipelines' throughput.
  - **Developer Experience Improvements**: Introduced a *Golang* microservices' instrumentation framework inspired by uber-go/fx to standardise how our microservices' are written and can be monitored. This lightweight frameworks allows developers to instrument their service with tracing, logging, metrics, health checks (and more) out of the box. It also allows to serve long lived components like http servers or *Kafka* consumer groups reliably and also handles their full life cycles gracefully. This framework is slowly gaining popularity and traction as our DevEx team is looking to push for it.
  - **Oncall Wizard**: Part of the EMEA oncall rotation for our team. I'm able to triage alerts, take actions to resolve them or escalate them as well as making sure that they won't happen anymore. Introduced several improvements for long lived alerts like slow processing of *Kafka* topics or fatal errors. Led postmortem discussions for incidents and helped other teams with higher priority incidents.
- **Curve OS** — London, United Kingdom
  Software Engineer — *Sep 2018 - Sep 2020*

- **Payments Leader**: During my last months in Curve, I worked on the payments domain as a technical team lead. I was responsible for end to end technical delivery, mentorship and high performance of a small team of backend engineers. During this time, my team successfully contributed to the legacy payments system by introducing new features and worked on critical fixes that led to considerable savings of more than £60.000 a month. I worked very closely to my team and improved the workflow and morale.
- **Wirecard Gate**: I was one of the key contributors to the weekend integration with a new acquirer due to the Wirecard Gate. The integration was very challenging as most of the legacy payments processing system depended on Wirecard as well as the existing funding cards and processes around them. I was tasked with tokenising the existing funding cards with the new partner, which was essential to make payments work. This has been the most critical situation that Curve has ever lived as Curve depends on such partners to operate payments.
- **Go Chapter Lead & Gophercon Lead**: I was the Lead of the *Go Chapter*, a bi-weekly meeting that enables go engineers to share ideas, demonstrate and proposing the adoption of new standards. I led *Curve*'s sponsorship of *Gophercon UK 2019*, the largest and most important *Golang* conference in UK.
- **Curve Samsung Pay Card**: Key Contributor to the integration with Samsung Pay to launch the Curve Samsung Pay Card. I proposed and won key stakeholder approval for my scalable and resilient architecture design, built and deployed new services in *Golang* as well as make changes to optimize the existing *Golang* and *PHP* services.
- **Microservices migration**: I helped the company migrate from 3 monolithic *PHP Symfony* applications to 150 *Golang* microservices deployed on *Kubernetes* using *Istio*, *Ambassador* and technologies like *Jenkins*, *CircleCI*, *MongoDB*, *PostgreSQL*, *RabbitMQ* and *GRPC*. Promoted clean code and testing culture (TDD) by introducing standards to the *Golang* services and writing high quality, well tested and maintainable production code as an example to more junior engineers.
- **User Onboarding Improvements**: Led the Improvement of customer onboarding and Curve's fraud and compliance system by re-building the legacy process for identifying possible fraudsters and politically exposed people. I Worked on and deployed a new event driven architecture with 0 down time to the production environment. The new system, built in *Golang* and utilizing *RabbitMQ* for resilience, increased the process pass rate from 40% to 60% as well as decreasing false positives. The new services enables faster changes in the future by decoupling from the monolith.

- **Easy Network s.r.l.** Cagliari, Italy
  Software Engineer *May 2016 - Sep 2018*
  - **Merchant Payment Application**: During my time at *Easy Network*, I worked on *MistralPay*, which provides end-to-end processing for merchant payments. As part of the team, I maintained the legacy platform built in *Django* and *PostgreSQL*. As there was a need to speed up development time as well as make the application more resilient, I drove the introduction of cloud solutions. This work consisted of moving the database from on-prem to *AWS RDS*, moving assets to *AWS S3* and deploying the app to an *AWS EC2* instance with a focus on autoscaling.
  - **Rebuilt Customer Onboarding**: I worked on designing, building and deploying a new event driven onboarding using *AWS Cognito* issued *Javascript Web Tokens*, *Go*, *NodeJS* and *Python* microservices deployed onto *AWS Lambda*, with data stored in *DynamoDB*. All the resources were created and deployed using *Terraform* and I even built out a new frontend using *Angular*.
  - **Cumlaude21.it**: *Cumlaude21.it* is a social marketing platform used to manage a entrepreneur's work/social network. As the company was very small at the time, I inherited this project when the main maintainer left. The project consisted of a front-end built using a custom *Javascript* framework and of a *Java Spring Boot* back-end that persisted data in a *PostgreSQL* database. As maintainer, I worked on the introduction of new features, as well as on bug fixing, introducing tests and documentation so other colleagues could assist in future.
  - **Price Tracking Service**: I worked on a platform that tracked *amazon.co.uk* product prices over time. The idea was to suggest user when was the best time to buy an item based on their price being lower than usual. I was the sole maintainer of a front-end built in *AngularJS* and contributed to a *NodeJS* back-end. The data acquired during this project gave me an idea for a side university project, *price probe*.

- **Xorovo s.r.l.** Sestu, Italy
  Internship *Mar 2015 - Jun 2015*
  - **University News App**: During my internship, I built a app for my university news feed. As this was my very first working experience, I was taught the basics of software development and web applications. The app was built using *Apache Cordova*, *AngularJS*, *HTML*, *CSS3* and *Bootstrap*. The news feed was populated with news in *JSON* format fetched from a *Apache2* web server. In order to render these news, I implemented a logic layer in the app to call the web server using *HTTP* calls and used the *MVC* pattern to show the results.

Education

- **Università degli Studi di Cagliari** — Cagliari, Italy
  Master of Science in Computer Science: 110/110 (1st Class) — *Oct 2015 – Feb 2018*
- **Università degli Studi di Cagliari** — Cagliari, Italy
  Bachelor of Science in Computer Science — *Oct. 2012 – Jul. 2015*

Projects & Talks

- **Price Probe**: During my MSc, I worked on price probe, a project used to predict *amazon.co.uk* prices based on their price history and other external factors like a product manufacturer's popularity and customer reviews over time. The idea was to understand if *amazon.co.uk* prices were influenced by external factors and not only by the market. I built a series of web crawlers using *Golang* and *Python* to collect the data. After crawling and data post-processing, I focused on forecasting the prices for each product. I built a *Machine Learning* model using *Python*, *Pandas* library and *Apache Spark*. For each product, I made a prediction using the basic time series and trying all the different combinations like *(price, date, popularity)*. The results were impressive. I discovered that the manufacturer's popularity influenced price trends and was able to predict a product's price for the next several days. The results and the study can be viewed in the paper *Forecasting E-Commerce Products Prices by Combining an Autoregressive Integrated Moving Average (ARIMA) Model and Google Trends Data* that was published on Future Internet. The source code is available on my github.
- **Go Microservices at Curve Talk**: On July 2019, I gave a talk about *Go microservices*, their project structure and testing best practices adopted at *Curve*. I gave the talk in Cagliari, the city where I grew up, during a *Go meetup*, a local *Golang* group that I found inspired by the UK one. The talk is available on my github.

Programming Skills & Interests

- **Languages**: Go, PHP, Javascript, Typescript, Java and Python.
- **Technologies**: Docker, Kubernetes, Kafka, RabbitMQ, GRPC, Protobuf, PostgreSQL, MongoDB, Prometheus, Grafana, PProf, Opentracing, Structured logging, AWS, Angular and Redux.
- **Interests**: Distributed Systems, TDD, Clean Code practices, Event Driven Architectures, Observability & Monitoring, Algorithms and Data Structures.