How to analyze dataflow IR cost when having TTL

Last updated by | Jackie Huang | Jan 4, 2022 at 12:24 AM PST

Contents

- Issue
- Pre-Requisite:
- Steps to troubleshoot
    - An example of how to calculate the charges.
- Recommendation:

Issue

How to analyze dataflow IR cost when having TTL

Pre-Requisite:

Please refer to this wiki to understand how TTL works.

Steps to troubleshoot

1. Open Billing report and analyzed costs for a specific day - 3/16/2021

Resource: /SUBSCRIPTIONS/96D4xxx-xxxxx-xxxxx-XXXX-XXXXXXXXX/RESOURCEGROUPS/APP-XXXXXXXX-RG/PROVIDERS/MICROSOFT.DATAFACTORY/FACTORIES/XXXXXXXXXX/INTEGRATIONRUNTIMES/XXXXXXXXXX

Consumed quantity: 3676.266667

Resource rate: 0.214499924

Extended cost: 788.5589195

2.

```
ActivityRuns
| where subscriptionId == "96D48B54-xxxxx"
| where TIMESTAMP >= datetime(2021-03-16 00:00:00) and TIMESTAMP <= datetime(2021-03-17 00:00:00)
| where effectiveIntegrationRuntime =~ "IR-ILSDC-ILMA-REPORTS"
```

For timestamp provided, on the mentioned IR, ran 1 pipeline: 485b1b64-xxxxx-xxxxx-xxxxx-xxxxxxx

3. Get IR configuration

```
GetComputeTypeAndCoreCount("Activity Run Id")
```

"computeType": "General", "coreCount": 80, "timeToLive": 60

4. Check if the same cluster is used for all activities or not. If there are different clusters, we need to add 1h (TTL) to the time of the run of activities. After the last activity of the cluster, we need to add 1 hour (TTL).

```
ActivityRuns
| where TIMESTAMP between (datetime(2021-03-16 00:00:00)..datetime(2021-03-17 00:00:00))
| where effectiveIntegrationRuntime contains "xxxxxxxxxxxxxx"
| where dataFactoryName has ""
| join (cluster('azuredmprod.kusto.windows.net').database("AzureDataMovement").CustomLogEvent
| where TIMESTAMP between (datetime(2021-03-16 00:00:00)..datetime(2021-03-17 00:00:00))
| where Message contains "Status: Ready, PoolId:") on $left.activityRunId == $right.ActivityId
| extend InstancePoolId=extract("Status: Ready, PoolId: (.*)</Text></LogProperties>", 1, Message)
| summarize InstancePooMIN = min(start) , InstancePoolMax = max(end) + time(01:00:00), InstancePoodiff = (max(
| summarize sum(InstancePoodiff)
```

There are 40 instance pools returned from the above query. This leads us to the scenario that the customer is running the activities in parallel.

durationInHours = 1.21:56:35.1973460 = ~46 hours

Formula: 46 h * 80 cores * 0.214499924 **Resource Rate** =~ 789

The reason we are seeing 40 instance pools is, only one job can run on a single cluster at a time. If there is an available cluster, but two data flows start, only one will use the live cluster. The second job will spin up its own isolated cluster. If most of your data flows execute in parallel, it is not recommended that you enable TTL.

If your data flows execute in parallel, it's recommended to not enable the Azure IR time to live property as it will lead to multiple unused warm pools.

Observations regarding the above Kusto calculation:

1. Billing is for every minute, if we have 2 min and 1 second, we will bill for 3 minutes. Same concept as for Copy Activity.
2. If you ceil the time duration for each cluster you will get very close to the price from the Billing report. These are times we are deducting from logs and logs are from different layers, so these are always going to be close approximations.
3. This cost is just dataflow. Orchestration, managed disk & bandwidth charge from data flow and other charges are separate, customers can see those in approximate billing report as well in monitoring UX.
4. If the customer has manually changed the IR payload after creation (SDK/Rest API/JSON editor on UX), then the real used TTL/core count/compute could be different than the log you are going to find on adftraceevent
5. The query in step #4 works based on if we do exclude the inactive instance pools. However, if one instance pool has no more idle VM after TTL, the backend service could choose either to reuse the existing instance pool or just create a new one. The current default behavior is always to create a new instance pool, just in case if the query data doesn't match with what the customer saw on the billing portal, make

sure to search this message “DataFlowExcludeInactiveInstancePool: True” based on the customer’s activity run from **TraceVerbose** table.

An example of how to calculate the charges.

Let's say for a given IR if there is a TTL of 30 min.

- For a given pipeline run and if there were 10 clusters created, but ran only 5 minutes activities, the charges will be 5* 10 (activity runtime) + 30*10 (TTL time) = 350 minutes.
- If an activity started a cluster, after the activity run (5 minutes) it will wait for 30 minutes (since the TTL was 30 min). Let's say if an activity run is submitted at 25 minutes, then this cluster will run the activity (10 minutes), it will again wait for 30 minutes. So, the total time being charged in this case is 5+25+10+30 = 70 minutes.

Recommendation:

Redesign the solution in order to run the activities in series or to disable the TTL property when it is parallel.

How good have you found this content?