Calculate 99.9% SLA for failed activities

Last updated by | Supreeth Vasudevamurthy | Mar 27, 2023 at 9:07 AM PDT

Contents

Calculation of SLA

SLA is calculated at the Data factory level.

Total Activity Runs - Total number of Activity Runs attempted during a given billing month for a given Microsoft Azure subscription.

System Errored Runs - Total number of Activity Runs failed due to failure of an ADF component (Azure Networking, logic apps, Synapse Spark etc) or an ADF outage

```
Monthly uptime % = (Total Activity Runs - System Errored runs)/Total Activity Runs
```

To find Customer's billing Start date and billing end date, relative to the Outage date, run below query. Change **dateOfOutage** and **SubscriptionId** accordingly

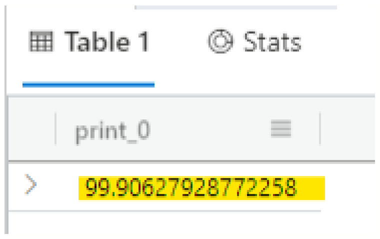
Please click on this link to get ARM cluster access

```
let dateOfOutage = datetime(2022-10-14 00:00:00.0000000);//Outage date
let startDate = toscalar(
cluster('https://armprod.kusto.windows.net').database('CosmosToKusto').Subscriptions
| where SubscriptionId contains "<Subscription_id>"
| distinct CreatedTime | top 1 by CreatedTime asc);
print billing_startDate = make_datetime(getyear(dateOfOutage), monthofyear(dateOfOutage), dayofmonth(startDate
billing_endDate = datetime_add('month',1,datetime_add('day',-1,make_datetime(getyear(dateOfOutage), monthofyea
```

Below Query will give an **SLA on the Data Factory level** and **information on dates for which you need to run Jarvis logs for**, input is either an Activity Run Id or a Pipeline Run Id

```
let runid = "afea17b6-d7be-4809-97a7-d3f2233fbdb6";//This can be activity Run Id or pipeline Run Id
let DataFactoryId = toscalar(
cluster('adfcus.kusto.windows.net').database('AzureDataFactory').ActivityRuns
|union cluster('adfneu.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| where activityRunId == runid or pipelineRunId  ==  runid
| distinct dataFactoryId
|take 1);
let activityruns =
cluster('adfcus.kusto.windows.net').database('AzureDataFactory').ActivityRuns
|union cluster('adfneu.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| where dataFactoryId =~ DataFactoryId
| where category == "ActivityRuns"
| where status !in ("Queued", "InProgress")
| extend dayofrun = startofday(PreciseTimeStamp)
| distinct activityRunId, status, failureType, PreciseTimeStamp,dayofrun ;
let totalruns = toscalar(activityruns| summarize totalcount = dcount(activityRunId));
let systemfailure = toscalar(activityruns |where failureType== "SystemError" | summarize failurecount = dcount
let timestamp_max = toscalar(activityruns | summarize arg_max(PreciseTimeStamp, activityRunId));
let timestamp_min = toscalar(activityruns | summarize arg_min(PreciseTimeStamp, activityRunId));
let dayDiff = datetime_diff('day', timestamp_max, timestamp_min);
print TotalRuns = totalruns, SystemErrorFailureRuns = systemfailure , SLA= round((((todouble(totalruns) - todo
,Information = strcat("SLA is calculated between the dates ", format_datetime(timestamp_min,"yyyy-MM-dd"), " a
" to ",format_datetime(datetime_add('day',-1,timestamp_min),"yyyy-MM-dd")," in jarvis" );
```

Above query will give results like below.



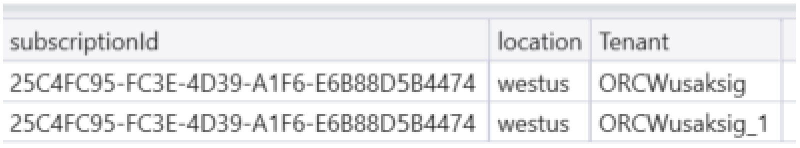
```
99.90627928772258
```

Further, If you need data for more than 21 days, use Jarvis and follow below steps.

Usually Jarvis queries throttle if the data exceeds 500000 rows, to avoid this **add Tenant and Region information under Scoping conditions**, run below query to get Tenant and Region information. Activity Run Id or Pipeline Run Id of the Customer is the input for the below query.

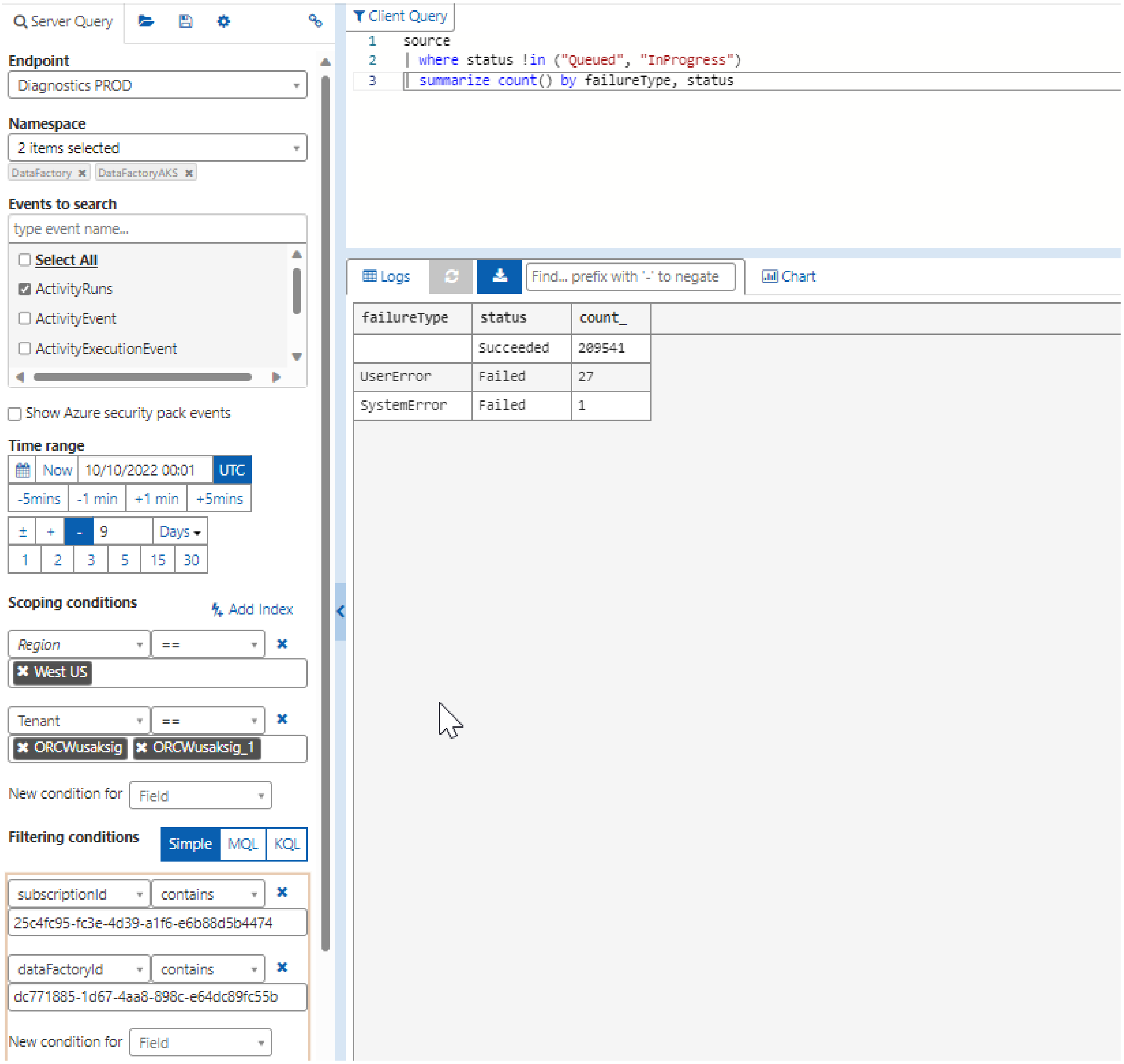
```
let runid = "afea17b6-d7be-4809-97a7-d3f2233fbdb6";
let datafactory = toscalar(cluster('adfcus.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| union cluster('adfneu.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| where activityRunId == runid or pipelineRunId  == runid
| project dataFactoryId);
cluster('adfcus.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| union cluster('adfneu.kusto.windows.net').database('AzureDataFactory').ActivityRuns
| where dataFactoryId == datafactory
| where status !in ('Queued', 'InProgress')
| extend failureType = coalesce(failureType, 'Passed') | distinct subscriptionId, location, Tenant
```

Below is the result, add all location and Tenant information in jarvis under region and Tenant respectively, under Scoping conditions.



subscriptionId	location	Tenant
25C4FC95-FC3E-4D39-A1F6-E6B88D5B4474	westus	ORCWusaksig
25C4FC95-FC3E-4D39-A1F6-E6B88D5B4474	westus	ORCWusaksig_1

Same is shown in the below screenshot



Endpoint: Diagnostics PROD

Namespace: 2 items selected (DataFactory, DataFactoryAKS)

Events to search: ActivityRuns [✓]; ActivityEvent; ActivityExecutionEvent

Show Azure security pack events

Time range: Now 10/10/2022 00:01 UTC; - 9 Days

Scoping conditions:
Region == West US
Tenant == ORCWusaksig, ORCWusaksig_1

Filtering conditions (Simple):
subscriptionId contains 25c4fc95-fc3e-4d39-a1f6-e6b88d5b4474
dataFactoryId contains dc771885-1d67-4aa8-898c-e64dc89fc55b

```
source
| where status !in ("Queued", "InProgress")
| summarize count() by failureType, status
```

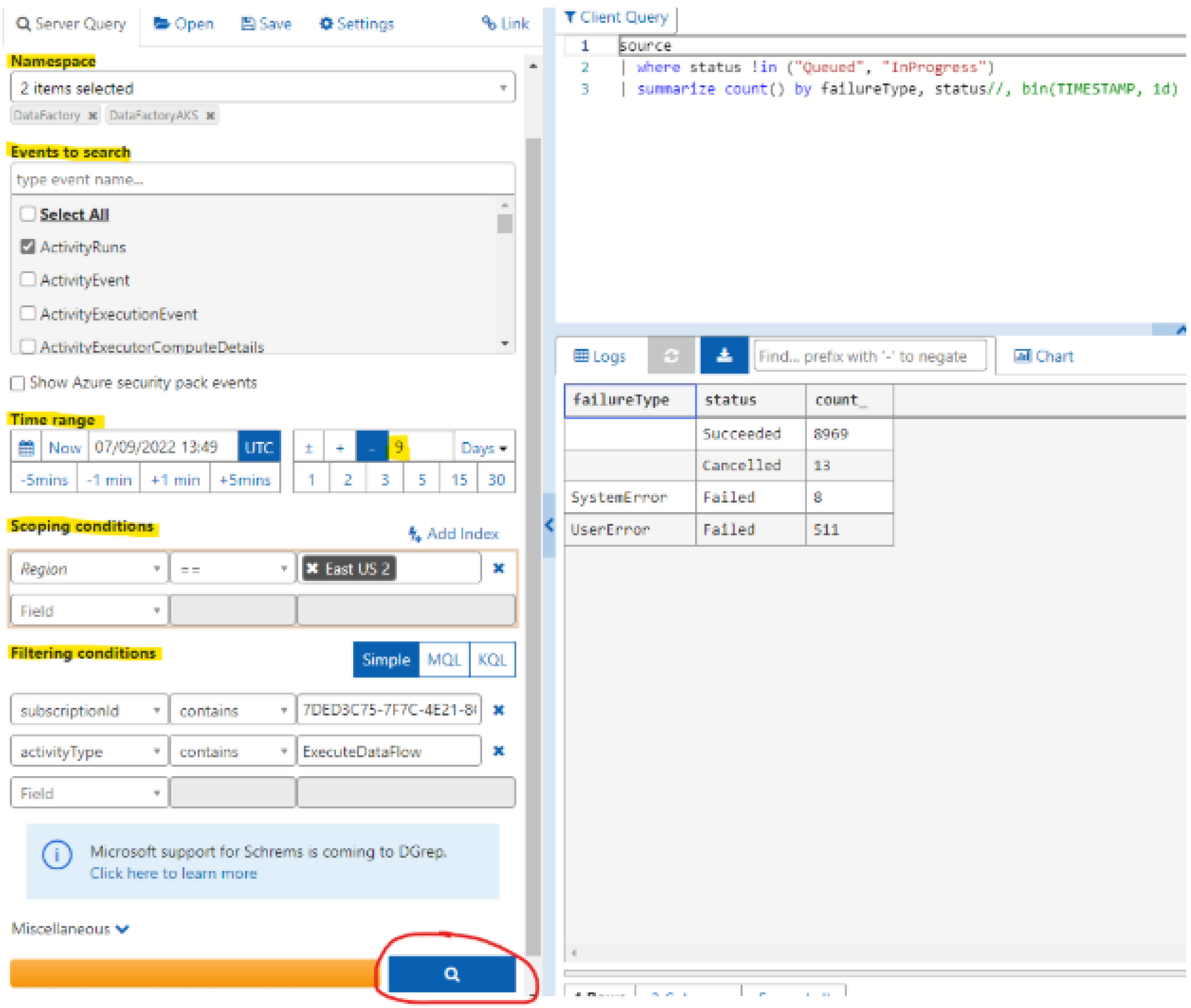
failureType	status	count_
	Succeeded	209541
UserError	Failed	27
SystemError	Failed	1

Kusto will give 21 days' worth of data. Then, for the 9 days before that, run the below in Jarvis, then add those results to above results for completed results. Jarvis link with query: https://portal.microsoftgeneva.com/s/440A19DB

For filtering conditions, Subscription ID, ActivityType, and region can be found in ActivityRuns table for run ID customer provided.

The Time range needs to be set to 22 days ago minus 9 days (since Kusto gives us results for last 21 days).

Below is example screenshot for how Jarvis should look (filtering conditions would need to be changed per the circumstance).



Namespace: 2 items selected — DataFactory, DataFactoryAKS

Events to search: type event name...

- [ ] Select All
- [x] ActivityRuns
- [ ] ActivityEvent
- [ ] ActivityExecutionEvent
- [ ] ActivityExecutorComputeDetails

- [ ] Show Azure security pack events

Time range: Now 07/09/2022 13:49 UTC − 9 Days

Scoping conditions: Region == East US 2

Filtering conditions (Simple):

subscriptionId	contains	7DED3C75-7F7C-4E21-8
activityType	contains	ExecuteDataFlow

Microsoft support for Schrems is coming to DGrep. Click here to learn more

Miscellaneous

```
source
| where status !in ("Queued", "InProgress")
| summarize count() by failureType, status//, bin(TIMESTAMP, 1d)
```

failureType	status	count_
	Succeeded	8969
	Cancelled	13
SystemError	Failed	8
UserError	Failed	511

Example: You'll get results similar to below.

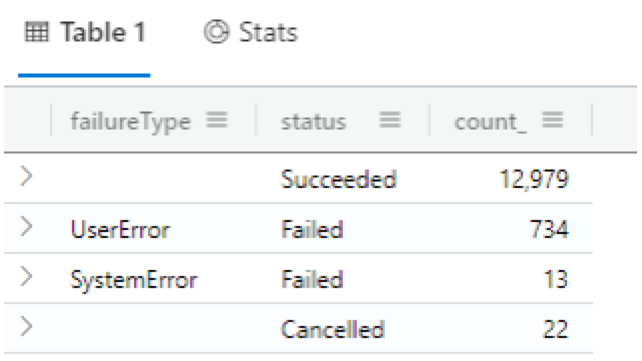


Table 1
Stats

failureType	status	count_
	Succeeded	12,979
UserError	Failed	734
SystemError	Failed	13
	Cancelled	22

SLA Calculation: (Total requests – System Errors) / Total requests.

SLA calculation: (13,748 - 13) / 13,748 = 0.99905441

SLA = 99.905

Final result:

```
Total Monthly uptime % = (Kusto Total Activity Runs + Jarvis Total Activity Runs - Kusto System Errored runs -
```

For Eg:

lets say Kusto has 999 activity runs and zero system Errored runs

lets say Jarvis has 1 activity runs and it failed(system Errored runs)

Total Monthly uptime % = (999 +1 - 0 - 1)/(999 + 1)=999/1000=0.999

SLA =99.9%

FAQs

Q: what if the logs have rolled over and the data is not there in Jarvis or Kusto?

A : Might need to check the outage which is related to Customer's issue. Assess the monthly deviation from usual Customer bill, engage relevant PG to see what best can be done for the customer.