Copy Activity being Stuck in Queued Stated while using the SHIR

Last updated by | Veena Pachauri | Mar 8, 2023 at 11:10 PM PST

Contents

We have added contents of this TSG to the new TSG here Activity Stuck Investigation Steps. This TSG will help you do investigation to find the root cause and root owner. It contains several known stuck patterns. WE WILL MAKE THIS TSG OBSOLETE IN COMING DAYS

Issue

The pipelines have been in queued state without fetching the data.

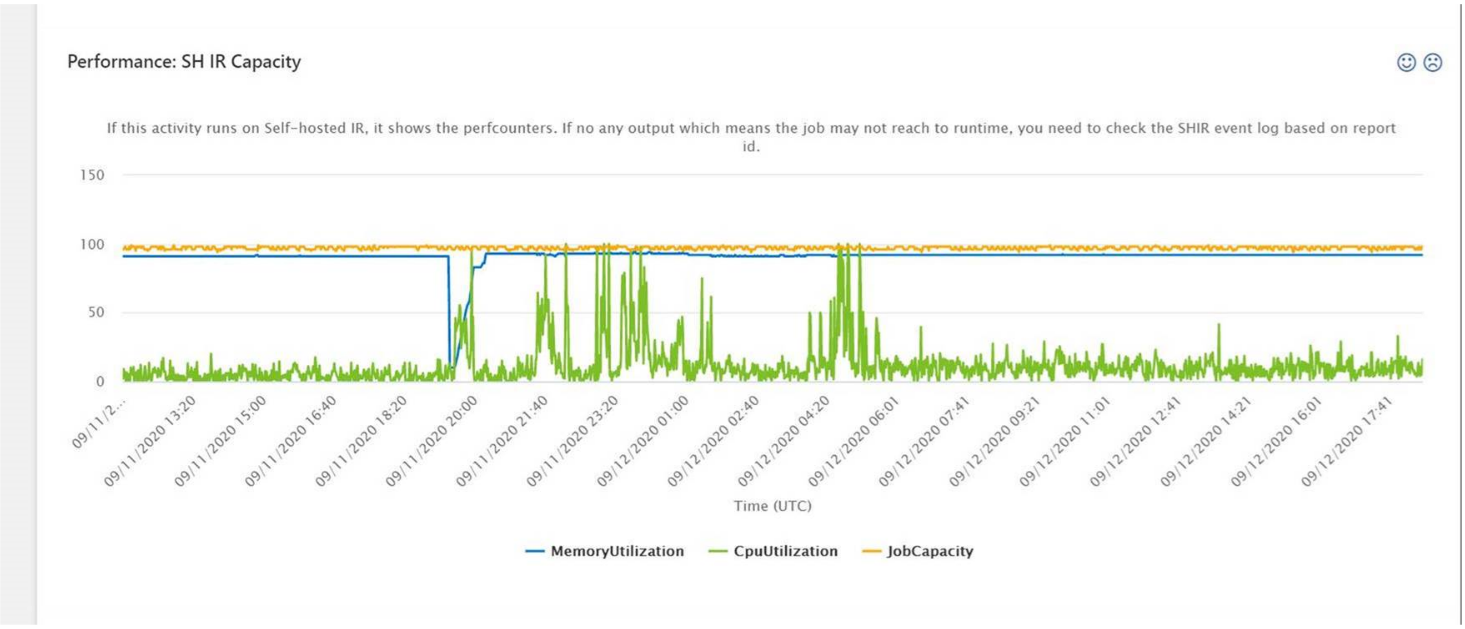
Cause

The reason might be related to the capacity constraints of the Self-hosted IR. The self-hosted IR might be running at it's full potentially and couldn't accept the new jobs.

Recommendations: This customer's SHIR run out of capacity. Please ask the customer to scale its ir. https://docs.microsoft.com/en-us/azure/data-factory/create-self-hosted-integration-runtime#scale-considerations.

Troubleshooting Steps

You can leverage the ASC to find the cause with problematic activity id:



Performance: SH IR Capacity

If this activity runs on Self-hosted IR, it shows the perfcounters. If no any output which means the job may not reach to runtime, you need to check the SHIR event log based on report id.

Alternatively, we can depend on the following detail steps to understand how to troubleshoot such similar issue:

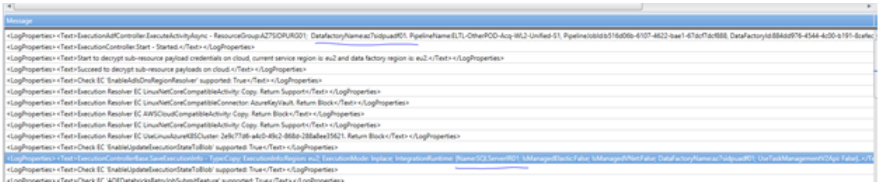
The below troubleshooting steps helps you to understand the reason for pipelines being in Queued State. Kusto Queries:

1. Try to get the Pipeline RunID for corresponding issue, After that please run the below Query

```
ActivityRuns
| where pipelineRunId has "b516d06b-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
| where activityType == "Copy".
```

2. From the above query please try to get the Activity ID which has been struck in the queued state and run below query to get the name of the SHIR machine on which these activities are running.

```
CustomLogEvent | where ActivityId == "92df7b84-xxxx-xxxx-xxxx-xxxxxxxxxxxx" //"8247e61b-xxxx-xxxx-xxxx-xxxxxxxx"
```



```
Message
<LogProperties><Text>ExecutionAdfController.ExecuteActivityAsync - ResourceGroup:AZ7SIDPURG01; DatafactoryName:az7sidpuadf01, PipelineName:ELTL-OtherPOD-Acq-WL2-Unified-S1, PipelineJobId:b516d06b-6107-4622-bae1-67dcf7dcf888, DataFactoryId:884dd976-4544-4c00-b191-8cefec
<LogProperties><Text>ExecutionController.Start - Started.</Text></LogProperties>
<LogProperties><Text>Start to decrypt sub-resource payload credentials on cloud, current service region is: eu2 and data factory region is: eu2.</Text></LogProperties>
<LogProperties><Text>Succeed to decrypt sub-resource payloads on cloud.</Text></LogProperties>
<LogProperties><Text>Check EC 'EnableAdlsDnsRegionResolver' supported: True</Text></LogProperties>
<LogProperties><Text>Execution Resolver EC LinuxNetCoreCompatibleActivity: Copy. Return Support</Text></LogProperties>
<LogProperties><Text>Execution Resolver EC LinuxNetCoreCompatibleConnector: AzureKeyVault. Return Block</Text></LogProperties>
<LogProperties><Text>Execution Resolver EC AWSCloudCompatibleActivity: Copy. Return Block</Text></LogProperties>
<LogProperties><Text>Execution Resolver EC LinuxNetCoreCompatibleActivity: Copy. Return Support</Text></LogProperties>
<LogProperties><Text>Execution Resolver EC UseLinuxAzureK8SCluster: 2e9c77d6-a4c0-49c2-868d-288a8ee35621. Return Block</Text></LogProperties>
<LogProperties><Text>Check EC 'EnableUpdateExecutionStateToBlob' supported: True</Text></LogProperties>
<LogProperties><Text>ExecutionControllerBase.SaveExecutionInfo - Type:Copy; ExecutionInfo:Region: eu2; ExecutionMode: Inplace; IntegrationRuntime: {Name:SQLServerIR01; IsManagedElastic:False; IsManagedVNet:False; DataFactoryName:az7sidpuadf01; UseTaskManagementV2Api: False}.</Te
<LogProperties><Text>Check EC 'EnableUpdateExecutionStateToBlob' supported: True</Text></LogProperties>
<LogProperties><Text>Check EC 'ADFDatabricksRetryJobSubmitFeature' supported: True</Text></LogProperties>
```

3. You can also run the following Query to get the Name of the SHIR Machine Directly from the logs,

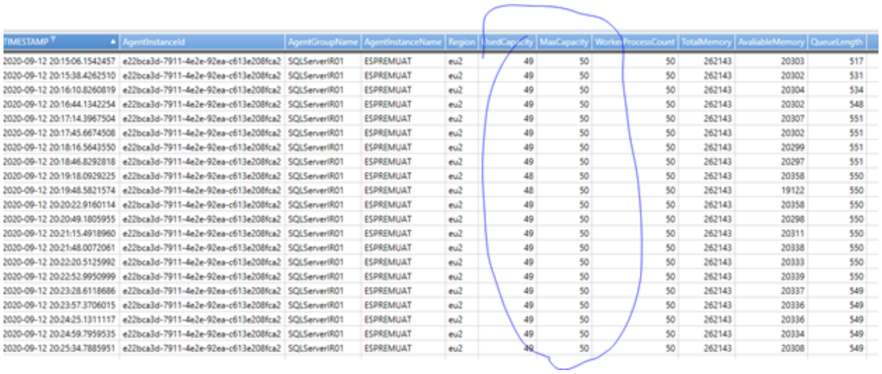
```
CustomLogEvent | where ActivityId == "6a041428-3360-4fb3-8f07-f1d2ee6be035"
| where Message contains "integrationRuntimeName"
```

4. Then check the IRMS log, you can find the agent id and the node id for this ir

```
Operations
| where DataFactoryName == "xxxxxxxxxx" and ResourceName == "xxxxxxxxx"
| where TIMESTAMP > datetime(2020-09-11) and TIMESTAMP < datetime(2020-09-13)
| distinct AgentId
```

5. Then run the below query against the Hearbeat log to find out if the SHIR ran out of capacity. See below example query and corresponding results which shows SHIR ran out of memory.

```
Heartbeats
| where AgentGroupId == "e4917e8f-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
| where TIMESTAMP > datetime(2020-09-12 20:15:00)
| project TIMESTAMP , t = parsejson(Telemetries) , AgentInstanceId, AgentGroupName, AgentInstanceName, Region
| extend UsedCapacity=toint(t["UsedCapacity"]), MaxCapacity=toint(t["MaxCapacity"]), WorkerProcessCount = toint(t["WorkerProcessCount"]), TotalMemory = toint(
AvaliableMemory = toint(t["AvaliableMemory"]), QueueLength = toint(t["QueueLength"])
| project-away t
```



TIMESTAMP	AgentInstanceId	AgentGroupName	AgentInstanceName	Region	UsedCapacity	MaxCapacity	WorkerProcessCount	TotalMemory	AvaliableMemory	QueueLength
2020-09-12 20:15:06.1542457	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20303	517
2020-09-12 20:15:38.4262510	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20302	531
2020-09-12 20:16:10.8260819	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20304	534
2020-09-12 20:16:44.1342254	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20302	548
2020-09-12 20:17:14.3967504	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20307	551
2020-09-12 20:17:45.6674508	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20302	551
2020-09-12 20:18:16.5643550	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20299	551
2020-09-12 20:18:46.8292818	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20297	551
2020-09-12 20:19:18.0929225	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	48	50	50	262143	20358	550
2020-09-12 20:19:48.5821574	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	48	50	50	262143	19122	550
2020-09-12 20:20:22.9160114	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20358	550
2020-09-12 20:20:49.1805955	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20298	550
2020-09-12 20:21:15.4918960	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20311	550
2020-09-12 20:21:48.0072061	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20338	550
2020-09-12 20:22:20.5125992	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20333	550
2020-09-12 20:22:52.9950999	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20339	550
2020-09-12 20:23:28.6118686	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20337	549
2020-09-12 20:23:57.3706015	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20336	549
2020-09-12 20:24:25.1311117	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20336	549
2020-09-12 20:24:59.7959535	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20334	549
2020-09-12 20:25:34.7885951	e22bca3d-7911-4e2e-92ea-c613e208fca2	SQLServerIR01	ESPREMUAT	eu2	49	50	50	262143	20308	549

Reference

https://portal.microsofticm.com/imp/v3/incidents/details/204800827/home