QBurst

Migrating
Core Business

APPLICATIONS TO THE CLOUD

Project Overview

Scalability and availability were two key drivers that contributed to the decision of migrating their core business application to the cloud. The existing computing infrastructure lacked the flexibility and scalability to respond to a surge in volume. Traffic spikes were often managed manually due to the seasonal nature of business. Underutilization of computing resources during off season was a challenge that could only be addressed with a sound cloud strategy.

Client Profile

A leading winter services and ground maintenance company based in Europe.

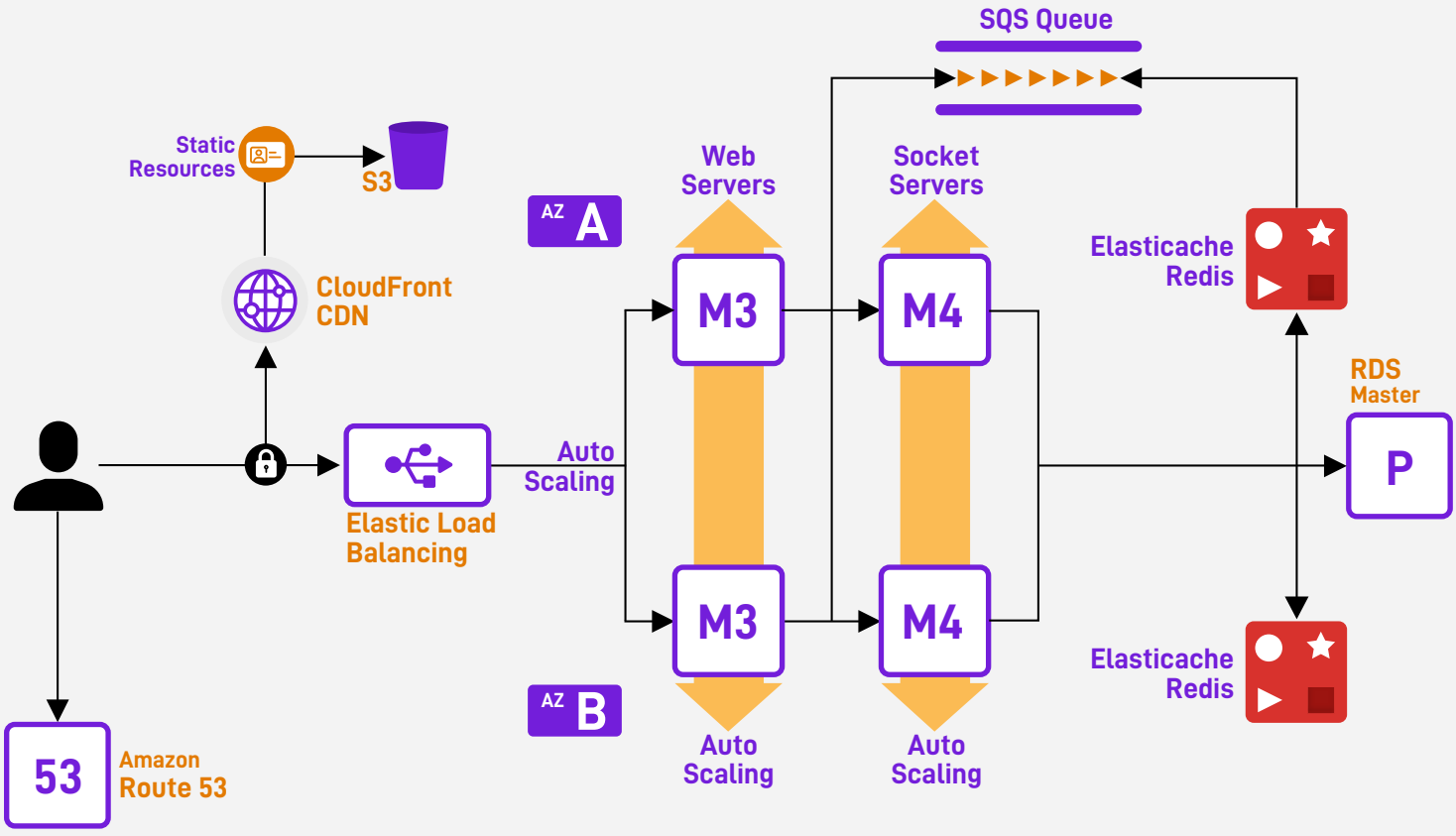
Limitations of the Existing Infrastructure

The application is used by field workers, supervisors, and managers. Disruptions in service often led to a ripple effect, where operators and managers were unable to complete tasks on time for customers.

- Inability to manage high traffic during peak season
- High overhead costs associated with maintaining server infrastructure
- Only certain parts of the infrastructure was elastic
- Servers could not be scaled vertically
- Servers are managed manually — spikes in traffic require human intervention
- Critical servers such as load balancers and databases are prone to downtime in the event of failure
- No redundancy in case of catastrophic data center failure

Solution

It was decided to migrate the existing workloads from the on-premises environment to AWS. This would lead to improvement in productivity and significant savings in IT cost. Based on the requirements, we proposed separate architectures for development, staging, and production environments. Separate AWS accounts for development, staging, and production ensures proper bifurcation and reduces the blast radius in case of outages/issues.



Highlights

- Load Balancer: Public-facing Application Load Balancer. EC2 instances were split across availability zones to ensure high availability.
- Autoscaling: Automatic scaling was enabled to deal with sudden spikes in load.
- Socket Driver Servers: Socket driver servers were placed under an auto-scaling group to ensure horizontal scaling and high availability. Two m4.large EC2 instances were used.
- Queue Service: Amazon's Simple Queue Service (SQS) was used to offer standard and FIFO queue. The existing queueing system was re-architected to work like SQS.
- Cache: A 2-node AWS ElastiCache with Redis cluster was used because of its in-memory caching capabilities, high availability, and speed.
- Database: AWS RDS service with MySQL.
- CDN: For content delivery, we used Amazon CloudFront which offers edge caching and faster content delivery.
- DNS Service: AWS Route53 DNS service.

Benefits

- 45% reduction in cost due to the efficiencies offered by AWS
- Time required to deploy infrastructure resources reduced from weeks to minutes
- The ability to scale up and down quickly and manage multiple workloads with ease saved a significant amount of time and effort