Design.pdf

Dataset A - File 1 with 10,000 detection events/sec

Dataset B - File 2 static reference table that is unchanged

Architecture and Tech Stack design considerations

Assume existing cloud provider used is **Azure** where company have the master agreement (MA) and SLA in place with this provider. This is to have cost optimization, streamlined budget and contractual management with other existing system, ready and available technical team, system administrator's skillsets for the operations and building the dashboard.

This existing MA contract shall be used to help align the acceptable downtime, Recovery Time Objective/Recovery Point Objective (SLA), and leverage on existing multi-region deployment.

1. **Azure Event Hubs with Apache Kafka** will be used. This is suitable for big data pipeline (10k/sec average) event streaming from various camera-sensors at the edge. (Service bus is not used as latter is costly and suited for high-value enterprise messaging, such as order processing, finance transactions.)
2. **Azure Databricks with Kafka connector** (structured streaming connector to Event Hubs) will be used to process the data from Event Hubs, using **Spark Structured Streaming** for near-real-time processing.
   Azure Databricks is a managed Apache Spark platform optimized for Cloud, and it helps to manage and optimize Spark clusters (one end-to-end platform enabling Data Engineers and Data Scientists to collaborate.)
3. **Power BI** streaming dashboard will be used to show the total of items detected for the desired time window at the geo-location, using JDBC connectors to Databricks for query.
4. Login authentication and authorization will use current company integrated Microsoft **Entra-id** for seamless single-sign-on
5. For added **security**, Azure Databricks control plane VNet is sent over TLS that are enabled by Network Security Groups(NSGs) and port IP filtering.
6. Databrick's integration with Delta Lake, enables use of Azure DataLake Storage Gen 2 **(ADLS Gen2) for Hot** storage, and **Parquet files for Cold** storage (time elapsed for the duration to go to Cold storage, and storage duration are influenced by the clarification questions in the next section).
   Power BI dashboard has direct path to the Hot storage.

7. With the above integration, ADLS Gen2 stores are exposed to Databricks via its file system to provide caching and optimized analysis. This is important for **production** system, where RDD will be **cached** after loading data from external filesystem, managed by Databricks on-top of the Spark tuning by storing in serialized form for smaller memory footprint using serialized storage levels in RDD persistence API (e.g. MEMORY_ONLY_SER vs MEMORY_AND_DISK_SER if memory is insufficient and require disk), compared to java deserialized object which are bigger size.
8. **Spark tuning** for production can be aided with Azure Databricks having Adaptive Query Execution (AQE) that can adjust shuffle partitions, manage data skew and broadcast joins. Z-Ordering/partition can be used for Delta Lake tables to optimize physical data layout for faster reads, on top of the standard Spark tuning by garbage collection tuning. It also has Photon engine that enhanced Spark performance by vectorizing queries.
9. Azure Databricks also has integration to Azure Synapse Analytics that can implement column-level security to restrict column access protecting sensitive data (for further expansion, of special output columns whose existence is not known to some users. It can implement **row-level security** such that a group of data (e.g. detection **events from cameras at restricted geo-locations** are not available for normal users). This is to cater for data compliance to organisation needs
10. This design can also leverage on Azure Monitor service for observability, e.g. to monitor resources issues.

Clarifying Questions:

- The averge is 10k events/second. What is the highest peak observed and hours, and it is correct that there are lull periods .e.g. at night where the locations are quiet and little/no traffic (**volume & velocity**).
This will help influence the design for the defining the **vertical scaling**, **horizontal** auto-scaling, number of **parallel** Databricks jobs, and sparks **partitions** (e.g. partition by several geo-locations that are grouped together)

- Is 30 secs **dashboard refresh rate** acceptable or 3-5 secs is desired.

- What is the aggregation count **accuracy** desired. An example is relative count, but not 100% accuracy required (taking note of the current requirement for no duplicates).
- What is the **latency** desired or accepted, e.g. is 3-5 secs reasonable (taking note that aggregation result should be out as soon as possible (**near real-time**) after the events has been published).
- What is the **time window** of interest for the near-real-time item count. Would the last 10 min be reasonable (taking note that there is also the cumulative total for the month

and year and historical records.
The above 3 questions will determine the processing strategy such as :

i) at least once processing (potential duplication of events, as fetched again), suitable for to show current values.
This is suitable for cumulative counter where duplicates are not important
ii) at most once processing - potential loss of events, as will not try to read old events(if failed earlier).
This is suitable for cumulative counter where 100% accuracy is not important;
iii) **exactly once processing** - suitable for exact cumulative counter
If the answer is 100% accuracy, then strategy (iii) will be used,
otherwise strategy (i) where the duplicates are handled at code level.

- The historical queries required from dashboard, which will also determine the **duration of data kept**. It is not relevant after 5 years, such that it is not used by dashboard but kept for another longer period e.g. 10 years for other data analytics used.
(This is also to determine, the data to be put in cold storage, and finally deleted to reduce cost).

- How do you consider **a new detection item** , such that the time window of an abandoned-object is considered a new detection item .e.g. after 10 mins, without occlusion (by e.g. human traffic) for more than 10mins is considered the same item
- Would you consider cross cameras tracking of the item, e.g. bicycle moving from one camera field-of-vision to another cameras be considered as same item for de-duplication algorithms.

- Would there be **new cameras** (at same location) and new locations (without cameras currently) be deployed, and how often would the existing cameras be replaced/tech-refreshed (taking note it is currently static, and replacement will still keep the old camera-oid, and update to this reference dataset would be **incremental**, not full replacement).

- Would there be **new types of sensors** deployed e.g. audio sensors, light (lidar). The code will need to cater for the various classes).

- Would the object item type list be extended in future, e.g. when object detection **algorithm technology improved**, and more items type detection are possible (i.e. new object item of interest, with progress in detection technology).

- How ofter is the data science **model updated**, would there be new output that are of interest.

- (related to above question) What are the current (future planned) semantics of the item description captured e.g. “Luggage Green 350mm height”. This is to determine the **features of item detected for the dashboard**, where the users can do a query such as “*find all luggage that is red of size 350mm in the last 10 mins at this location*”.

- Are there requirements for **alert** mechanism, such as when the total count of particular type of item exceeds a certain threshold. How would the alert be shown, e.g. on-screen with email or phone messages.