How to investigate EventTrigger being fired twice for the same blob

Last updated by | Supreeth Vasudevamurthy | Feb 15, 2022 at 11:56 AM PST

Contents

Important Note

This step-by-step guide will assist you in resolving/isolating the BlobEventTrigger related issues. Before proceeding with the investigation, please ensure you've followed the below TSGs:

- Blob Event Trigger firing twice
- Event Trigger Issues

Case Study Scenario

EventTrigger being fired more than once for the same blob. Customer is facing issue on file create trigger run more than once. Engineer confirmed that the trigger was firing twice from the portal and confirmed that the upload method is BlockBlob and IgnoreEmptyBlobs was enabled.

Information shared by customer

- **Issue Start Time (UTC) (Required):** 2020-05-12T14:00:00.000Z
- **Data Factory Name:** CustomerDataFactory
- **Pipeline Run ID [separate with commas]:** XXXXXXXXXXXXXXXXX, XXXXXXXXXXXXXXXXXXXXXXXXX
- **Trigger Name:** TriggerTargetHours
- **File/Folder path(uri):**
  - "blobPathBeginsWith": "/ingest/blobs/dbo.TargetHours.txt",
  - "blobPathEndsWith": "dbo.TargetHours.txt",
  - "ignoreEmptyBlobs": true,

Investigation of the issue

1. Query the **ApiOperationEvent** table to confirm if the eventTrigger was executed twice. You may notice more than one different **TraceCorrelationId** (PipelineRunId) which customer has also shared.

```
ApiOperationEvent
| where env_time >= datetime(2022-02-10 06:50:49.3063252) and env_time <= datetime(2022-02-10 06:52:49.3063252
| where DataFactoryName =="CustomerDataFactory"                              //DataFactory Name
| where operationName has "PUT TriggerEvent/ProcessTriggerEventAsync"
| where RequestUri has "/BlobEventsTrigger/TriggerTargetHours"        //TriggerName
| project env_time, TraceCorrelationId, ApiName, RequestUri, TenantId, DataFactoryName
| sort by env_time asc
```

Output:

env_time	TraceCorrelationId	ApiName	RequestUri
2022-02-10 06:51:49.3063252	e841c95e-8716-49fc-a9d3-9584d35b1d58	PUT TriggerEvent/ProcessTriggerEventAsync	PUT https://pmeastus.svc.datafactory.azure.com/triggerevent/BlobEventsTrigger/Production Blob Trigger abs files
2022-02-10 06:51:59.0275482	7251efbe-a9a3-46ef-a868-aa3ac7df4215	PUT TriggerEvent/ProcessTriggerEventAsync	PUT https://pmeastus.svc.datafactory.azure.com/triggerevent/BlobEventsTrigger/Production Blob Trigger abs files
2022-02-10 06:52:05.5097516	0507be64-d018-4ee6-beb2-774d9ac9b131	PUT TriggerEvent/ProcessTriggerEventAsync	PUT https://pmeastus.svc.datafactory.azure.com/triggerevent/BlobEventsTrigger/Production Blob Trigger abs files
2022-02-10 06:52:15.0221309	7d24b6ee-b8ef-45f9-8b3a-1a91f7e9a100	PUT TriggerEvent/ProcessTriggerEventAsync	PUT https://pmeastus.svc.datafactory.azure.com/triggerevent/BlobEventsTrigger/Production Blob Trigger abs files
2022-02-10 06:52:22.2760817	b597962f-46c0-4217-b6a7-95cb8cb272ff	PUT TriggerEvent/ProcessTriggerEventAsync	PUT https://pmeastus.svc.datafactory.azure.com/triggerevent/BlobEventsTrigger/Production Blob Trigger abs files

2. Query the **AdfTraceEvent** logs with TraceCorrelationId as XXXXXXXXXXXXXXXXXXXXXX to get the TriggerId

```
AdfTraceEvent
| where env_time >= datetime(2022-02-10 06:50:49.3063252) and env_time < datetime(2022-02-10 06:52:49.3063252)
| where TraceCorrelationId in ("e841c95e-8716-49fc-a9d3-9584d35b1d58",
"7251efbe-a9a3-46ef-a868-aa3ac7df4215",
"0507be64-d018-4ee6-beb2-774d9ac9b131",
"7d24b6ee-b8ef-45f9-8b3a-1a91f7e9a100",
"b597962f-46c0-4217-b6a7-95cb8cb272ff") // //Add multiple PipelineRunId here, since there is more than one Pip
| where ComponentId == "PipelineManager"
| where Message matches regex "TriggerId" | extend TriggerId = substring(extract("TriggerId=[{]?[0-9a-fA-F]{8}
| project env_time, TriggerId
```

Output:

env_time	TriggerId
2022-02-10 06:51:46.7703051	1dd025b3-5a5e-4f26-93a1-5144b840a0a1
2022-02-10 06:51:58.8524966	6b75bd94-e226-4dff-805c-0154ef6e1286
2022-02-10 06:52:05.2794041	72f7f185-8102-47b1-8f2a-746d768b6bfa
2022-02-10 06:52:14.8057986	1373e547-c05a-45da-800e-c8a4bac69d98
2022-02-10 06:52:22.0749519	608d37b2-91a1-4e7d-bcf0-9fccaf58f5f9

3. Query the **AdfTraceEvent** Log again, but this time TraceCorrelationId as the **TriggerId**, to get the blob name and path which triggered this particular pipeline

```
AdfTraceEvent
| where env_time >= datetime(2022-02-10 06:50:49.3063252) and env_time < datetime(2022-02-10 06:52:49.3063252)
| where TraceCorrelationId in ("1dd025b3-5a5e-4f26-93a1-5144b840a0a1",
"6b75bd94-e226-4dff-805c-0154ef6e1286",
"72f7f185-8102-47b1-8f2a-746d768b6bfa",
"1373e547-c05a-45da-800e-c8a4bac69d98",
"608d37b2-91a1-4e7d-bcf0-9fccaf58f5f9") // Trigger Ids from above query
| where ComponentId == "PipelineManager"
| where Message has "Processing event for triggerName:"
| project env_time, Message,TraceCorrelationId | extend EventId = substring(extract("Id=[{]?[0-9a-fA-F]{8}-([0
| extend EventType = substring(extract("EventType=*.*",0, Message, typeof(string)),28, 11)
| extend File_name_with_path = substring(extract("Subject=([^,]+)",0, Message, typeof(string)),8, 1000)
| project env_time, EventId, EventType, File_name_with_path
| sort by env_time asc
```

Output:

env_time	EventId	EventType	File_name_with_path
2022-02-10 06:51:43.5187509	d40c08d7-f01e-00b1-234a-1e230806148b	BlobCreated	/blobServices/default/containers/albertsons/blobs/processed20220210.abs
2022-02-10 06:51:50.2298484	c884765e-701e-00e4-444a-1ec87f066557	BlobCreated	/blobServices/default/containers/albertsons/blobs/processed20220210.abs
2022-02-10 06:51:57.3678695	05d47c38-f01e-001b-7d4a-1ef5e706bb4f	BlobCreated	/blobServices/default/containers/albertsons/blobs/processed20220210.abs
2022-02-10 06:52:02.9654996	45a45be7-701e-00ef-434a-1ed00b06fe3d	BlobCreated	/blobServices/default/containers/albertsons/blobs/processed20220210.abs
2022-02-10 06:52:10.3014788	0e328d61-a01e-00a9-374a-1e0e9d06583c	BlobCreated	/blobServices/default/containers/albertsons/blobs/processed20220210.abs

4. As it can be seen in the above screenshot, there are 5 different event ids, which means that there are 5 different "BlobCreated" events which Storage sent ADF
5. At this time, if you see that number of pipelines triggered is equal to the number of event Ids generated, that means Storage sent us those many number of events for which, a pipeline is triggered each. If the number of pipelines triggered is more than the number of Event Ids generated (for the timestamp and Blob path combination), get SME help in Ava to engage Orchestration PG.

P.S - The Event id in our logs is considered to be source of truth wherein we can confidently tell the Customers that Storage has sent us those many requests, but the caveat here is that there might be a Microsoft Service which is ingesting data to ADLS Gen2 Storage, for Eg; Azure Stream Analytics (ASA) is a client which might ingest data to ADLS Gen2 Storage, during cases like this, it makes sense to contact the Client (if Microsoft service) and check the logs to see why there are multiple requests of Put blob for one instance of Blob file upload. Some clients like ASA batch the data in one file and create multiple PUT requests to upload the data, which might trigger multiple events due to which ADF triggers multiple pipelines. If the client who is ingesting the Blob data is outside of Microsoft, suggest Customer to contact the client directly and look at their implementation.

Additional Information:

- **Icm References**: ICM# 185500886, 281529743, 285476315, 284153928
- **Author**: vimals
- **Reviewer**: vimals
- **Updated**: supreethv on 2/14/2022
- **Keywords**:

How good have you found this content?