A Dynamic and Cost-Efficient Multi-Region Object Storage System Using Erasure Coding

Marcell Feher
Chocolate Cloud ApS

John Byabazaire
University College Dublin

January 22nd, 2025, Barcelona

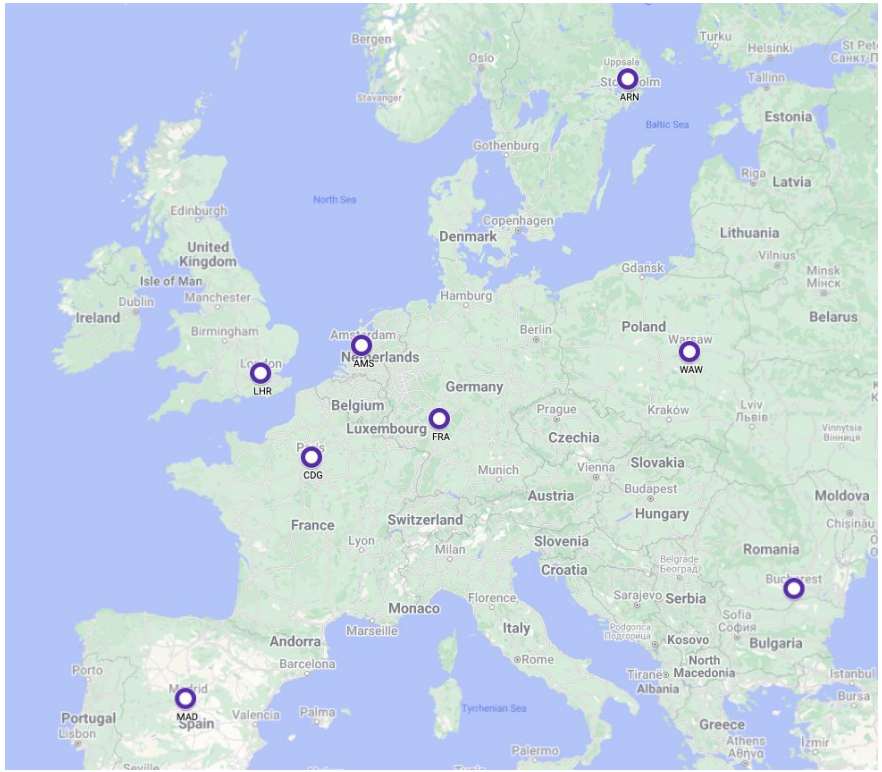
Object Storage Overview

- One of the key technologies in cloud computing
- Purpose: store unstructured data for long term
  - Images, videos, 3D models, documents, AI models, etc.
  - Basically anything that does not fit into a database
- Objects (name+contents+metadata) organized in Buckets (containers)
- Pioneered by Amazon Web Services with S3 in 2006
  - Became the industry standard
  - Several companies offer S3-compatible object storage service
  - Including us: Chocolate Cloud from Denmark

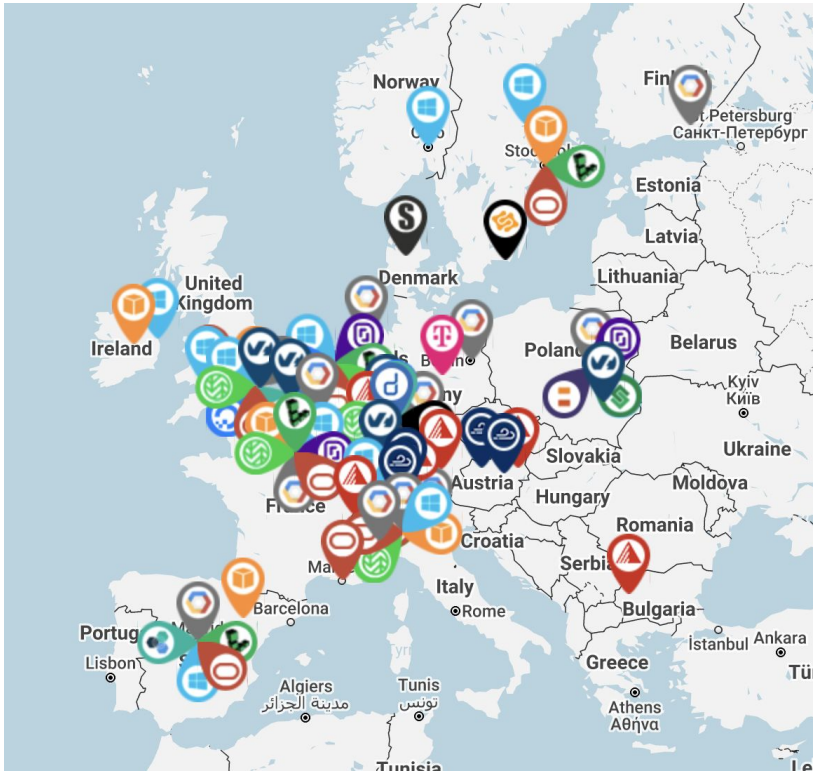
Multi-Region Object Storage

- Limitation we are addressing: single fixed bucket location
  - AWS S3 buckets are stored in one user-selected region forever
  - Low latency only close to that region
  - Replication to a different bucket possible, but cumbersome
  - CDN needed for large area coverage, extra config & cost
- Our approach
  - Objects are erasure-coded with redundancy (e.g., 4+2)
  - Data fragments distributed to various storage locations
  - Gateways provide S3 API and perform erasure coding (+encryption, etc)
    - Single URL, user requests routed to the closest one
  - Fragment locations and redundancy follows real demand

Gateways: 8 in EU



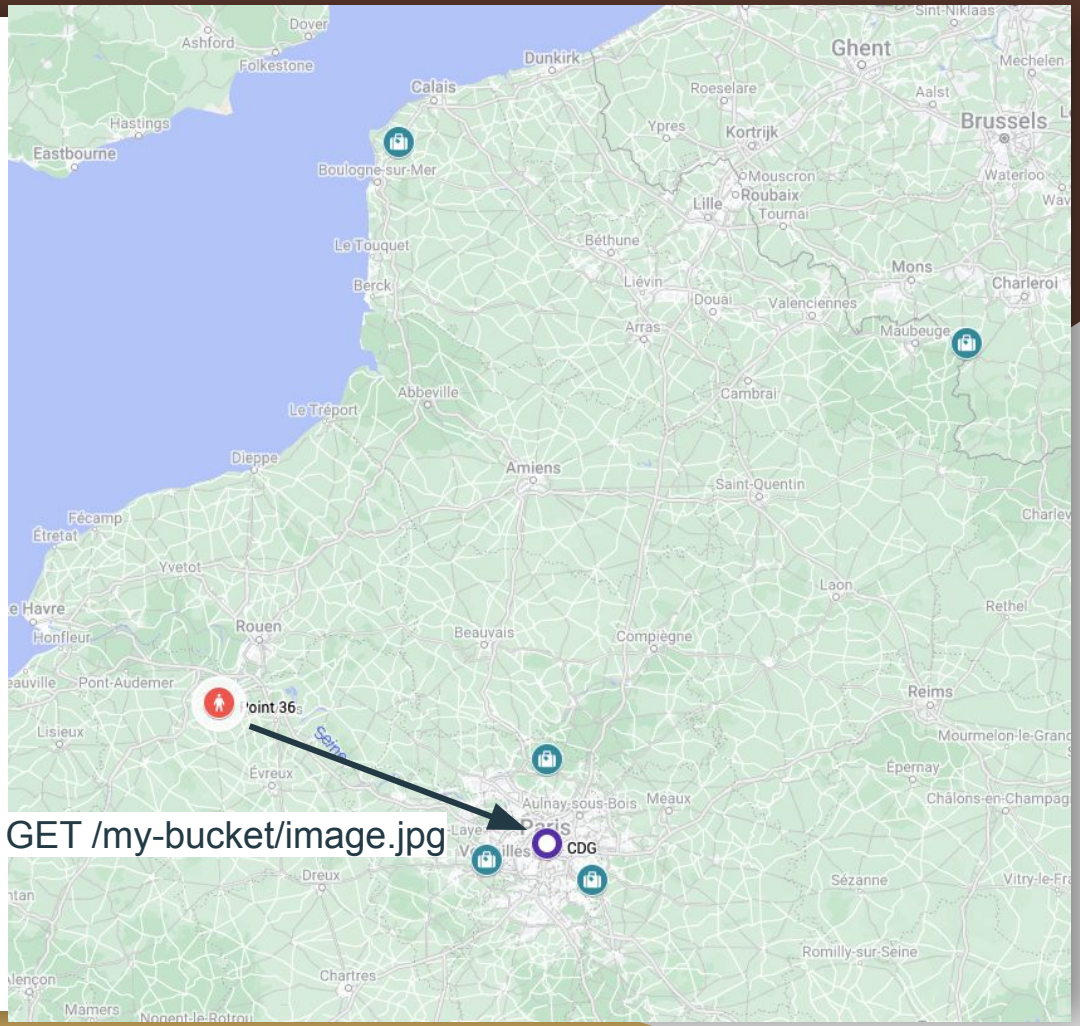
Fragment Storage: ~90 in the EU



Download Example

- User close to Paris
- EC config: 4+1
- Fragments stored at the 5 green markers

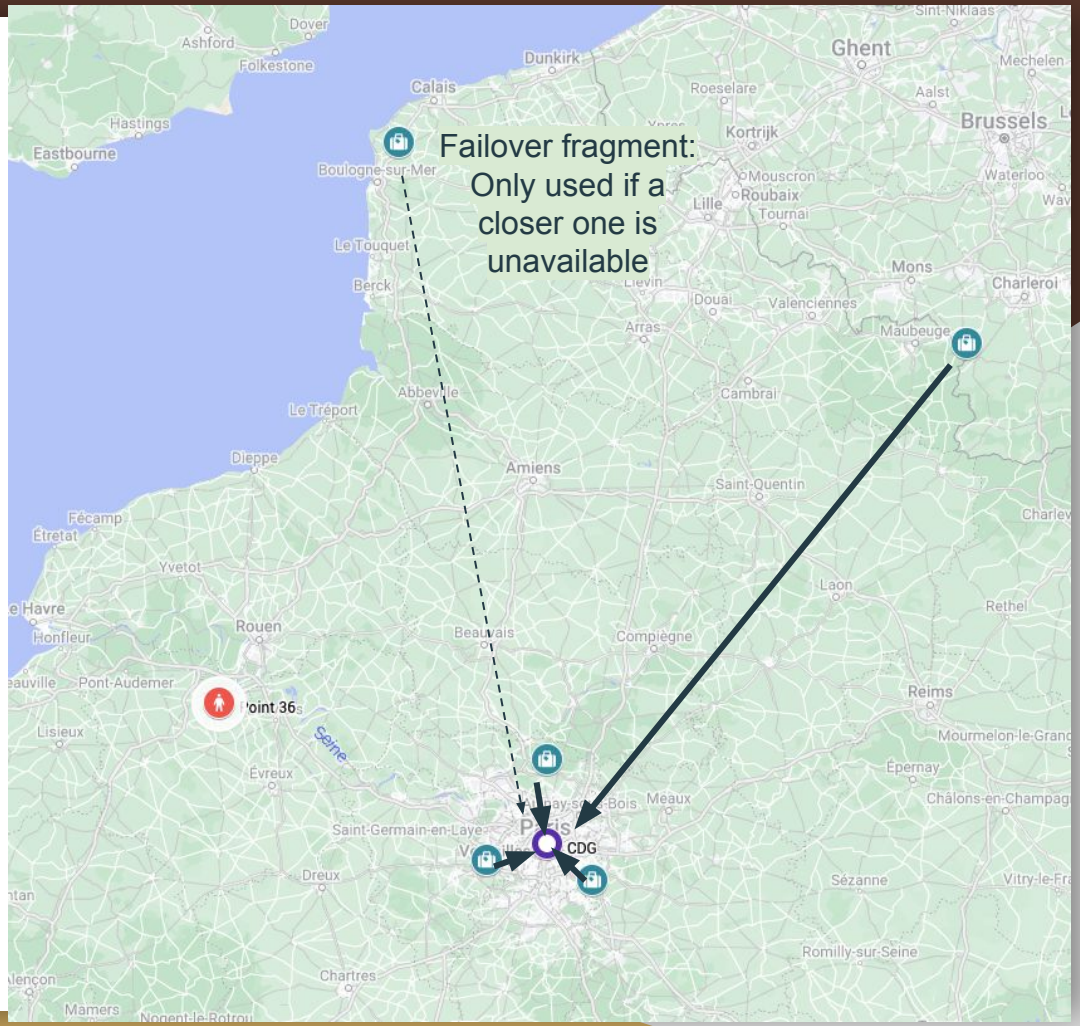
1. Request arrives to the Gateway in Paris
2. The request is authorized, bucket & object config loaded



Download Example

- User close to Paris
- EC config: 4+1
- Fragments stored at the 5 green markers

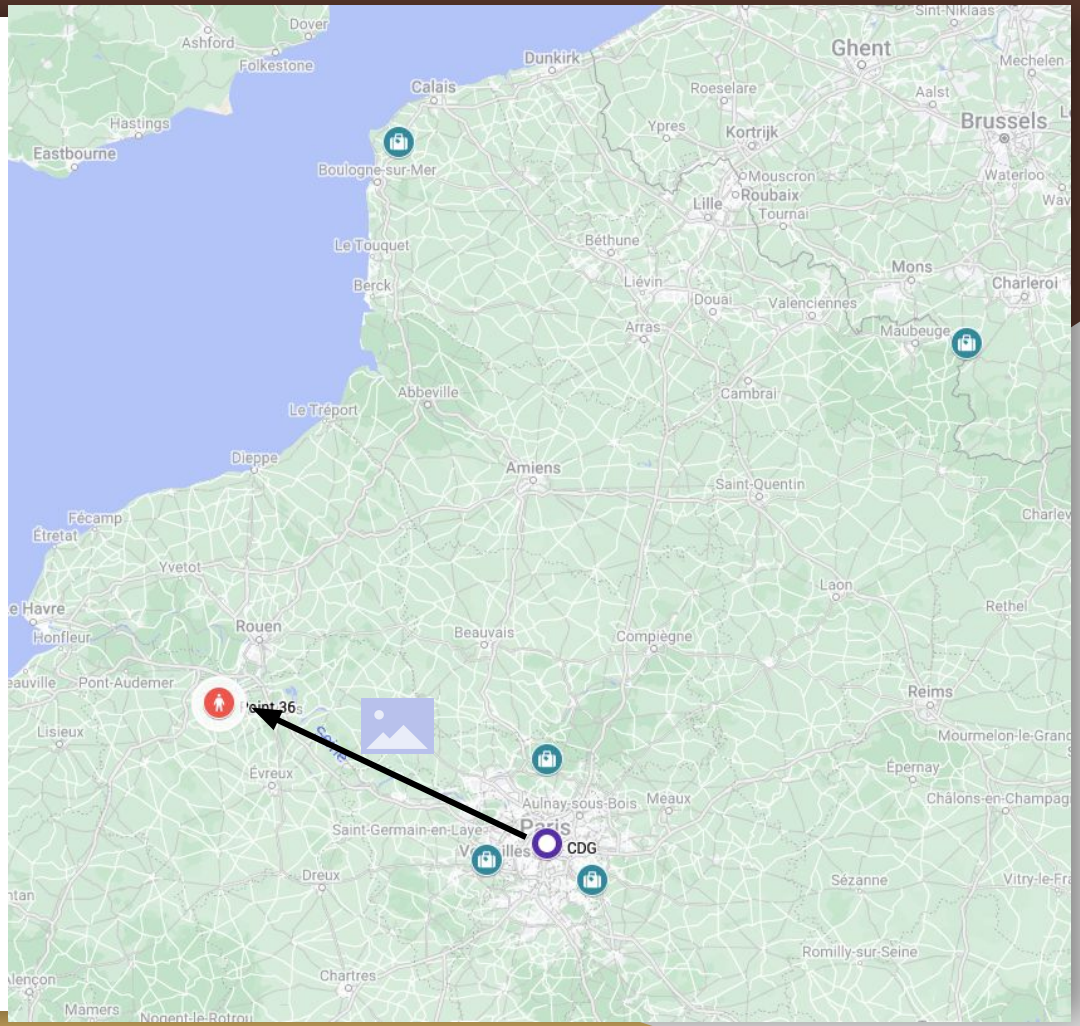
1. Request arrives to the Gateway in Paris
2. The request is authorized, bucket & object config loaded
3. Gateway downloads the 4 closest fragments



Download Example

- User close to Paris
- EC config: 4+1
- Fragments stored at the 5 green markers

1. Request arrives to the Gateway in Paris
2. The request is authorized, bucket & object config loaded
3. Gateway downloads the 4 closest fragments
4. Data reconstructed & returned



Using ML

- Bucket demand prediction
  - Proactively move or create additional fragments before an expected traffic spike
- Fragment storage performance prediction
  - Model the available latency and transfer speed of each storage location
  - Hosted by different companies, downtimes occur, speed fluctuates
  - Can be used to choose which fragments are downloaded by the Gateway
- Cost minimization
  - Continuously monitor and predict bucket traffic, try to find the cheapest config that satisfies the expected demand
  - User can also set download speed requirements

Status & Challenges

- The proposed system is partially ready, ML parts are underway
  - The object storage service works with fixed bucket configuration
  - Config can be changed manually, data migrated in the background
  - Edge storage nodes can be added (hybrid cloud-edge storage)
  - Currently training & integrating prediction models
- Some challenges
  - Controlling costs: max redundancy, nonzero cost of config change
  - Sensitivity: how quickly should the system up/downscale on demand changes (config change also takes time)
  - Late evaluation: goodness of model decisions only known later
  - Retraining: when to trigger

Thank You!