Presto at Pinterest

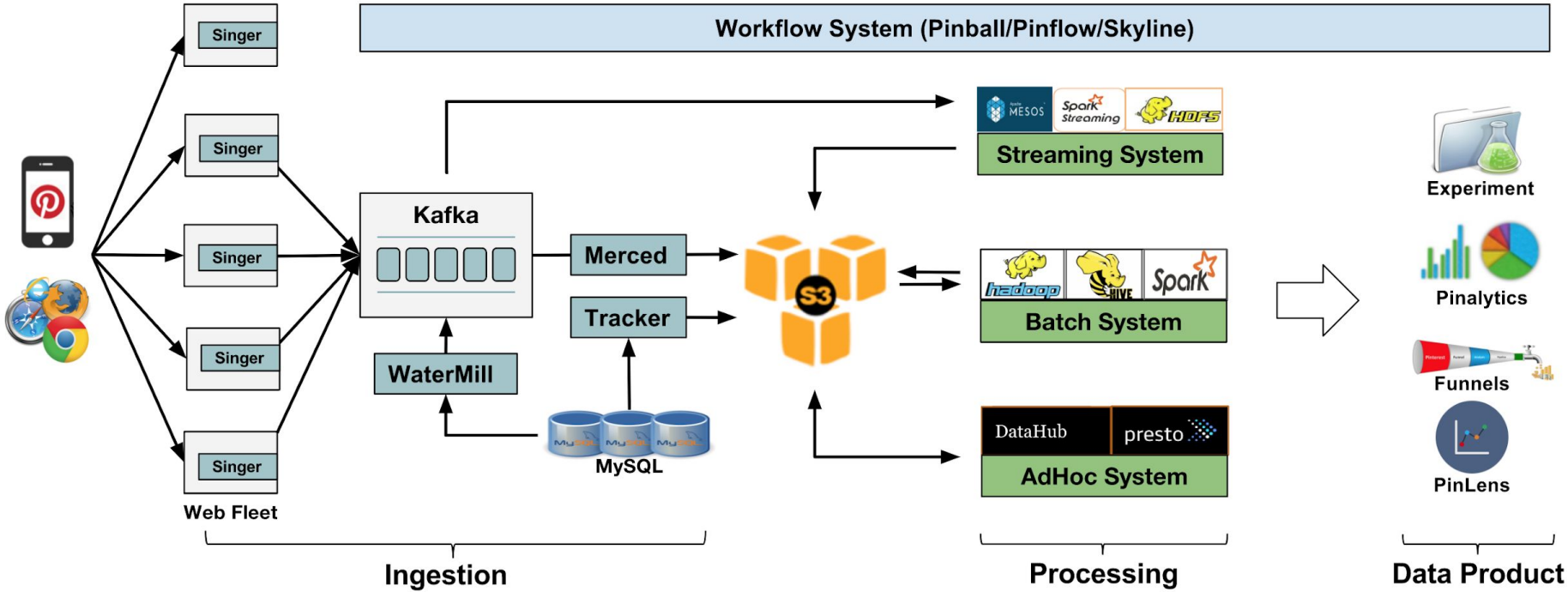
Ashish Singh

Pinterest : The World's Catalog of Ideas

Scale at Pinterest

- Biz Scale
  - 300M+ MAUs
  - 200B+ Pins
  - 4B+ Boards
- Data Scale
  - 400+ PB @ S3
  - Peak of 80k Hadoop jobs per day
  - 10,000+ Hive/Hadoop nodes
  - ~500 Presto workers (dedicated nodes + k8s)
  - > 110,000 Hive Tables
- Everything in Cloud(AWS)

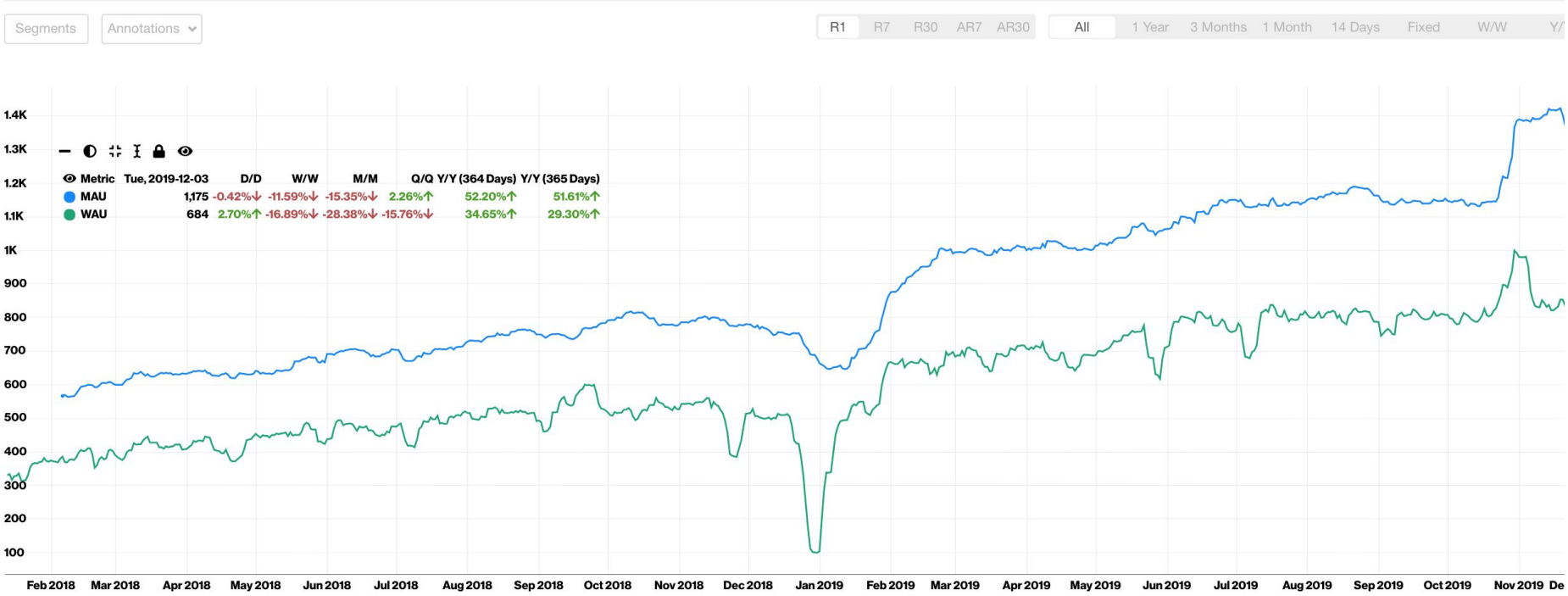
Data at Pinterest



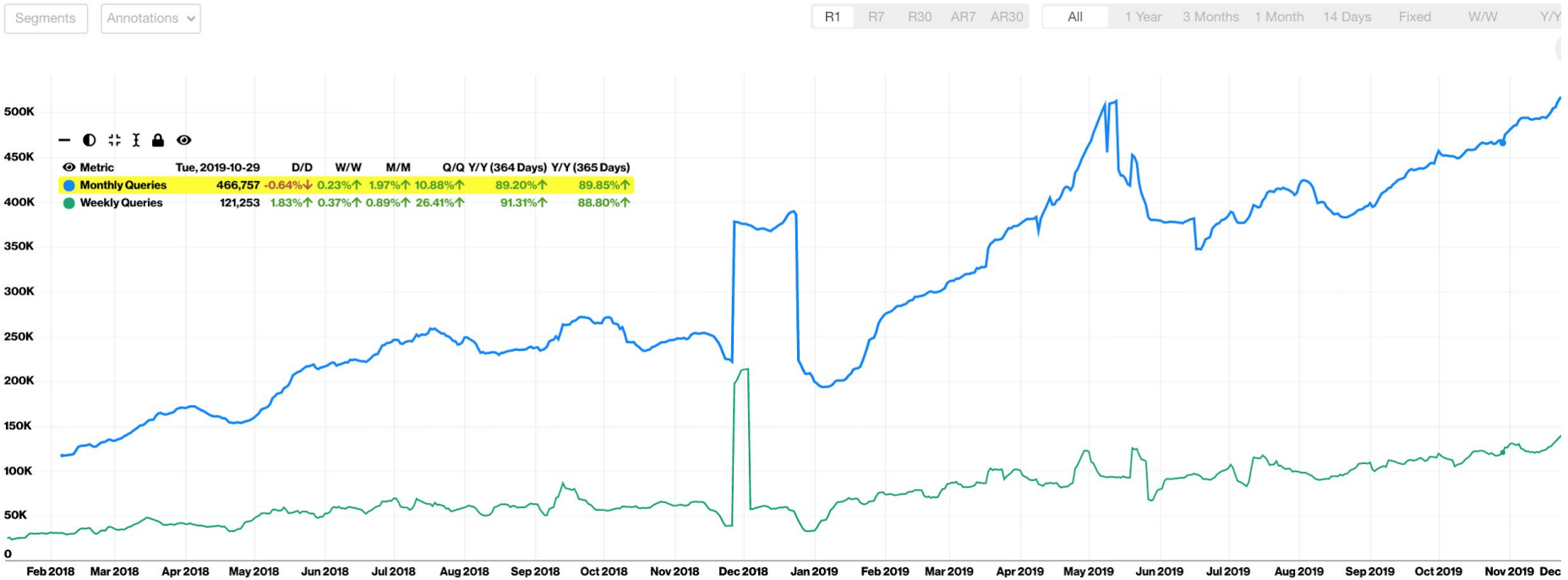
Evolution

2014	2016 Q2	2016 Q3	2016 Q4	2018	2019
Qubole + Redshift Better Support Qubole: - MR - HMS - Hive - Spark	**Qubole + Redshift + Hadoop** Better Support Happy Security Team	**Qubole + Redshift + Hadoop + Presto + HMS(RO)** Better Support	**Qubole + Redshift + Hadoop + Presto + HMS(RO) + Spark** Better Support	**Hadoop Presto (RO) + Spark + Hive + HMS (RW)** Better Support	**Hadoop Presto (RW) + Spark + Hive + HMS (RW) + Spark SQL Flink**

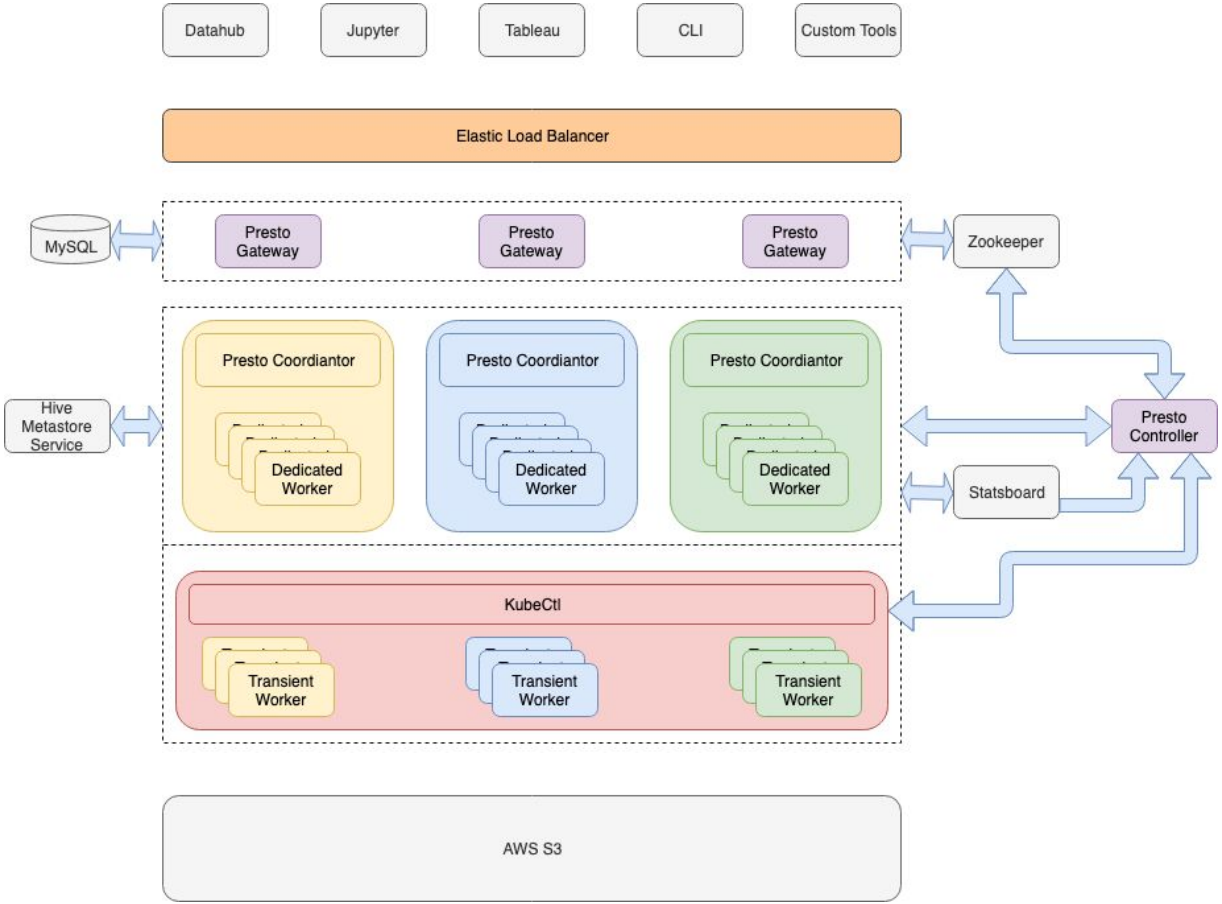
Presto at Pinterest



Presto at Pinterest



Presto at Pinterest



Presto Controller

- An in-house service critical to our Presto deployments.
- Following major functionalities are served by the controller.
  1. Health check
  2. Slow worker detection
  3. Heavy query detection
  4. Rolling restarts of Presto clusters
  5. Scaling clusters

Presto Gateway

- A service that sits between clients and Presto clusters.
- It essentially is smart http proxy server.
- Makes clients agnostic of specific Presto clusters and enables the following.
    1. Rules-based routing of queries
    2. Resource usage limits and current usage visibility for users
    3. Overall Presto clusters’ health visibility

User Query Testing Service

- Ensures Presto upgrades do not break critical user queries
- Uses Benchto
- Users register their critical queries with the service
- Used during version upgrades/ rebases

Challenges

Challenges: Deeply Nested Large Thrift Schemas

- Prime reason for coordiantor getting stuck/ crashes
- Example: A popular and commonly used large Thrift schema has over 12 million primitives and a depth of 41 levels. This schema when serialized to string takes over 282 MB.
- Close to 500 hive tables with over 100K primitives in their schemas.
- Coordinator fetches table schema from Hive Metastore and then serialize that schema in each task request it sends to workers.
    - Keeps Hive Metastore service from getting bombarded with requests from workers.
    - Adverse effect on coordinator memory and network when schemas are very large.

Challenges: Deeply Nested Large Thrift Schemas

- Our large and deeply nested schemas issue is only limited to tables using Thrift schemas.
- Thrift schema Java archive (jar) file is created and put into the classpaths of coordinator and each worker of a Presto cluster and is loaded at service start time.
- Completely got rid of schemas from tasks' requests: instead, only a Thrift class name is passed as part of the request.

Challenges: Slow or stuck workers

- Presto gains a part of its efficiency and speed from the fact that it always has JVMs up and is ready to start running tasks on workers. A single JVM is shared for multiple tasks from multiple queries on a Presto worker.
- Presto uses a multilevel feedback queue to ensure slow tasks aren't slowing down all tasks on a worker.
- This can lead to a worker having a lot of slow tasks accumulated over time, as quick tasks would be prioritized and will quickly finish.
- Slow IO tasks can also accumulate on a worker.
- As all our data sits in AWS S3 and S3 can throttle down requests if a prefix is being hit hard, which can further slow down tasks.
- If a worker is slow or stuck, the slowness gradually spreads through the Presto cluster. Other workers waiting on pages from a slow worker would slow down and will pass down the slowness to other workers.

Challenges: Slow or stuck workers

- Solving this problem requires a good detection and a fair resolution mechanism. We resorted to following checks to detect workers getting slow.
  - Check if a worker's CPU utilization is lower than cluster's average CPU utilization over a threshold and this difference is sustained over some time.
  - Check if a number of queries are failing with internal errors, indicating failure while talking to a worker over a threshold over some time.
  - Check if a worker has open file descriptors higher than a threshold for more than some time.
- Once a worker matches any of the above criteria, Presto Controller would mark the worker for a shutdown.
- A graceful shutdown is first attempted, however failure to gracefully shutdown a worker in a few attempts will lead to controller forcibly terminating the EC2 instance for dedicated workers or shutting down the Kubernetes pod hosting the worker.

Challenges: Unbalanced resources in multiple clusters

- To efficiently utilize the available resources across all Presto clusters, a new query should be sent to an under-utilized cluster or resources from an under-utilized cluster must be moved to a cluster where the query is going to run.
- It would be easier to do the former, however at Pinterest different Presto clusters have different access patterns and different characteristics.
- Moving a dedicated EC2 instance from one cluster to another would have required us to terminate and re-provision the instance. This process can easily take close to or more than ten minutes. Ten minutes in the Presto world, where our P90 query latency is less than five minutes, is a long time.

Challenges: Unbalanced resources in multiple clusters

- Instead, the Kubernetes platform provides us with the capability to add and remove workers from a Presto cluster very quickly. The best-case latency on bringing up a new worker on Kubernetes is less than a minute.
- Some other advantages of deploying on Kubernetes platform is that our Presto deployment becomes agnostic of cloud vendor, instance types, OS, etc.
- Presto controller service is responsible for scaling Presto clusters based on schedule, demand or both.

Challenges: Ungraceful cluster shut down

- Each night we restart all Presto clusters to load updated configuration parameters, Thrift schemas, custom Hive Serializer/Deserializer (SerDe) and User Defined Functions (UDFs).
- In open source Presto, there is no way to initiate a graceful shutdown of a cluster.
- Presto operators at various organizations handle graceful shutdowns by controlling traffic to the clusters.
- We’re starting to do the same at Pinterest too with Presto Gateway.
- We added a graceful shutdown capability to Presto coordinator to perform a graceful shutdown of an entire cluster.
- We plan to add a capability to reload jars without restarting processes and make some configuration parameters dynamic to get away from the need to restart clusters frequently.

Challenges: Inconsistent serdes/ schemas jar versions between coordinator and workers

- To deal with slow workers, we gracefully restart them
- While starting up, workers pull in latest schema, serdes and plugins jars
- This can lead to inconsistent jars between workers and coordiantor
- During scheduled restarts, only restart workers that have outdated jars

Challenges: Inconsistent serdes/ schemas jar versions between coordinator and workers