#ASLI ENGINEERING

How Twitter scales and keeps their search stable

SWIPE

BY

ARPIT BHAYANI

Scalability and Stability of Search @ Twitter

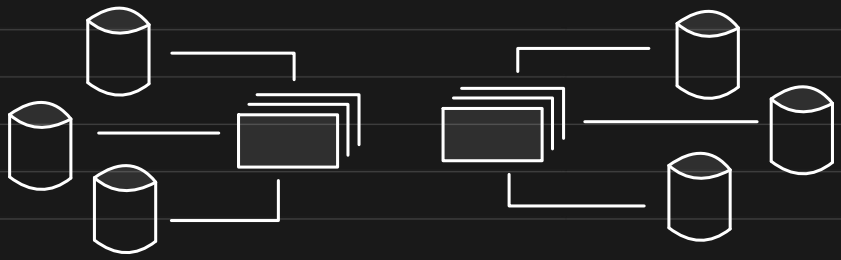
Twitter uses ElasticSearch to power search of tweets, user, directmessages

Why ElasticSearch?

Speed, Scalability, distributed nature, and simple REST APIs

Internal customers / services directly talk to ES and get things done

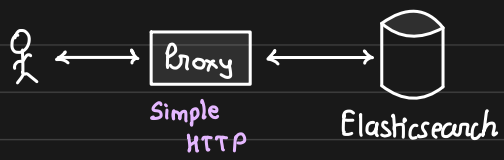
Hence, a need of standardization, stability and performance.



Elasticsearch Proxy

Querying, Indexing, monitoring, metrics was all manual & to be done separately. Proxy standardized throttling, routing, security, authentication.

monitoring – cluster health, success rate, failure rate, latency



Ingestion Service

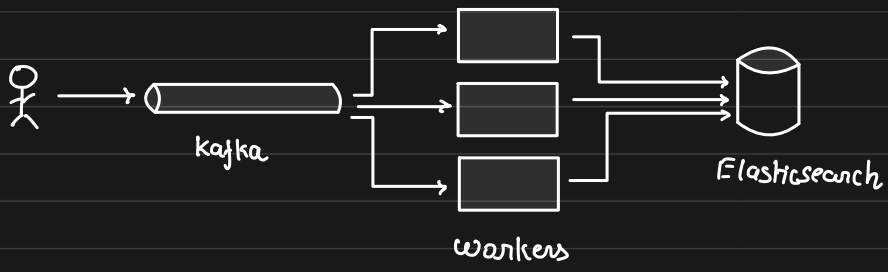
When there is a massive surge in traffic, Elasticsearch gives up!

↳ increased indexing latency

↳ increased query latency

To handle heavy ingestion gracefully, Twitter built an ingestion service

Ingestion service queues indexing request in kafka later to be consumed by workers



Advantages:

Request batching : batch write on ES cluster

Backpressure : consuming at its own pace

Throttling : slowing down if ES is overwhelmed

Retries : if cluster down, we can retry easily

Backfill Service (ingesting 100s of TB of data in ES)

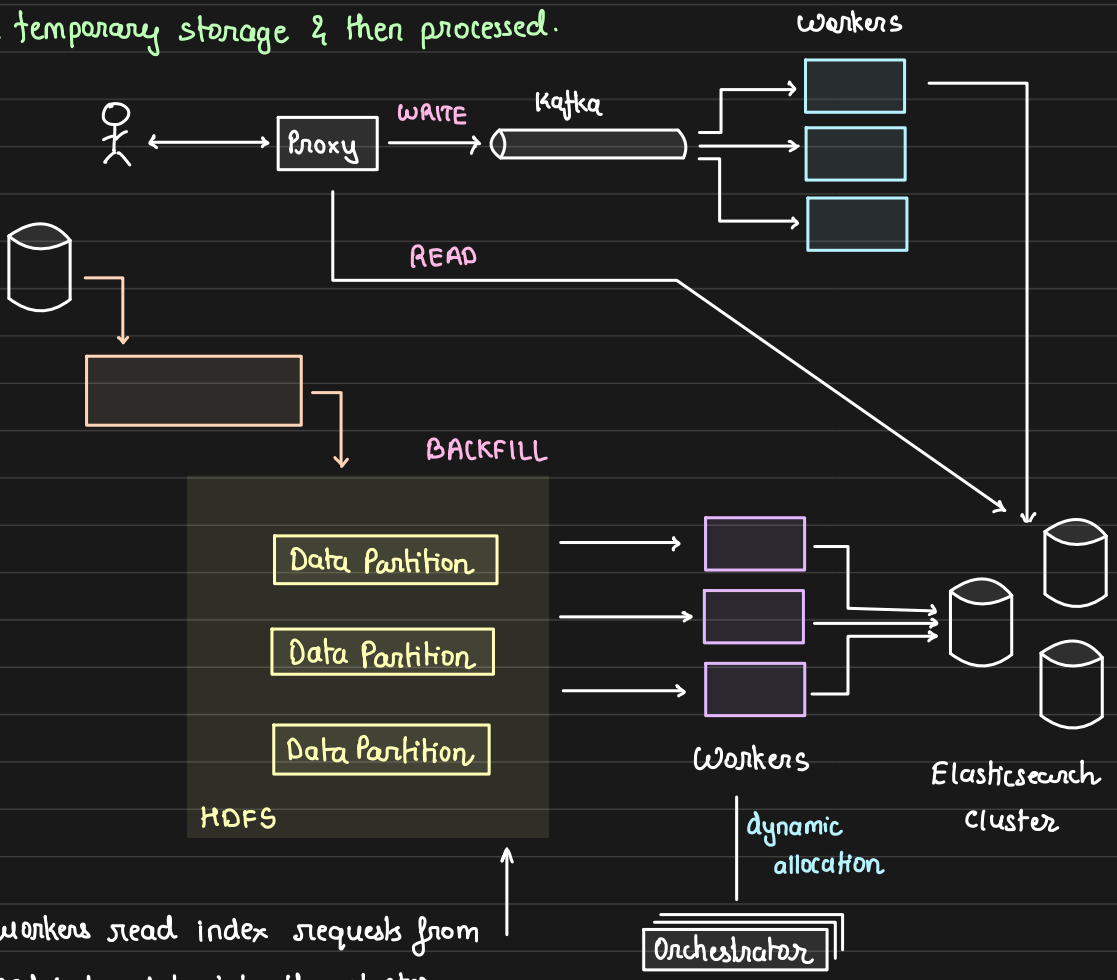
Directly ingesting massive data using MapReduce synchronously does not work

↓

Elasticsearch cannot handle this huge ingestion in one shot.

Indexing happened in sync from MapRed.

Now the indexing request is dumped in temporary storage & then processed.



Workers read index requests from and index data into the cluster.