Historic data export to Azure blob storage

About this article

This article presents a method to use Logic Apps to query historic data from a Log Analytics workspace and store it in Azure storage. Use this article when you need to export your Azure Monitor log data for auditing and compliance scenarios or to allow another service to retrieve this data.

Overview

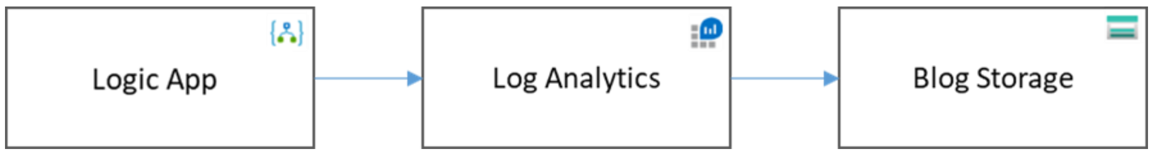
The recommended way for export Log Analytics data is at the pipeline via Diagnostic Settings for Azure resources, or Log Analytics data export, which is in preview.

There are cases where data is already in your workspace and you need to have a copy of it in storage for longer retention, use it in other Azure service or another cloud. The procedure in this doc describe a way to export historic data to blob storage using Azure Logic App, but it can easily be altered to any other destination such as Event Hub.

To overcome the API limits for size and record count, the export method is based on batches of records.

The basic method used in this solution is to create a Logic App that queries data from the Log Analytics workspace and writes to an Azure storage account. The workspace and storage account can be in different regions, but sending data to another region incurs additional charges.

The workflow is based on row export starting from particular date and time onward, making sure that no data is missed. The data is exported and saved is storage separated to folders per batch run.



Prerequisites

The following are the requirements for the Azure resources used in this scenario:

1. A Log Analytics workspace in Azure Monitor. The user who creates the logic app must have at least read permission to the workspace.
2. Azure storage account as destination for your log data. The user who creates the logic app must have write permission to the storage account.
3. Before creating your Logic App, make sure you have the following information:
    - Subscription ID of your workspace
    - Resource group of your workspace
    - Log Analytics workspace name
    - Storage account name

Create storage account

You can use an existing Azure storage account or use the procedure at Create a storage account to create a new one. Use the procedure at Create a container to add a container to the storage account. The storage account doesn't have to be in the same subscription as your Log Analytics workspace.

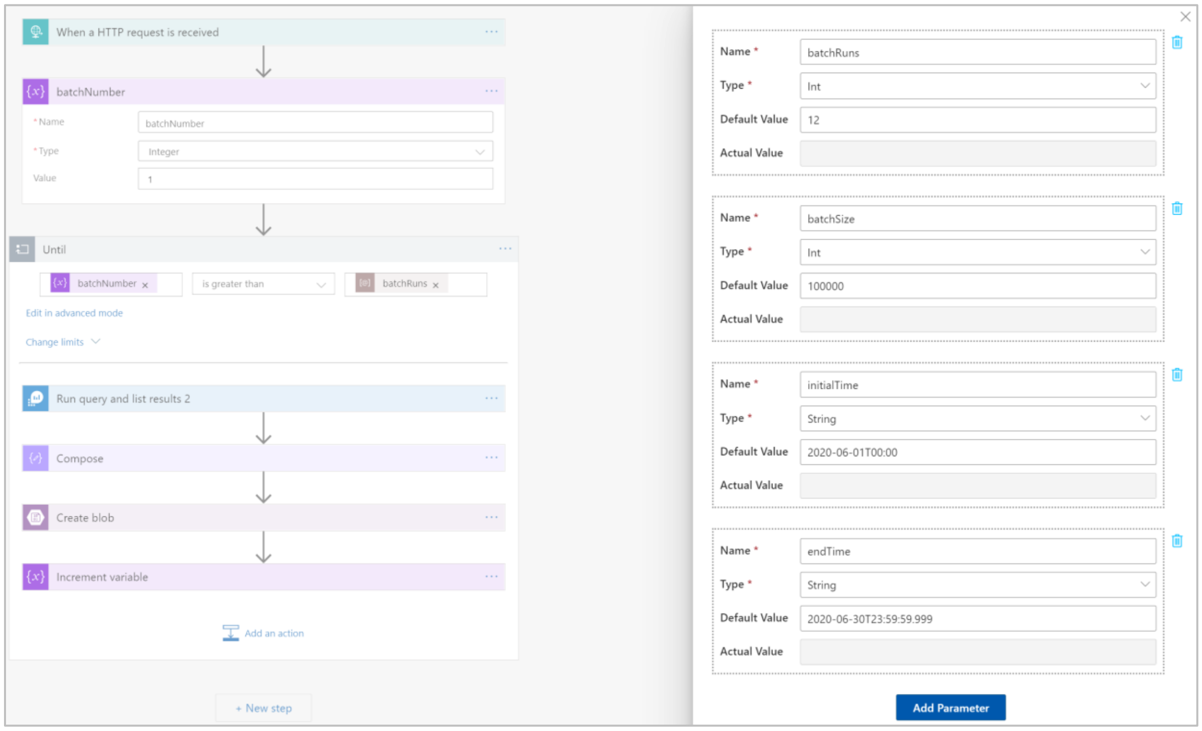
The name used for the container with the **Create Blob** action below is **loganalytics-data**, but you can use any name.

Create Logic App

This paragraph describe the steps to create a Logic App with the following components:

- HTTP request to trigger the workflow, but since data is exported once, you normally trigger your Logic App manually.
- Parameters to simplify the configuration and management of your export:
    - startTime -- export start time
    - endTime-- export end time
    - batchSize -- the number of records exported in each batch -- Can be 500,000 or less.
    - batchRuns -- the number of export execution batches
- Variables to be used in Logic App loop
    - batchNumber -- It's the initial export batch and always set to '1'
- **Until** control execute the batches
- A **Log Analytics connector** to run query and list results.
- (Optional) **Parse JSON** to parse the output from the Log Analytics query into individual JSON records.
- **Compose** to create

- A **Blob Storage connector** to create a blob in the storage account with the query results.
- **Increment** variable used for the recurrence loop



batchNumber

* Name	batchNumber
* Type	Integer
Value	1

Until: batchNumber is greater than batchRuns

Name *	batchRuns
Type *	Int
Default Value	12
Actual Value	

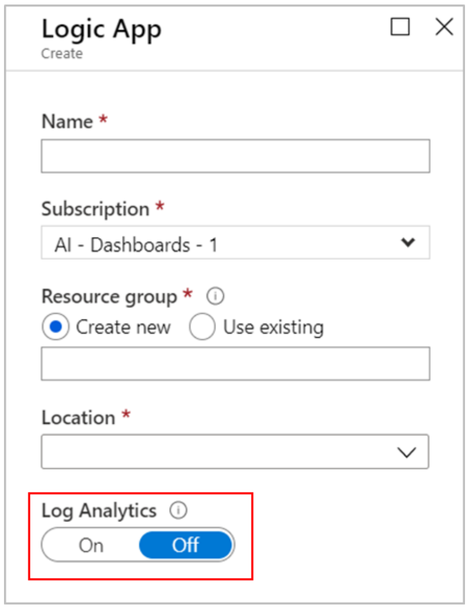
Name *	batchSize
Type *	Int
Default Value	100000
Actual Value	

Name *	initialTime
Type *	String
Default Value	2020-06-01T00:00
Actual Value	

Name *	endTime
Type *	String
Default Value	2020-06-30T23:59:59.999
Actual Value	

Create new blank workflow

1. Go to the **Logic Apps** menu in the Azure portal.
2. Click **Add** to create a new workflow.
3. Provide a name such as **ArchiveLogAnalytics**. You can select an existing resource group or create a new one. The logic app doesn't need to be in the same resource group or location as the workspace.
4. You do not need to select the **Log Analytics** option. This sends runtime events to a Log Analytics workspace each time the workflow runs. It's unrelated to retrieving data from the workspace.
5. Logic App Type : **Consumption**


Logic App
Create

Name *

Subscription *
AI - Dashboards - 1

Resource group *
● Create new ○ Use existing

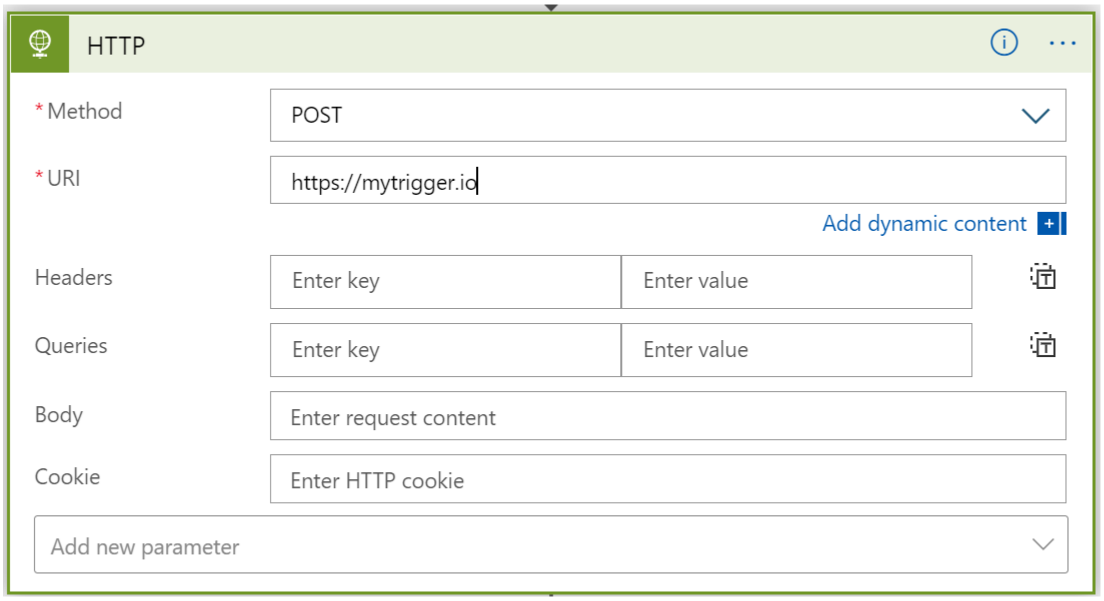
Location *

Log Analytics
On [✓ Off]

6. Click **Create** to create the blank workflow.
7. You may need to click **Refresh** to see your new workflow. Click on it to start editing.
8. Click **Blank Logic App** to start the enter with a blank workflow.

Add HTTP trigger

The trigger to initiate the workflow. Since historic data is likely exported once, you can simply trigger your Logic App manually as well.


HTTP

* Method	POST
* URI	https://mytrigger.io
Headers	Enter key / Enter value
Queries	Enter key / Enter value
Body	Enter request content
Cookie	Enter HTTP cookie

1. Type *Request* in the search box and then select the HTTP request trigger.
2. Set POST method and provide any URI. HTTP POST URL will be generated automatically when you save the workflow.

Add parameters to the workflow

Parameters makes the export configuration easier.

- **startTime** – Start time range for export
- **endTime** – End time range for export
- **batchSize** – Number of records in each batch, can be 500,000 or less. Let's set it to 100,000 in this example.
- **batchRuns** – Determine the number of batchSize executions to perform.

  Let's say that you want to export data for this query:

```
SecurityEvent
| where TimeGenerated between(startofday(datetime(2020-010-
01)) .. endofday(datetime(2020-10-03)))
| project _BilledSize, TimeGenerated, Computer, Activity, AccountType, Eve
ntID
```

  Execute the following query in your workspace to get the total number of records, total volume in MB and average record volume – If time rage is large and your query times-out, use a fraction of the time range and then apply the proportion on the calculation below:

```
SecurityEvent
| where TimeGenerated between(startofday(datetime(2020-010-
01)) .. endofday(datetime(2020-10-03)))
| project _BilledSize, TimeGenerated, Computer, Activity, AccountType, Eve
ntID
| summarize RecordCount = count(), volumeMiB = sum(_BilledSize)/pow(1024,2
)
| extend AvgRecordVolumeMiB = volumeMiB / RecordCount
```

  Example query results:

RecordCount	volumeMiB	AvgRecordVolumeMiB
10521409	1169.8412733078003	0.0009572847992070644

Parameters’ calculation

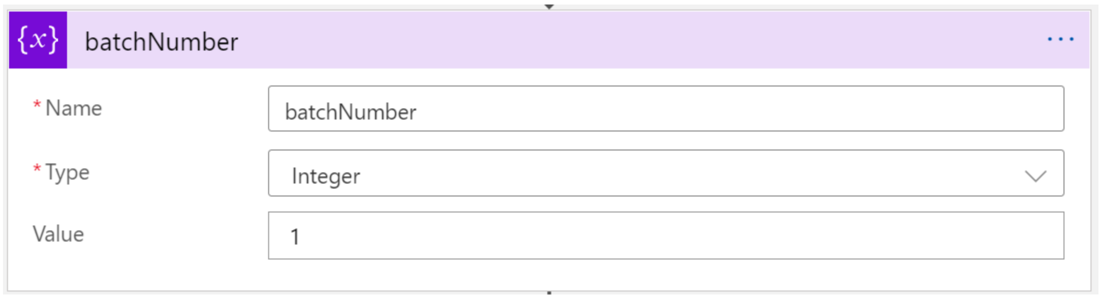
Verify the number of iterations needed for the export considering the connector limit for volume (16MiB) and for records (500,000 records). Data volume in Log Analytics and storage account are calculate differently – storage includes the JSON envelope and therefore shoew higher volume. 16 MiB in Log Analytics query translated to about 23 MiB in storage.

- **batchRuns**: Calculate iteration count. It’s recommended to increase the number of iterations by 10% to assume that the size and number of records are in safe margin: 110% * Int(volumeMiB / 16MiB) + 1 = 81
- **batchSize**: Calculate records per iteration, int(RecordCount / 81) + 1 = 129,894

Add variable to the workflow

Variables make the export configuration easier

- batchNumber – this always set to ‘1’



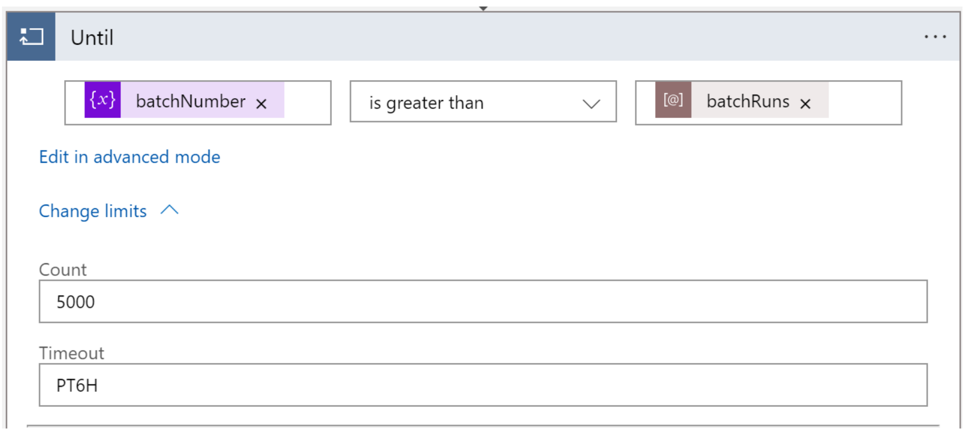
batchNumber

* Name	batchNumber
* Type	Integer
Value	1

Add Until control

The control executes the recurrences on workspace query and data export to blob storage using the parameters values you provide.

1. Add the start iteration number – *batchNumber* start at ‘1’
2. Set condition to ‘is greater than’
3. Add the end iteration number – *batchRuns* parameter
4. Expand “Change limits” and replace the default with value larger than your ‘batchRuns’ to allow completion – the max supports value is 5000. Timeout is 1 hour by default and you may need to increase it for longer iterations.


Until

batchNumber · is greater than · batchRuns

Count: 5000

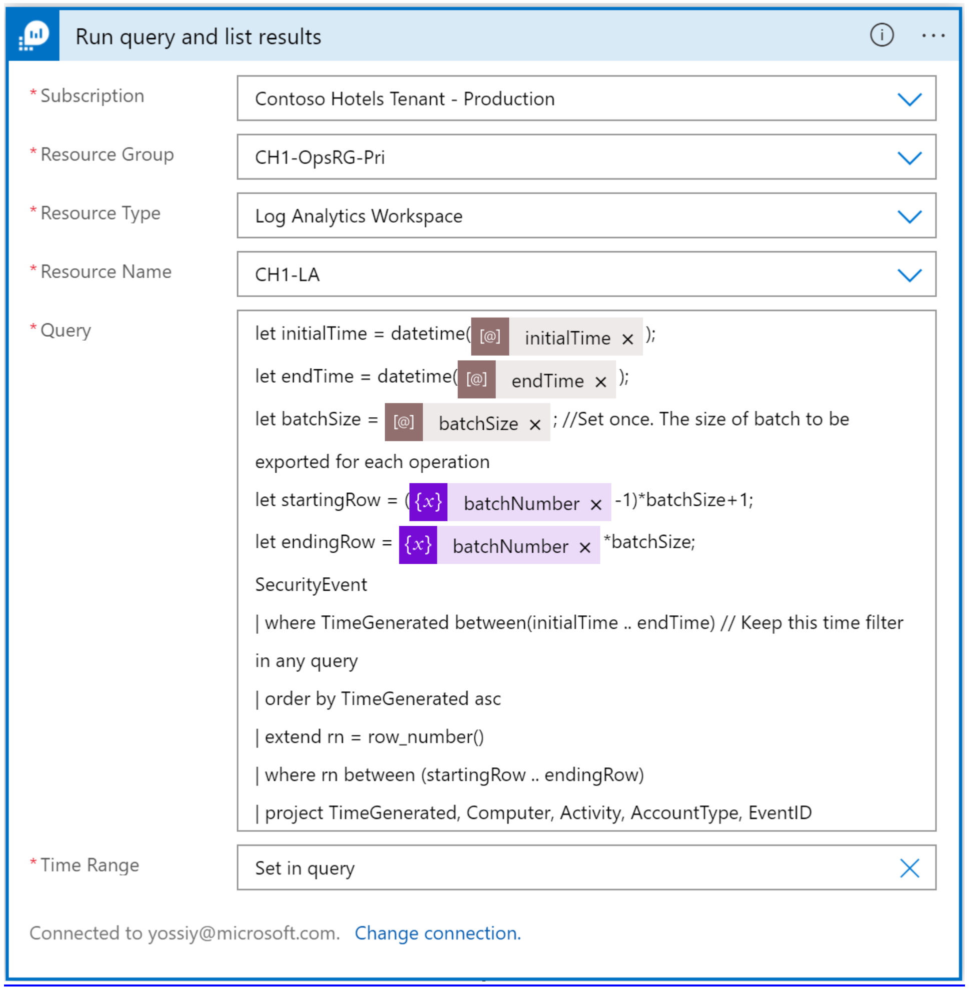
Timeout: PT6H

Add Azure Monitor Logs in Until control

Azure Monitor logs are stored in Log Analytics workspace in multitenancy environment. Like any multitenancy service, it has limits intended to protect customers and isolate them from “noisy neighbor” and for maintaining quality of service. When querying a large dataset, you should consider the following limits that affect how Logic App recurrence is configured:

- Log queries cannot return more than 500,000 rows.
- Log queries in Logic App cannot return more than 16,384,000 bytes.
- Log queries cannot run longer than 10 minutes by default.
- Log Analytics connector is limited to 100 call per minute.

You should filter and aggregate your data to reduce it to the minimum required for export. For example, if you need to archive sign-in events, you could filter on sign-in events and project only those fields that you need.


Run query and list results

* Subscription	Contoso Hotels Tenant - Production
* Resource Group	CH1-OpsRG-Pri
* Resource Type	Log Analytics Workspace
* Resource Name	CH1-LA

* Query

```
let initialTime = datetime( [@] initialTime × );
let endTime = datetime( [@] endTime × );
let batchSize = [@] batchSize × ; //Set once. The size of batch to be
exported for each operation
let startingRow = ( {x} batchNumber × -1)*batchSize+1;
let endingRow = {x} batchNumber × *batchSize;
SecurityEvent
| where TimeGenerated between(initialTime .. endTime) // Keep this time filter
in any query
| order by TimeGenerated asc
| extend rn = row_number()
| where rn between (startingRow .. endingRow)
| project TimeGenerated, Computer, Activity, AccountType, EventID
```

* Time Range: Set in query

Connected to yossiy@microsoft.com. Change connection.

The **Run query and list results** action runs a query in your Log Analytics workspace and returns the results as json.

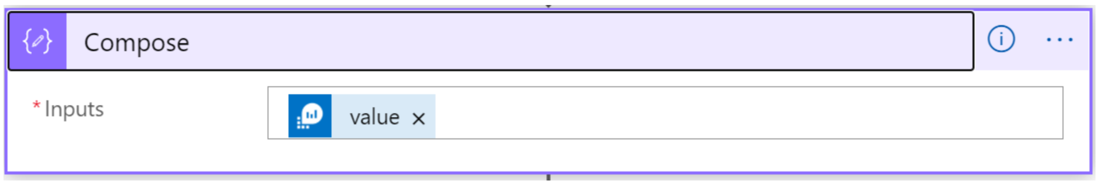
1. Click **New Step** to add an action.
2. Type “Azure Monitor” in the search box and the select the **Run query and list results** action.
3. You will be prompted to select a tenant and grant access to the Log Analytics workspace with the account that the workflow will use to run the query.
4. Specify the **Subscription**, **Resource Group**, **Resource Type**, **Resource Name (this is the Log Analytics workspace name)**.
5. Paste the query into the **Query** box. See below for details.
6. Set **Time Range** to ‘Set in query’

Query example for export of TimeGenerated, Computer, Activity, AccountType, EventID fields in SecurityEvent table starting at particular date and time:

```
let startTime = datetime(<startTime parameter>);
let endTime = datetime(<endTime parameter>);
let batchSize = <batchSize variable>;
let startingRow = (<batchNumber parameter>-1)*batchSize+1;
let endingRow = batchNumber*batchSize;
SecurityEvent
| where TimeGenerated between(startTime .. endTime)
| order by TimeGenerated asc
| extend rn = row_number()
| where rn between (startingRow .. endingRow)
| project TimeGenerated, Computer, Activity, AccountType, EventID
```

Add the Compose action in Until control

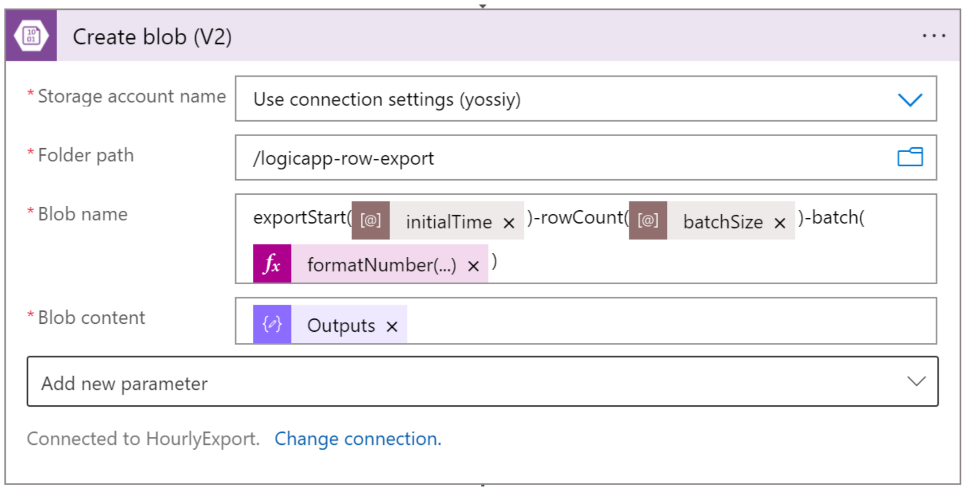
The **Compose** action takes the parsed JSON output and creates the object that we need to store in the blob.



1. Click **New Step** to add an action.
2. Type “compose” in the search box and the select the **Compose** action.
3. Click the **Inputs** box display a list of values from previous activities.
4. Select **Body** from the **Parse JSON** action. This is the parsed output from the log query.

Add the Create blob action in Until control

Use the **Create Blob** action to write the composed JSON to storage.


Create blob (V2)

* Storage account name	Use connection settings (yossiy)
* Folder path	/logicapp-row-export
* Blob name	exportStart(initialTime)-rowCount(batchSize)-batch(formatNumber(...))
* Blob content	Outputs

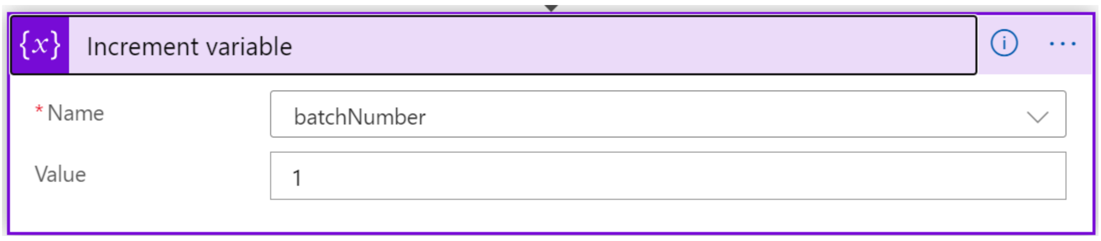
Add new parameter

Connected to HourlyExport. Change connection.

1. Click **New Step** to add an action.
2. Type "Azure blob" in the search box and the select the **Create blob** action.
3. Type a name for the connection to your storage account in **Connection Name**.
4. Click the folder icon in the **Folder path** box and select the container in your storage account.
5. Click the **Blob name** and enter this expression: exportStart(@{parameters('startTime')})-rows(@{parameters('batchSize')})-batch(@{formatNumber(variables('batchNumber'), '0000000', 'en-us')})
6. Click the **Blob content** box and select **Outputs** from the **Compose** section on the right.

Add Increment variable in Until control

The control increments the batchNumber after each batch export iteration. This is used to set the export cursor to aim to the next batch of rows.

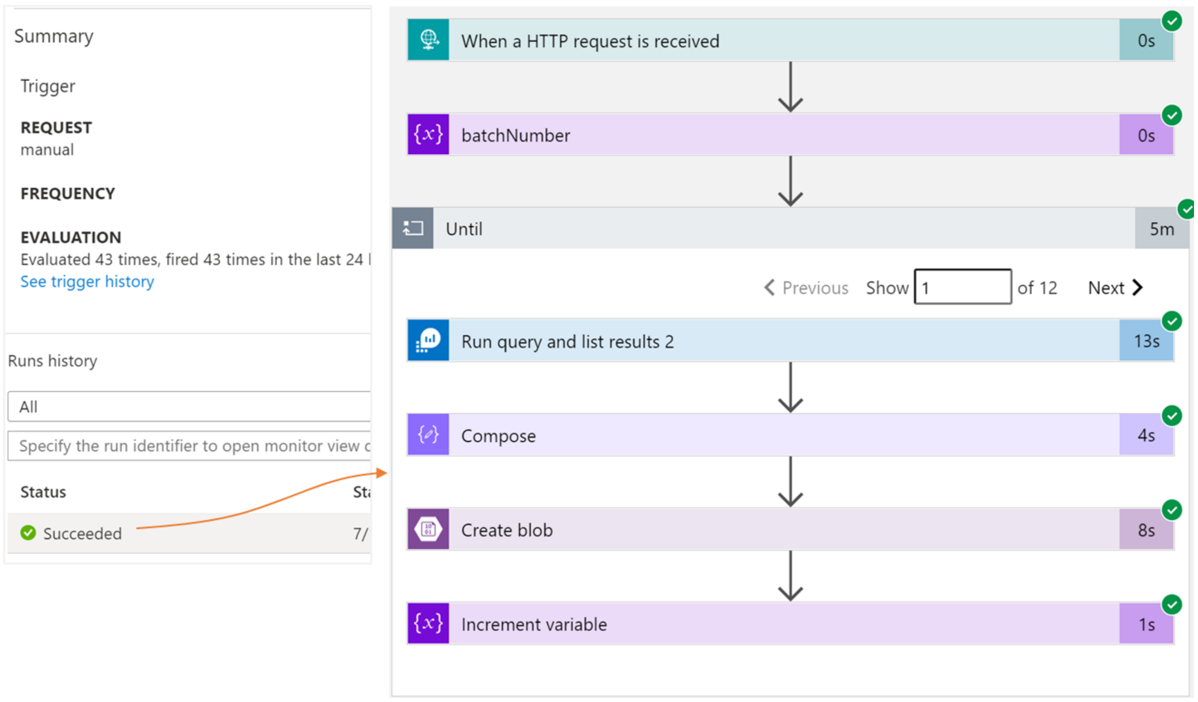

Increment variable

* Name	batchNumber
Value	1

Click **Save** to save the workflow.

Test the Logic App

Test the workflow by clicking **Run**. If the workflow has errors, it will be indicated on the step with the problem. You can view the executions and drill in to each step to view the input and output to investigate failures. See [Troubleshoot and diagnose workflow failures in Azure Logic Apps]() if necessary.



Summary

Trigger

REQUEST
manual

FREQUENCY

EVALUATION
Evaluated 43 times, fired 43 times in the last 24 l
See trigger history

Runs history

All

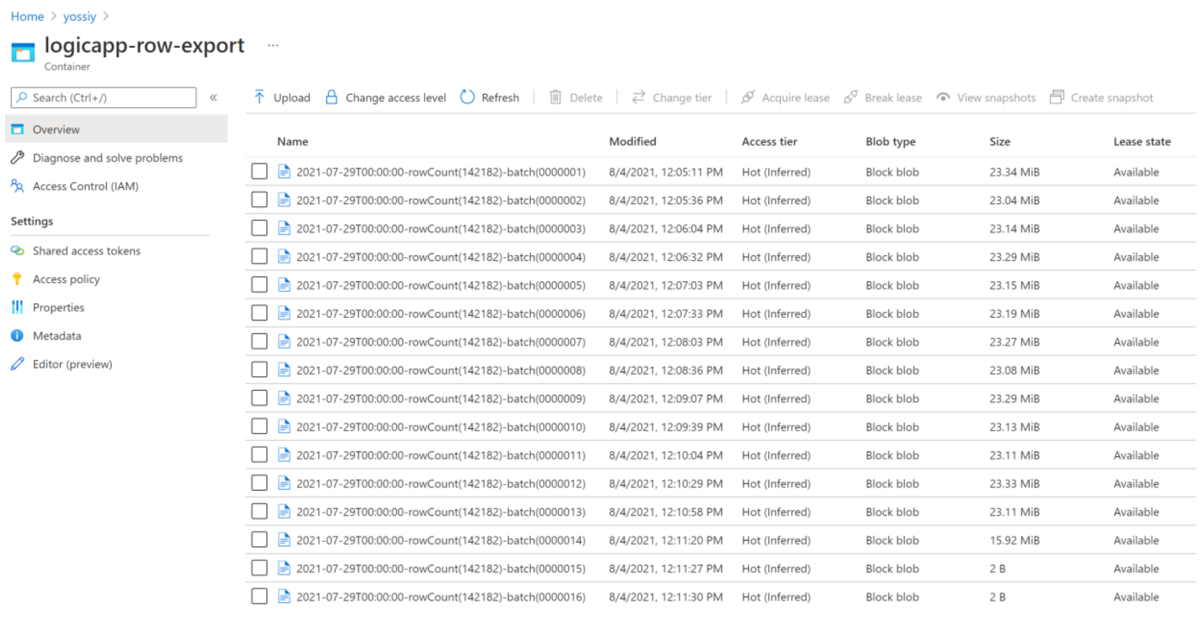
Specify the run identifier to open monitor view c

Status	Sta
Succeeded	7/

Step	Duration
When a HTTP request is received	0s
batchNumber	0s
Until	5m
Run query and list results 2	13s
Compose	4s
Create blob	8s
Increment variable	1s

View logs in Storage

1. Go to the **Storage accounts** menu in the Azure portal.
2. Select your storage account.
3. Click the **Blobs** tile.
4. Select the container you specified in the **Create blob** action.
5. Select one of the blobs then **download** to open.



Name	Modified	Access tier	Blob type	Size	Lease state
2021-07-29T00:00:00-rowCount(142182)-batch(0000001)	8/4/2021, 12:05:11 PM	Hot (Inferred)	Block blob	23.34 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000002)	8/4/2021, 12:05:36 PM	Hot (Inferred)	Block blob	23.04 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000003)	8/4/2021, 12:06:04 PM	Hot (Inferred)	Block blob	23.14 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000004)	8/4/2021, 12:06:32 PM	Hot (Inferred)	Block blob	23.29 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000005)	8/4/2021, 12:07:03 PM	Hot (Inferred)	Block blob	23.15 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000006)	8/4/2021, 12:07:33 PM	Hot (Inferred)	Block blob	23.19 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000007)	8/4/2021, 12:08:03 PM	Hot (Inferred)	Block blob	23.27 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000008)	8/4/2021, 12:08:36 PM	Hot (Inferred)	Block blob	23.08 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000009)	8/4/2021, 12:09:07 PM	Hot (Inferred)	Block blob	23.29 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000010)	8/4/2021, 12:09:39 PM	Hot (Inferred)	Block blob	23.13 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000011)	8/4/2021, 12:10:04 PM	Hot (Inferred)	Block blob	23.11 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000012)	8/4/2021, 12:10:29 PM	Hot (Inferred)	Block blob	23.33 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000013)	8/4/2021, 12:10:58 PM	Hot (Inferred)	Block blob	23.11 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000014)	8/4/2021, 12:11:20 PM	Hot (Inferred)	Block blob	15.92 MiB	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000015)	8/4/2021, 12:11:27 PM	Hot (Inferred)	Block blob	2 B	Available
2021-07-29T00:00:00-rowCount(142182)-batch(0000016)	8/4/2021, 12:11:30 PM	Hot (Inferred)	Block blob	2 B	Available

Notes

- The size in the workspace (_BilledSize) and blob are calculated differently since blog size includes JSON envelop. It's expected that the size you see in blobs is greater.
- An assurance that all records were exported is when last blobs are empty and size is 2B.

Logic App template

```
{
  "definition": {
    "$schema": "https://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-06-01/workflowdefinition.json#",
    "actions": {
      "Until": {
        "actions": {
          "Compose": {
            "inputs": "@body('Run_query_and_list_results_2')?['value']",
            "runAfter": {
              "Run_query_and_list_results_2": [
                "Succeeded"
              ]
            },
            "type": "Compose"
          },
          "Create_blob_(V2)_2": {
            "inputs": {
              "body": "@outputs('Compose')",
              "headers": {
                "ReadFileMetadataFromServer": true
              },
              "host": {
                "connection": {
                  "name": "@parameters('$connections')['azureblob_2']['connectionId']"
                }
              },
              "method": "post",
                            "path": "/v2/datasets/@{encodeURIComponent(encodeURIComponent('yossiy'))}/files",
                            "queries": {
                                "folderPath": "/logicapp-row-export",
                                "name": "exportStart(@{parameters('startTime')})-rows(@{parameters('batchSize')})-
batch(@{formatNumber(variables('batchNumber'), '0000000', 'en-us')})",
                                "queryParametersSingleEncoded": true
                            }
                        },
                        "runAfter": {
                            "Compose": [
                                "Succeeded"
                            ]
                        },
                        "runtimeConfiguration": {
                            "contentTransfer": {
                                "transferMode": "Chunked"
                            }
                        },
                        "type": "ApiConnection"
                    },
                    "Increment_variable": {
                        "inputs": {
                            "name": "batchNumber",
                            "value": 1
                        },
                        "runAfter": {
                            "Create_blob_(V2)_2": [
                                "Succeeded"
                            ]
                        },
                        "type": "IncrementVariable"
                    },
                    "Run_query_and_list_results_2": {
                        "inputs": {
                            "body": "let startTime = datetime(@{parameters('startTime')}); \nlet endTime = datetime(@{parameters('endTime')}); \nlet batch
Size = @{parameters('batchSize')}; //Set once. The size of batch to be exported for each operation\nlet startingRow = (@{variables('batchNumber
')}-
1)*batchSize+1;\nlet endingRow = @{variables('batchNumber')}*batchSize;\nSecurityEvent\n| where TimeGenerated between(startTime .. endTi
me) // Keep this time filter in any query\n| order by TimeGenerated asc\n| extend rn = row_number()\n| where rn between (startingRow .. endi
ngRow)\n| project TimeGenerated, Computer, Activity, AccountType, EventID",
                            "host": {
                                "connection": {
                                    "name": "@parameters('$connections')['azuremonitorlogs_1']['connectionId']"
                                }
                            },
                            "method": "post",
                            "path": "/queryData",
                            "queries": {
                                "resourcegroups": "CH1-OpsRG-Pri",
                                "resourcename": "CH1-LA",
                                "resourcetype": "Log Analytics Workspace",
                                "subscriptions": "ebb79bc0-aa86-44a7-8111-cabbe0c43993",
                                "timerange": "Set in query"
                            }
                        },
                        "runAfter": {},
                        "type": "ApiConnection"
                    }
                },
                "expression": "@greater(variables('batchNumber'), parameters('batchRuns'))",
                "limit": {
                    "count": 10000,
                    "timeout": "PT6H"
                },
                "runAfter": {
                    "batchNumber": [
                        "Succeeded"
                    ]
                },
                "type": "Until"
            },
            "batchNumber": {
                "inputs": {
                    "variables": [
                        {
                            "name": "batchNumber",
                            "type": "integer",
                            "value": 1
                        }
                    ]
                },
                "runAfter": {},
                "type": "InitializeVariable"
            }
        },
        "contentVersion": "1.0.0.0",
        "outputs": {},
        "parameters": {
            "$connections": {
                "defaultValue": {},
                "type": "Object"
            },
            "batchRuns": {
                "defaultValue": 80,
                "type": "Int"
            },
            "batchSize": {
                "defaultValue": 142182,
                "type": "Int"
            },
            "endTime": {
                "defaultValue": "2021-08-02T15:32:23.786",
                "type": "String"
            },
            "startTime": {
                "defaultValue": "2021-07-29T00:00:00",
                "type": "String"
            }
        },
        "triggers": {
            "manual": {
                "inputs": {
                    "schema": {}
                },
                "kind": "Http",
                "type": "Request"
            }
        }
    },
    "parameters": {
        "$connections": {
            "value": {
                "azureblob_2": {
                    "connectionId": "/subscriptions/b1550e13-52b0-4ed1-8a7a-
    8929cee47022/resourceGroups/yossiy/providers/Microsoft.Web/connections/azureblob",
                    "connectionName": "azureblob",
                    "id": "/subscriptions/b1550e13-52b0-4ed1-8a7a-8929cee47022/providers/Microsoft.Web/locations/eastus/managedApis/azureblob"
                },
                "azuremonitorlogs_1": {
                    "connectionId": "/subscriptions/b1550e13-52b0-4ed1-8a7a-
    8929cee47022/resourceGroups/yossiy/providers/Microsoft.Web/connections/azuremonitorlogs-4",
                    "connectionName": "azuremonitorlogs-4",
                    "id": "/subscriptions/b1550e13-52b0-4ed1-8a7a-
8929cee47022/providers/Microsoft.Web/locations/eastus/managedApis/azuremonitorlogs"
                }
            }
        }
    }
}
```