Creating a filter with Sumo Logic's New Dashboard

dev.classmethod.jp/articles/sumo-logic-filter_in_newdashboard

酒井剛

July 7, 2022

I want to visualize AWS CloudTrail logs. I operate multiple accounts, and I need to check which users in each account logged in to the AWS console and how many times.

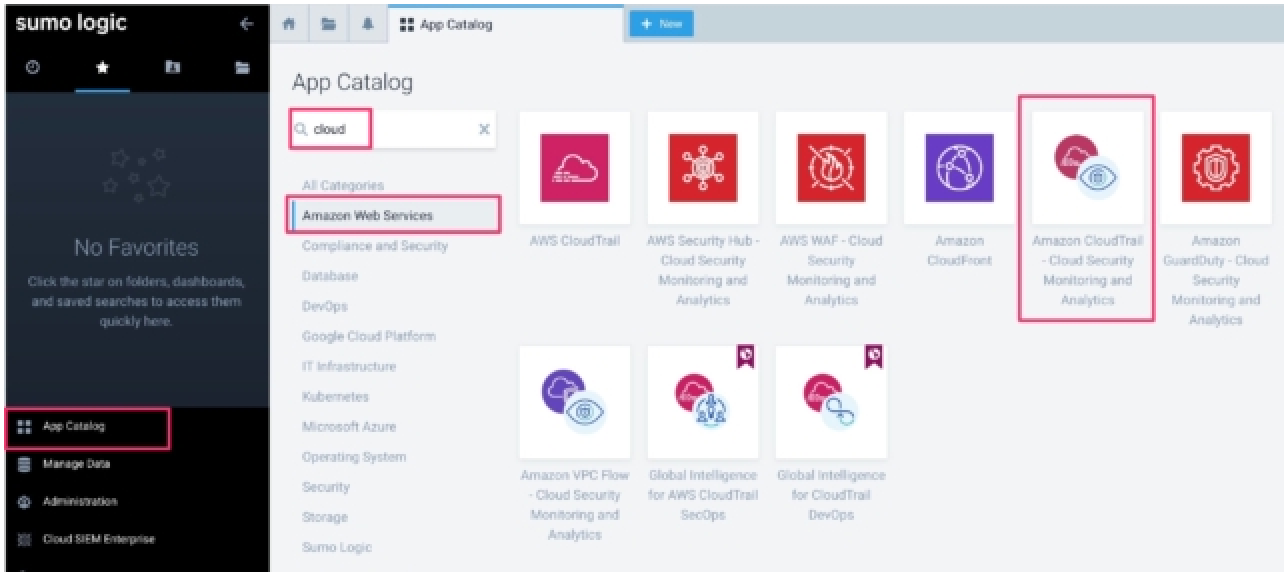
We will meet this requirement by slightly customizing Sumo Logic's built-in app.

Dashboard (App) that visualizes CloudTrail logs

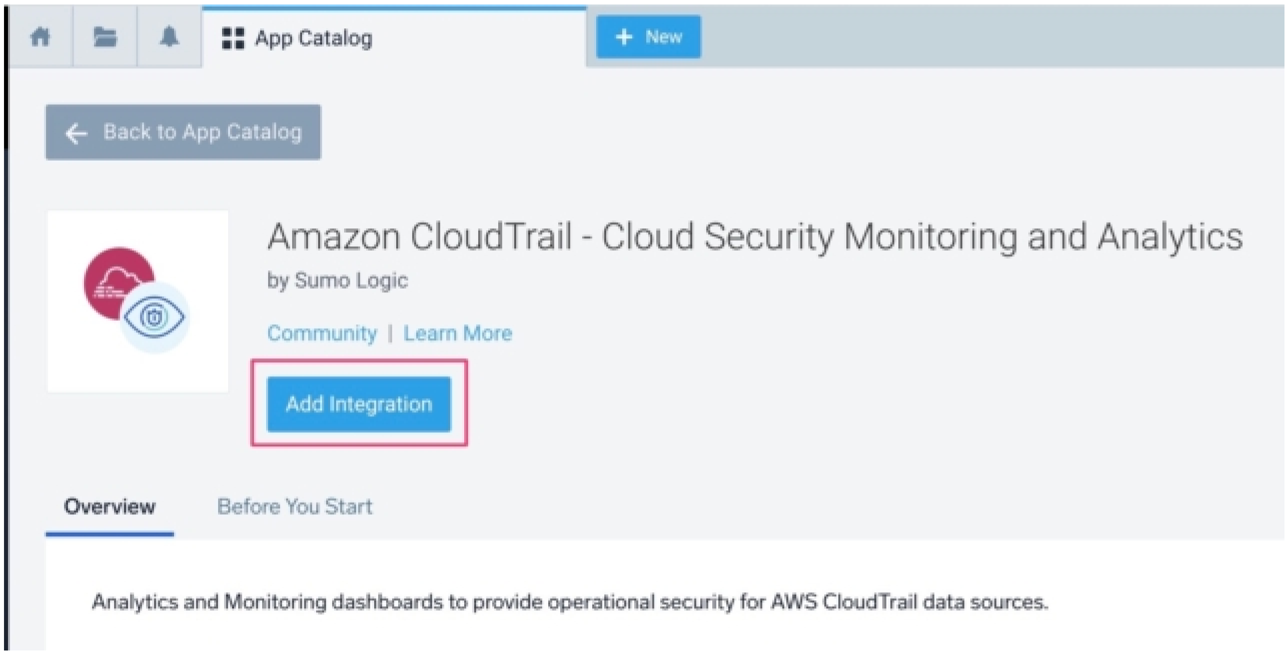
Sumo Logic offers over 240 (as of July 2022) apps, built-in dashboards and search query sets for major infrastructure and applications, so you can start analyzing immediately after setting up log ingestion.

This time, we will install "Amazon CloudTrail - Cloud Security Monitoring and Analytics," which can be used for CloudTrail logs, and check whether it meets the requirements. (For information on setting up CloudTrail logs from S3 to be imported into Sumo Logic, please refer to
this blog .)

Select App Catalog from the left menu, `Amazon Web Services`select a category, or `cloud`search in the search box, and install **"Amazon CloudTrail - Cloud Security Monitoring and Analytics."**


sumo logic
App Catalog
New
No Favorites
Click the star on folders, dashboards, and saved searches to access them quickly here.
App Catalog
Manage Data
Administration
Cloud SIEM Enterprise
cloud
All Categories
Amazon Web Services
Compliance and Security
Database
DevOps
Google Cloud Platform
IT Infrastructure
Kubernetes
Microsoft Azure
Operating System
Security
Storage
Sumo Logic
AWS CloudTrail
AWS Security Hub - Cloud Security Monitoring and Analytics
AWS WAF - Cloud Security Monitoring and Analytics
Amazon CloudFront
Amazon CloudTrail - Cloud Security Monitoring and Analytics
Amazon GuardDuty - Cloud Security Monitoring and Analytics
Amazon VPC Flow - Cloud Security Monitoring and Analytics
Global Intelligence for AWS CloudTrail SecOps
Global Intelligence for CloudTrail DevOps

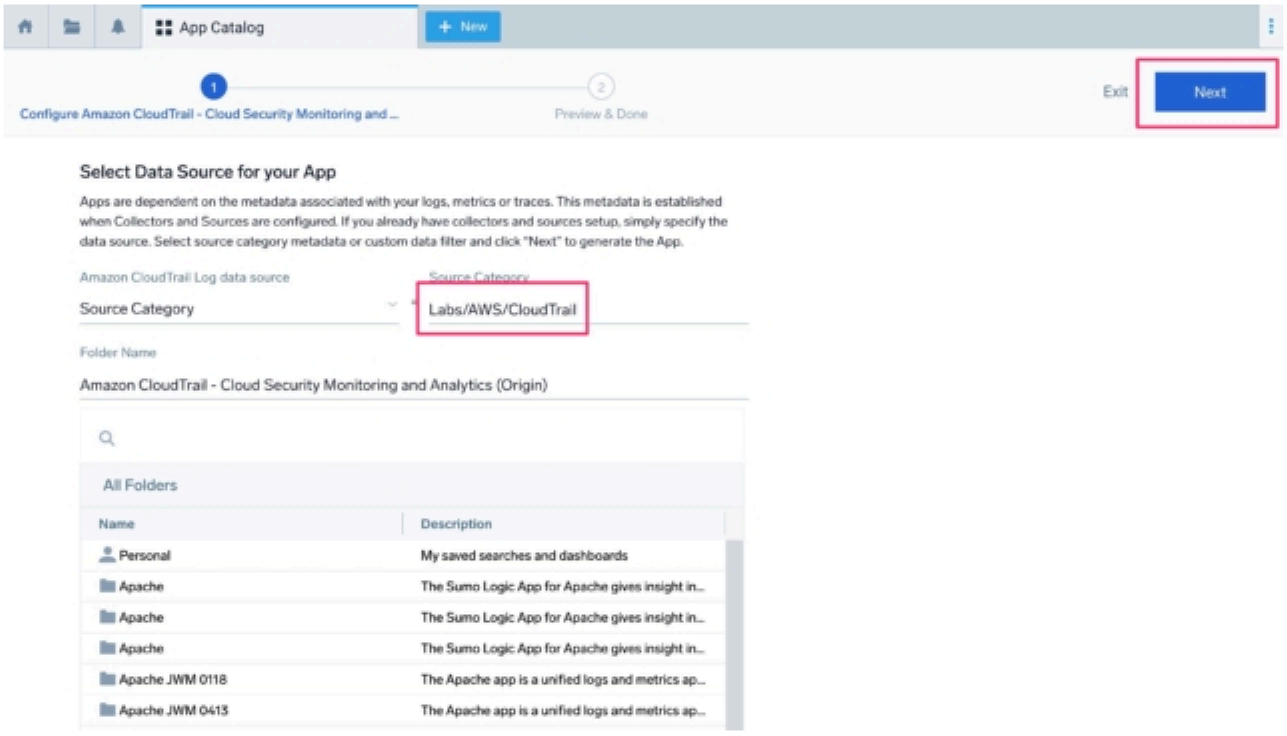
Select Add Integration.



Analytics and Monitoring dashboards to provide operational security for AWS CloudTrail data sources.

Specify the Source Category. *Source Category is like a group of logs that can be specified arbitrarily when configuring data collection. We recommend that you design it carefully when using it in production. For information on designing Source Categories, please refer to here

. If you are creating a new Folder Name, enter a new one, or if you want to deploy it in an existing folder, select it from the menu at the bottom, and then proceed to Next.


App Catalog
+ New
1
Configure Amazon CloudTrail - Cloud Security Monitoring and ...
2
Preview & Done
Exit
Next

Select Data Source for your App

Apps are dependent on the metadata associated with your logs, metrics or traces. This metadata is established when Collectors and Sources are configured. If you already have collectors and sources setup, simply specify the data source. Select source category metadata or custom data filter and click "Next" to generate the App.

Amazon CloudTrail Log data source: Source Category

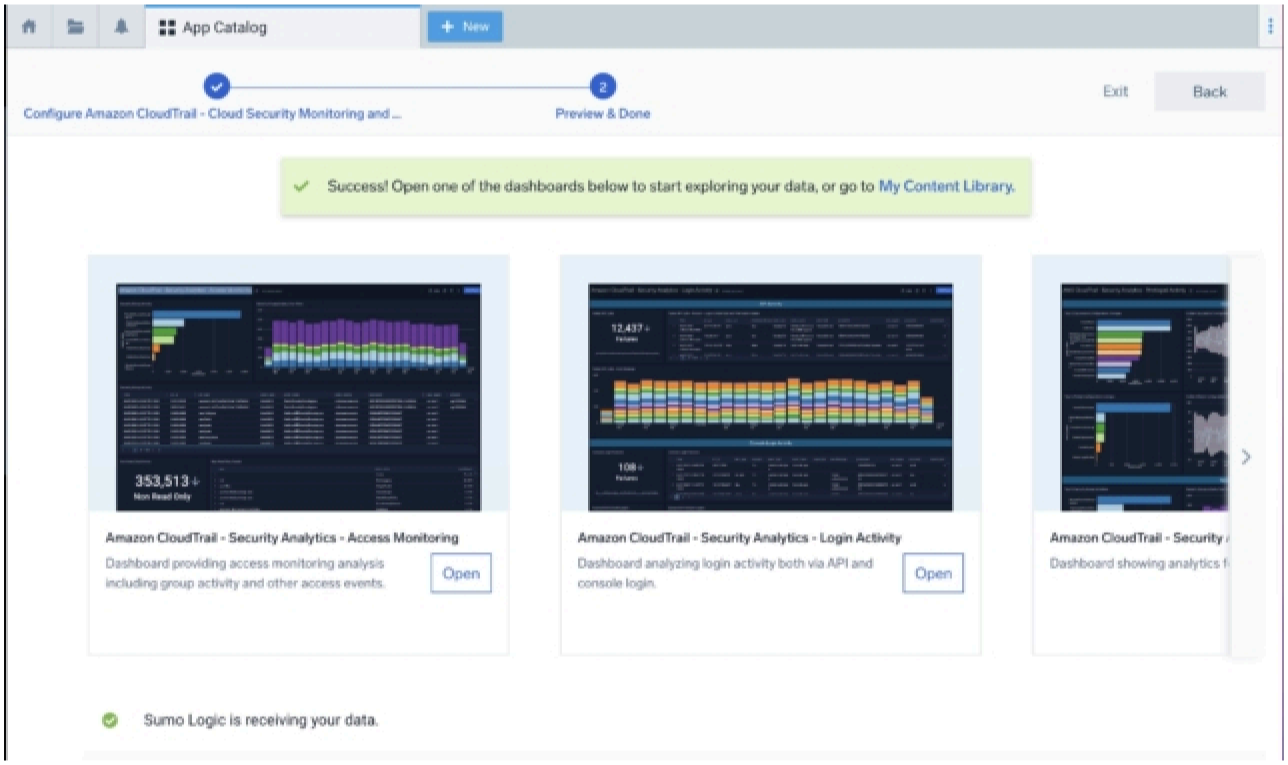
Source Category: Labs/AWS/CloudTrail

Folder Name: Amazon CloudTrail - Cloud Security Monitoring and Analytics (Origin)

All Folders

Name	Description
Personal	My saved searches and dashboards
Apache	The Sumo Logic App for Apache gives insight in...
Apache	The Sumo Logic App for Apache gives insight in...
Apache	The Sumo Logic App for Apache gives insight in...
Apache JWM 0118	The Apache app is a unified logs and metrics ap...
Apache JWM 0413	The Apache app is a unified logs and metrics ap...

If you see the screen below, the app installation is complete.


App Catalog
+ New
Configure Amazon CloudTrail - Cloud Security Monitoring and ...
2
Preview & Done
Exit
Back
Open
Open

Success! Open one of the dashboards below to start exploring your data, or go to My Content Library.

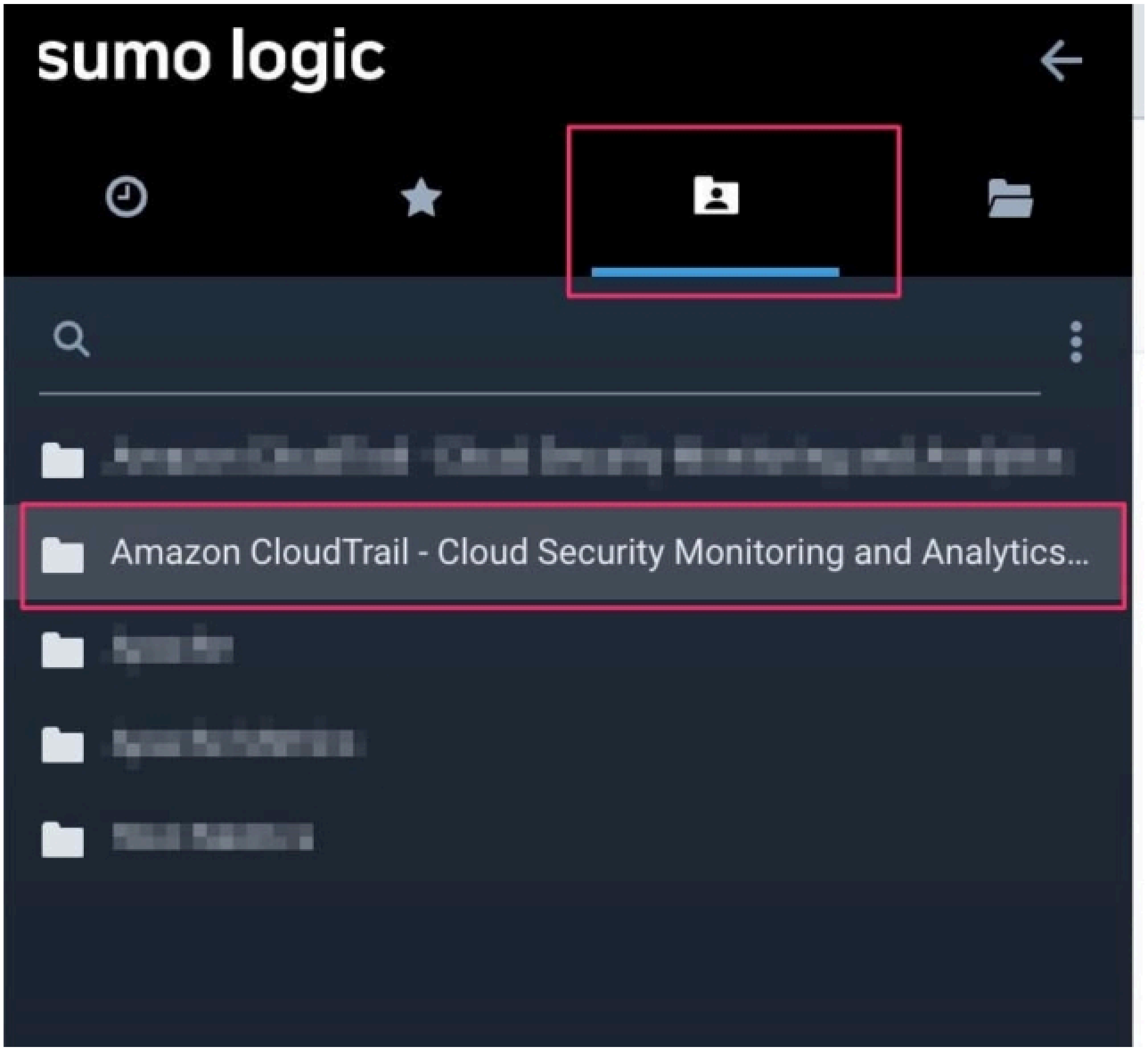
Amazon CloudTrail - Security Analytics - Access Monitoring
Dashboard providing access monitoring analysis including group activity and other access events.

Amazon CloudTrail - Security Analytics - Login Activity
Dashboard analyzing login activity both via API and console login.

Amazon CloudTrail - Security
Dashboard showing analytics f

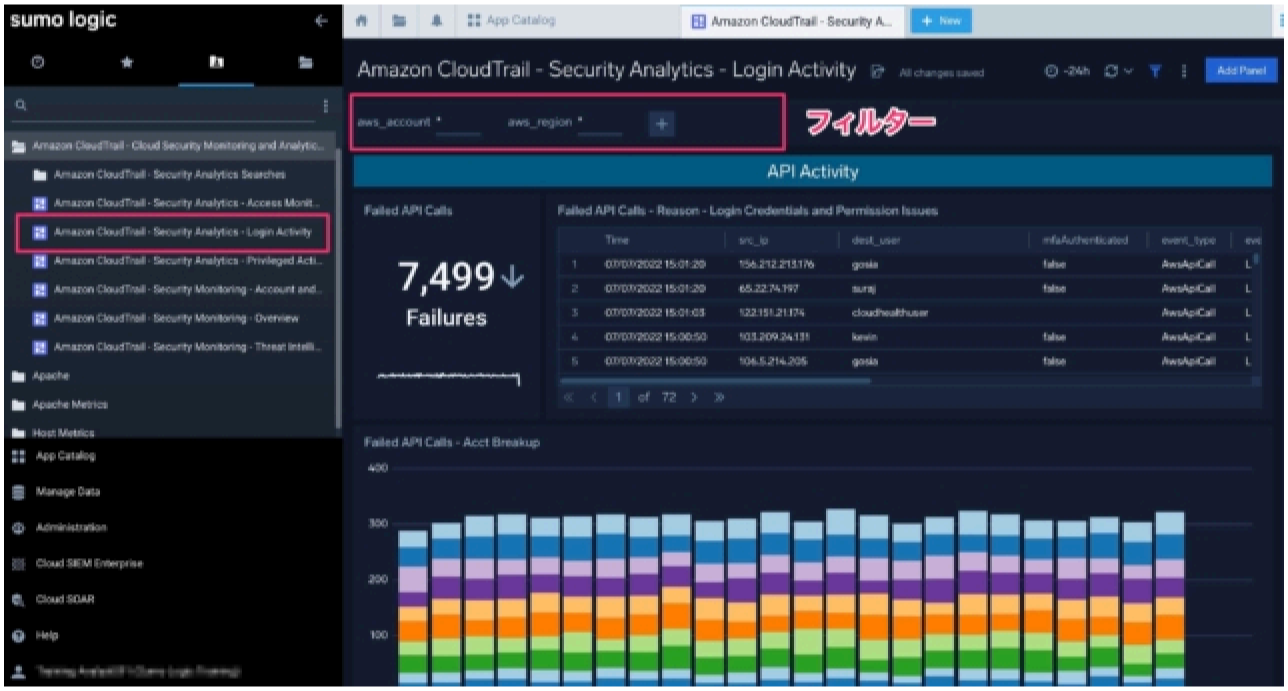
Sumo Logic is receiving your data.

Check that the app has been created in the Personal Folder on the left menu. *If it is not reflected, please refresh the screen.


sumo logic
Amazon CloudTrail - Cloud Security Monitoring and Analytics...

Use the New Dashboard filters to check the number of logins to the AWS Management Console by account and user.

Expand the folder of the installed app and open the dashboard **(Amazon CloudTrail - Security Analytics - Login Activity) where you can check login status to the AWS Management Console.**

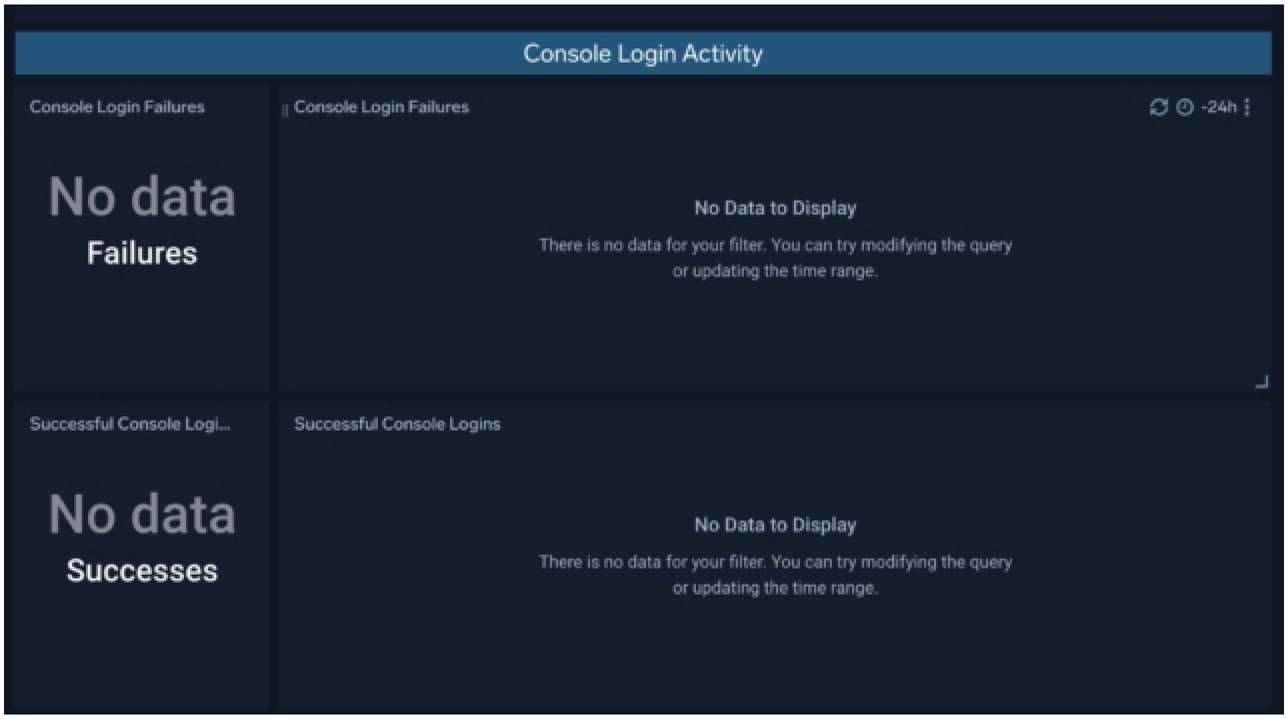

sumo logic
App Catalog
Amazon CloudTrail - Security A...
+ New
Amazon CloudTrail - Security Analytics - Login Activity
All changes saved
-24h
Add Panel
aws_account *
aws_region *
フィルター
Amazon CloudTrail - Cloud Security Monitoring and Analytic...
Amazon CloudTrail - Security Analytics Searches
Amazon CloudTrail - Security Analytics - Access Monit...
Amazon CloudTrail - Security Analytics - Login Activity
Amazon CloudTrail - Security Analytics - Privileged Acti...
Amazon CloudTrail - Security Monitoring - Account and...
Amazon CloudTrail - Security Monitoring - Overview
Amazon CloudTrail - Security Monitoring - Threat Intelli...
Apache
Apache Metrics
Host Metrics
App Catalog
Manage Data
Administration
Cloud SIEM Enterprise
Cloud SOAR
Help
API Activity
Failed API Calls
7,499
Failures
Failed API Calls - Acct Breakup
400
300
200
100

Failed API Calls - Reason - Login Credentials and Permission Issues

	Time	src_ip	dest_user	mfaAuthenticated	event_type
1	07/07/2022 15:01:20	156.212.213.176	gosia	false	AwsApiCall
2	07/07/2022 15:01:20	65.22.74.197	suraj	false	AwsApiCall
3	07/07/2022 15:01:03	122.151.21.174	cloudhealthuser		AwsApiCall
4	07/07/2022 15:00:50	103.209.24.131	kevin	false	AwsApiCall
5	07/07/2022 15:00:50	106.5.214.205	gosia	false	AwsApiCall

1 of 72

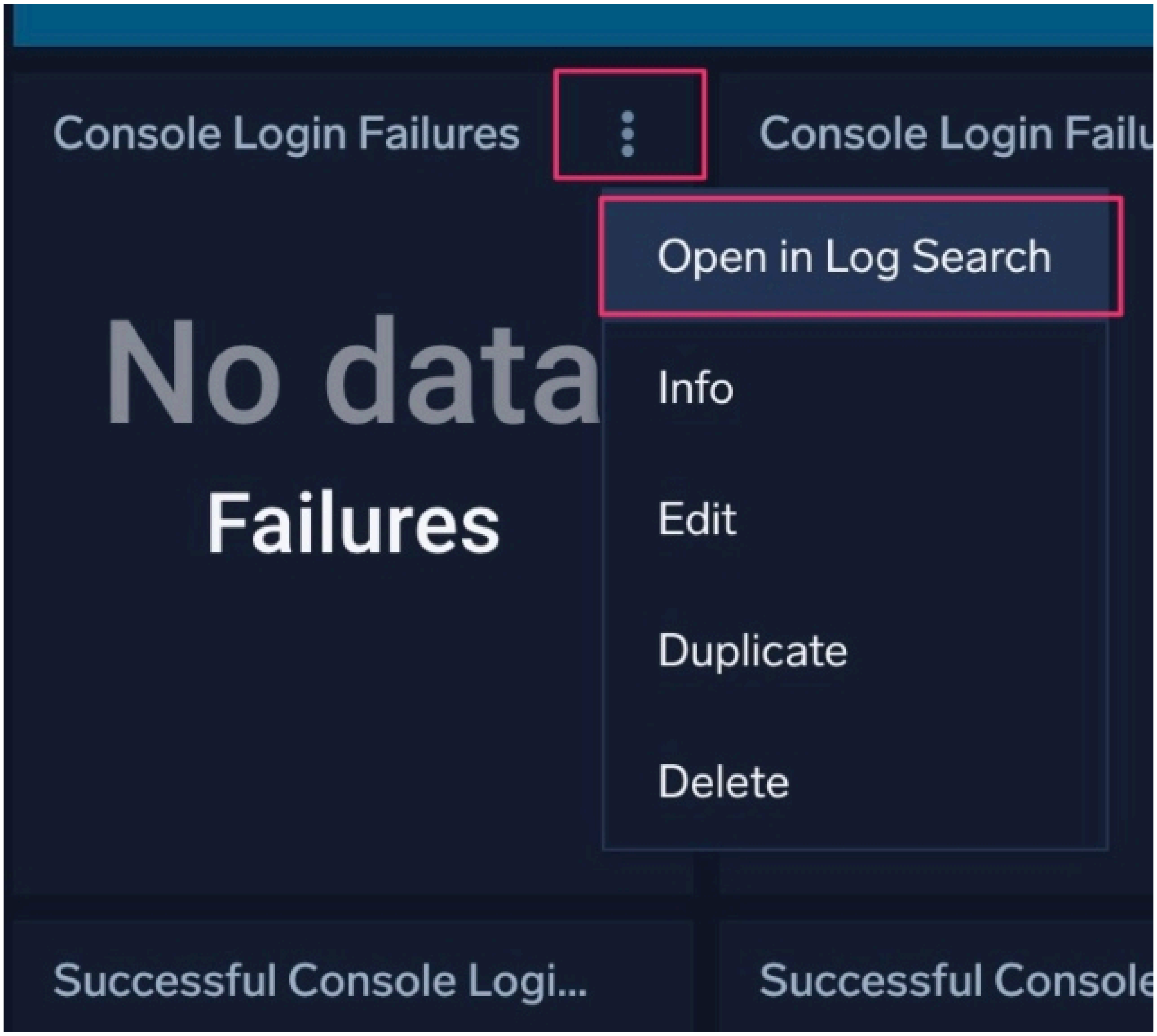
There is a section called **"Console Login Activity,"** but unfortunately, it is not visualized.


Console Login Activity
Console Login Failures
No data
Failures
Console Login Failures
-24h
No Data to Display
There is no data for your filter. You can try modifying the query or updating the time range.
Successful Console Logi...
No data
Successes
Successful Console Logins
No Data to Display
There is no data for your filter. You can try modifying the query or updating the time range.

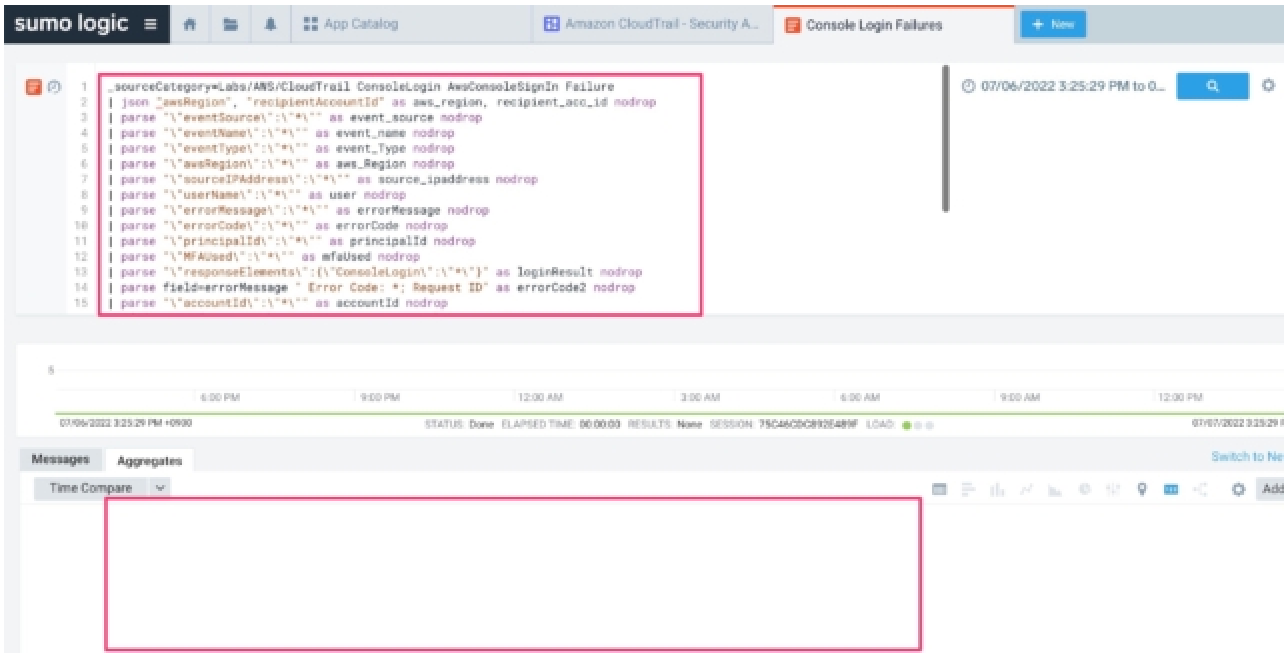
Since it says "No Data," you might think that the data itself failed to be imported, but other graphs are displayed, so it seems that the data was imported. If all other dashboards are not visualized, it may be that the data was not imported in the first place, or that data that is not the target of the app is specified. Please check this.
Also, behind the dashboard is a search query, and data may not be displayed because logs that do not match the search criteria just happen not to appear, or because no logs were generated during the search period.

Troubleshooting the cause of No Data

Let's take a closer look at the search query.
First, we'll look at Open in Log Search from the ellipsis in the Console Login Failures panel.



The search query to display this panel is written here. The output should be displayed at the bottom, but it is not.



```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| parse "\"responseElements\":{\"ConsoleLogin\":\"*\"}" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
```

```
07/06/2022 3:25:29 PM +0900    STATUS Done  ELAPSED TIME 00:00:00  RESULTS None  SESSION 75C46CDC892E489F  LOAD    07/07/2022 3:25:29
```

Let's look at the query statement.

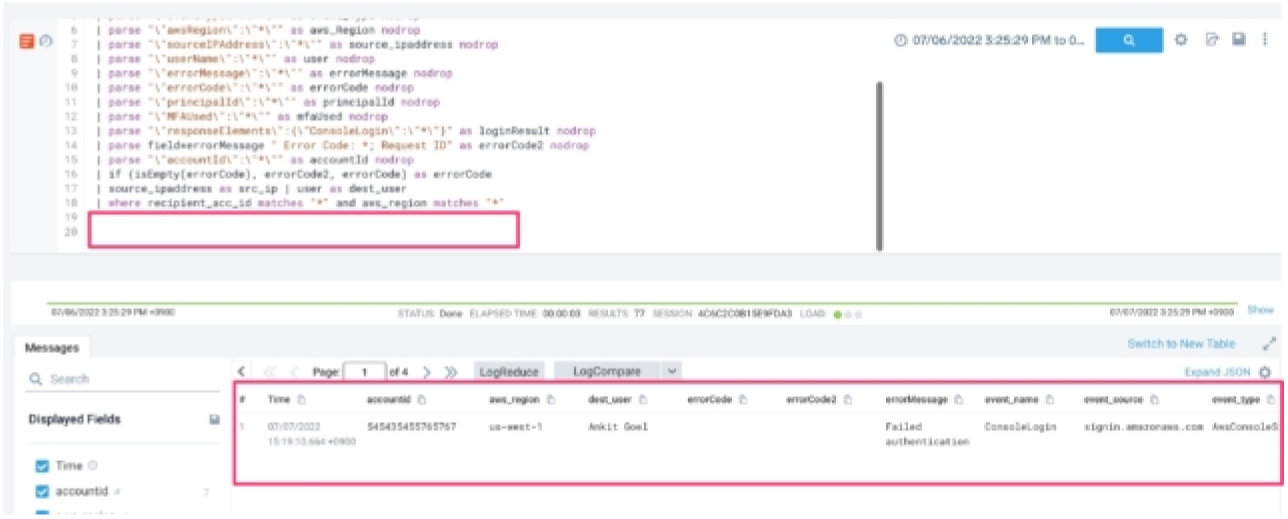
```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| parse "\"responseElements\":{\"ConsoleLogin\":\"*\"}" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
| if (isEmpty(errorCode), errorCode2, errorCode) as errorCode
| source_ipaddress as src_ip | user as dest_user
| where recipient_acc_id matches "*" and aws_region matches "*"
| where event_name="ConsoleLogin" and event_type="AwsConsoleSignIn" and
loginResult="Failure"
| fields src_ip, dest_user, mfaUsed, event_Type, event_name, errorCode,
errorMessage, principalId, aws_region, source_ipaddress, accountId
| timeslice 15m
| count as eventCount by _timeslice
| sort _timeslice
```

In Sumo Logic search queries, log filtering is primarily performed by " Parse (analyzing log structure) " operators (lines 2-15) or " Where (conditional search) " operators (lines 18-19). Where can be used to **"display results where the value of a specific field is XX"** and has some SQL-like aspects that make it easy to understand. However, it's important to note that filtering is also performed by Parse operators.
In this query, `json`and `parse`are operators for analyzing log structure. Parse operators have two roles **: 1. They structure logs using regular expressions or patterns, and use them as fields in later query statements for aggregation and processing.** 2. They exclude **logs that do not match regular expressions or patterns from search results** . However, if the **"nodrop"** option is enabled, function 2 can be skipped, and unmatched logs can also be displayed in the search results.
Therefore, we can see that the reason the search results were not displayed was due to function 2 of the Parse operators.

Next, let's look at the last Where in the previous query (line 19).

```
| where recipient_acc_id matches "*" and aws_region matches "*"
| where event_name="ConsoleLogin" and event_type="AwsConsoleSignIn" and
loginResult="Failure"
| fields src_ip, dest_user, mfaUsed, event_Type, event_name, errorCode,
errorMessage, principalId, aws_region, source_ipaddress, accountId
```

There may not be any logs that match this condition filter. Delete lines 19 and after and try searching.



```
6  | parse "\"awsRegion\":\"*\"" as aws_Region nodrop
7  | parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
8  | parse "\"userName\":\"*\"" as user nodrop
9  | parse "\"errorMessage\":\"*\"" as errorMessage nodrop
10 | parse "\"errorCode\":\"*\"" as errorCode nodrop
11 | parse "\"principalId\":\"*\"" as principalId nodrop
12 | parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
13 | parse "\"responseElements\":{\"ConsoleLogin\":\"*\"}" as loginResult nodrop
14 | parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
15 | parse "\"accountId\":\"*\"" as accountId nodrop
16 | if (isEmpty(errorCode), errorCode2, errorCode) as errorCode
17 | source_ipaddress as src_ip | user as dest_user
18 | where recipient_acc_id matches "*" and aws_region matches "*"
19
20
```

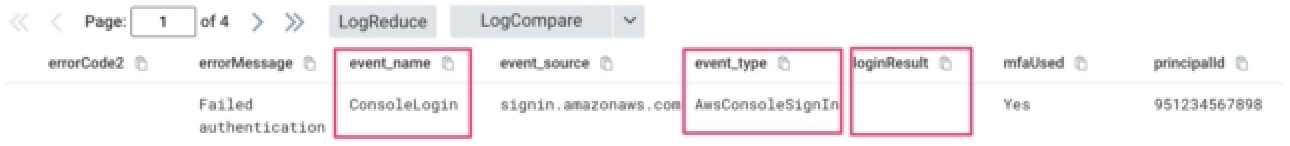
07/06/2022 3:25:29 PM +0900 STATUS Done ELAPSED TIME 00:00:03 RESULTS 77 SESSION 4C6C2C0B1SE9FDA3 07/07/2022 3:25:29 PM +0900

#	Time	accountid	aws_region	dest_user	errorCode	errorCode2	errorMessage	event_name	event_source	event_type
1	07/07/2022 15:19:10.664 +0900	545435455765767	us-west-1	Ankit Goel			Failed authentication	ConsoleLogin	signin.amazonaws.com	AwsConsoleS

I got some output, so this statement seems to be relevant.

```
where event_name="ConsoleLogin" and event_type="AwsConsoleSignIn" and
loginResult="Failure"
```

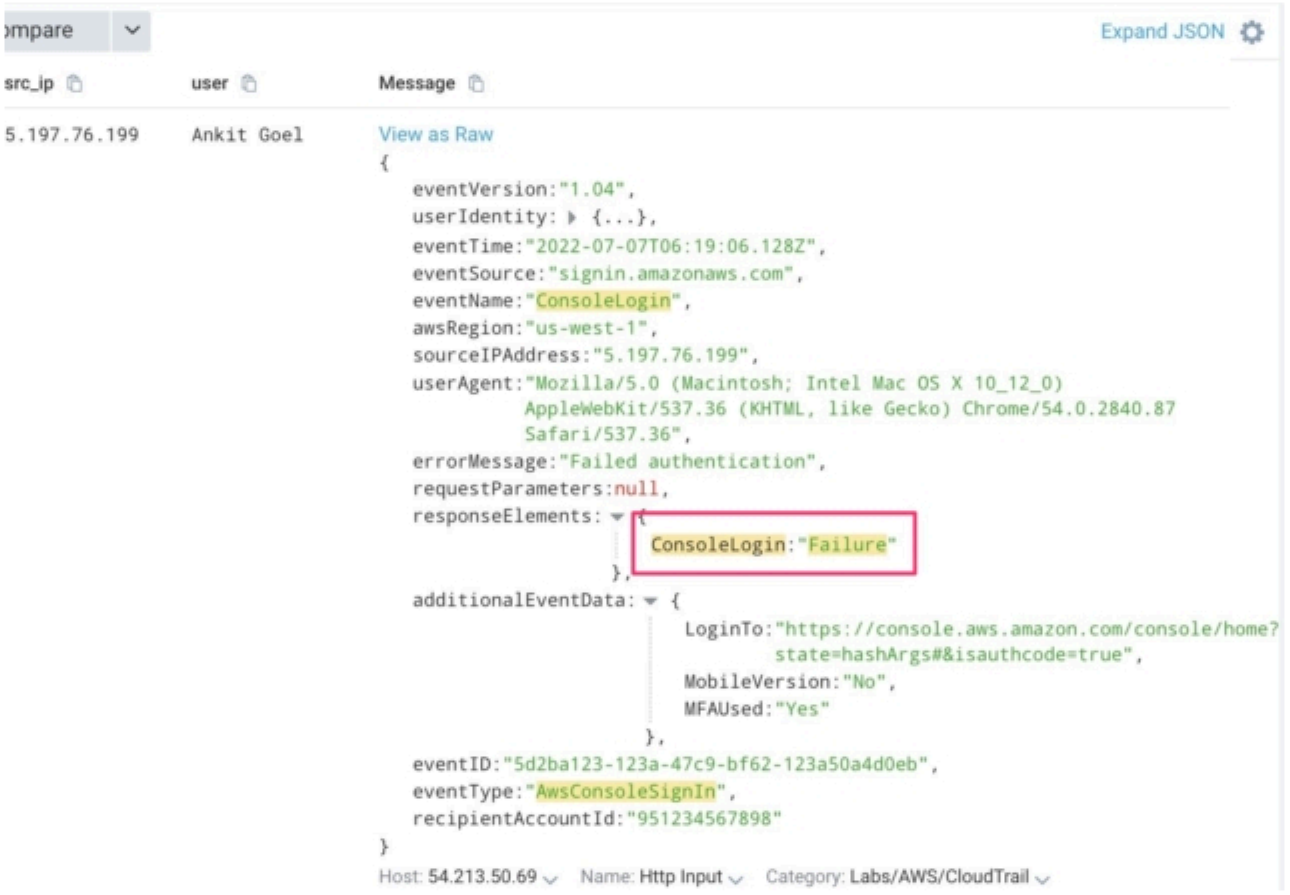
`where`Reading this, we can see that the search has been narrowed down to records where the value of the field "event_name" is "ConsoleLogin", "event_type" is "AwsConsoleSignIn", and "loginResult" is "Failure". If we check the search results after deleting lines 19 and onwards, we can see that the values of event_name and event_type are indeed displayed, but the value of loginResult is empty. Because the value of this "loginResult" field was not "Failure", it did not match the conditions of the where statement and the dashboard could not be displayed. So, were there really no records in the log where the value of the "loginResult" field was "Failure"?



Page: 1 of 4 LogReduce LogCompare

errorCode2	errorMessage	event_name	event_source	event_type	loginResult	mfaUsed	principalId
	Failed authentication	ConsoleLogin	signin.amazonaws.com	AwsConsoleSignIn		Yes	951234567898

Feeling a bit suspicious, I shifted the search results further to the right and found a section where I could see the original raw log message. When I checked there, I found the words "ConsoleLogin: "Failure"" in the responseElements key.



ompare Expand JSON

src_ip	user	Message
5.197.76.199	Ankit Goel	View as Raw

```
{
   eventVersion:"1.04",
   userIdentity: ▸ {...},
   eventTime:"2022-07-07T06:19:06.128Z",
   eventSource:"signin.amazonaws.com",
   eventName:"ConsoleLogin",
   awsRegion:"us-west-1",
   sourceIPAddress:"5.197.76.199",
   userAgent:"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_0)
             AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.87
             Safari/537.36",
   errorMessage:"Failed authentication",
   requestParameters:null,
   responseElements: ▾ {
                        ConsoleLogin:"Failure"
                      },
   additionalEventData: ▾ {
                            LoginTo:"https://console.aws.amazon.com/console/home?
                                     state=hashArgs#&isauthcode=true",
                            MobileVersion:"No",
                            MFAUsed:"Yes"
                          },
   eventID:"5d2ba123-123a-47c9-bf62-123a50a4d0eb",
   eventType:"AwsConsoleSignIn",
   recipientAccountId:"951234567898"
}
```

Host: 54.213.50.69 Name: Http Input Category: Labs/AWS/CloudTrail

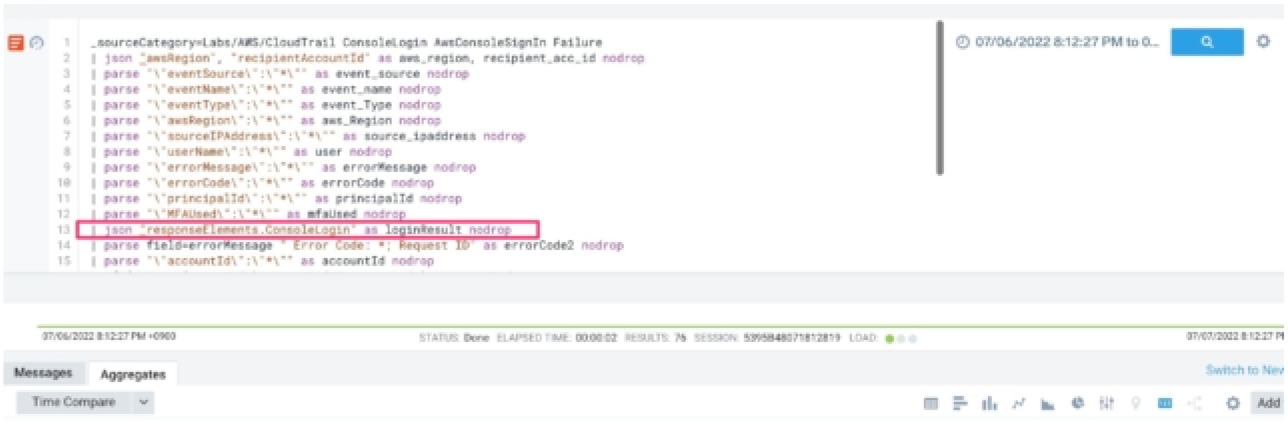
Let's go back to the query statement and take a look at the part of the query that is parsed into the "loginResult" field.

```
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| parse "\"responseElements\":{\"ConsoleLogin\":\"*\"}" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
```

We can see that the pattern "ConsoleLogin: XX" is parsed as a loginResult field within the responseElements key. The raw log shows "ConsoleLogin: "Failure"", so the value of the "loginResult" field should be "Failure", but since it is blank, the parsing itself may be incorrect. Let's consider how to
parse to correctly extract the "loginResult" field. There are several ways to do this, but we will use the (json operator)
. The json operator syntax allows parsing with **"json <key> as <field name>"** . Also, if json is nested, you can parse the inner part by using **<key name>.<key name>**
. Therefore, let's modify the parsing to the following and try the search again. (Line 13)

```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| json "responseElements.ConsoleLogin" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
| if (isEmpty(errorCode), errorCode2, errorCode) as errorCode
| source_ipaddress as src_ip | user as dest_user
| where recipient_acc_id matches "*" and aws_region matches "*"
| where event_name="ConsoleLogin" and event_type="AwsConsoleSignIn" and
loginResult="Failure"
| fields src_ip, dest_user, mfaUsed, event_Type, event_name, errorCode,
errorMessage, principalId, aws_region, source_ipaddress, accountId
| timeslice 15m
| count as eventCount by _timeslice
| sort _timeslice
```

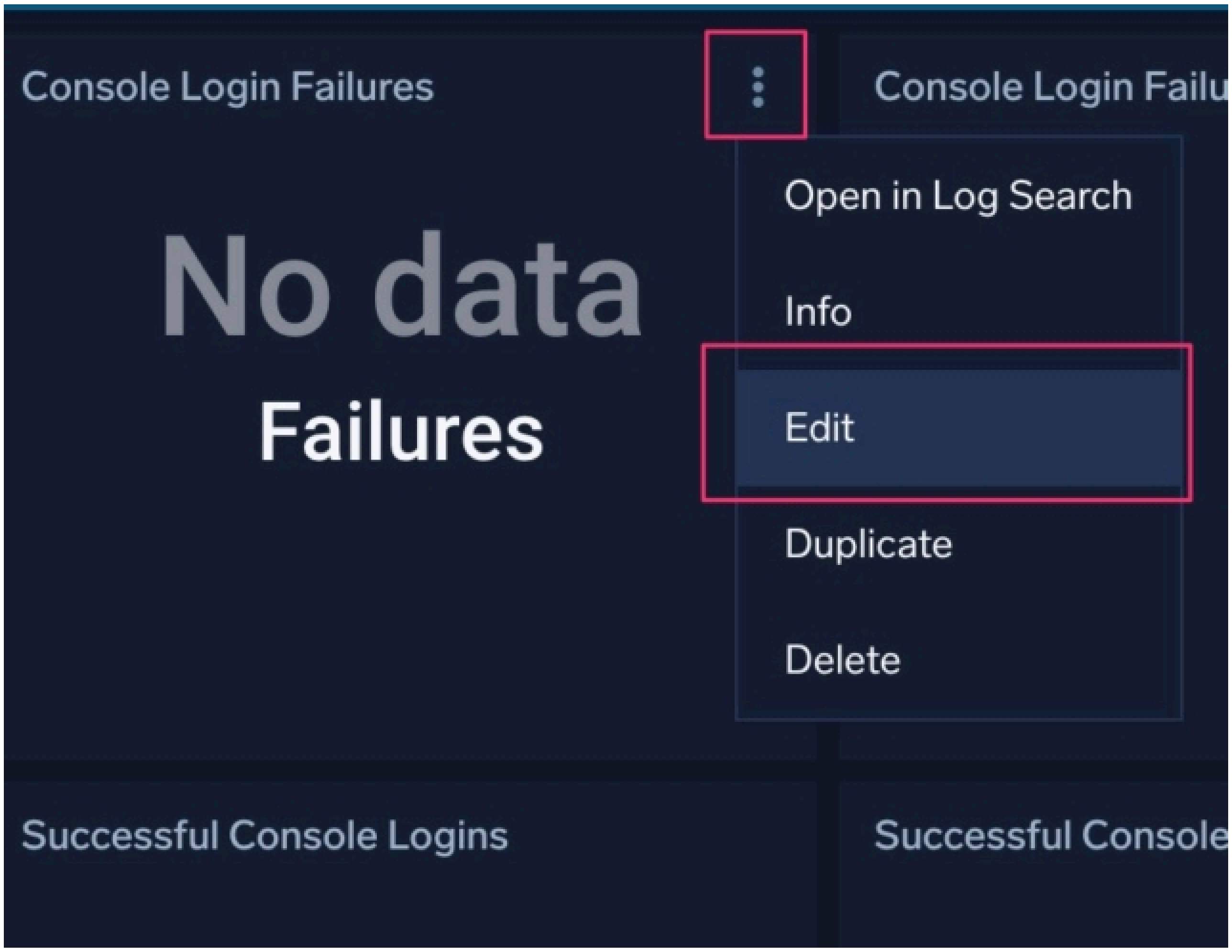
And it worked! It seems that the log was parsed properly and I was able to find a value where "loginResult" was "Failure." It wasn't that there were no logs that matched the conditions, but rather that I couldn't visualize them because the parsing was incorrect.



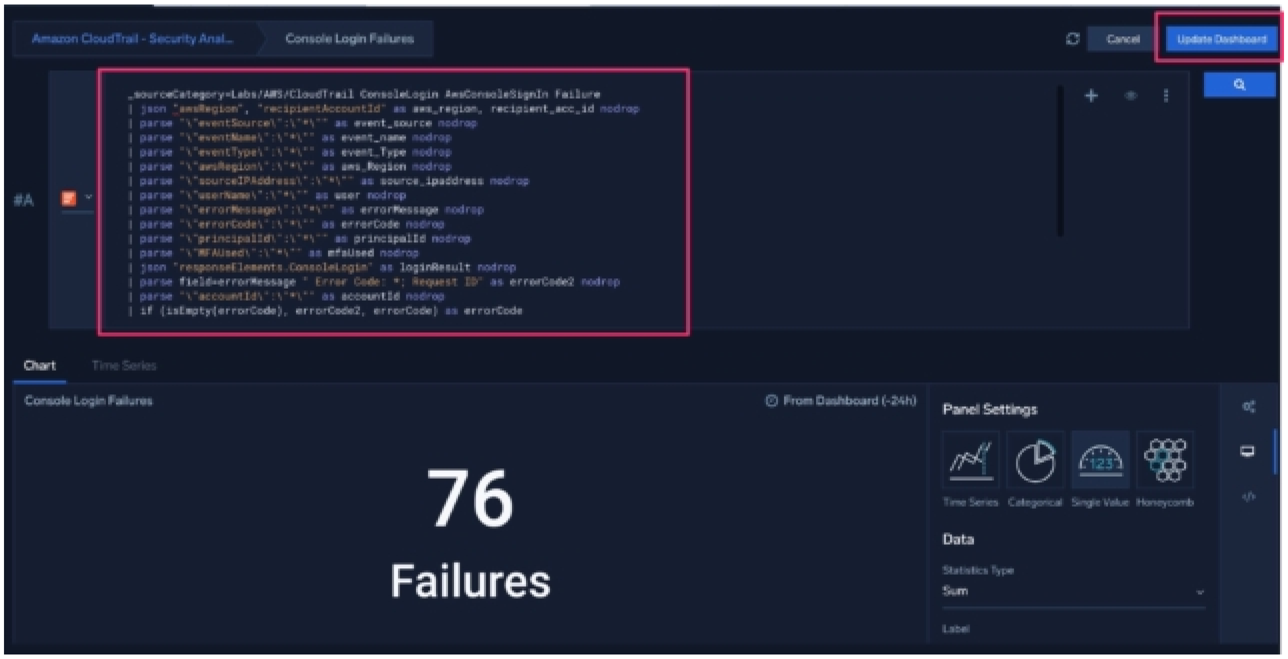
```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| json "responseElements.ConsoleLogin" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
```

07/06/2022 8:12:27 PM to 0...
07/06/2022 8:12:27 PM +0900
STATUS: Done ELAPSED TIME: 00:00:02 RESULTS: 76 SESSION: 5395848071812819 LOAD:
07/07/2022 8:12:27 PM
Messages
Aggregates
Switch to New
Time Compare
Add

By the way, if you want to update the dashboard, you need to modify and save the query statement, which can be opened by clicking Edit from the three dots on the dashboard panel.



Copy and paste the query above here and save it.



```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| json "responseElements.ConsoleLogin" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
| if (isEmpty(errorCode), errorCode2, errorCode) as errorCode
```

Console Login Failures

From Dashboard (-24h)

76

Failures

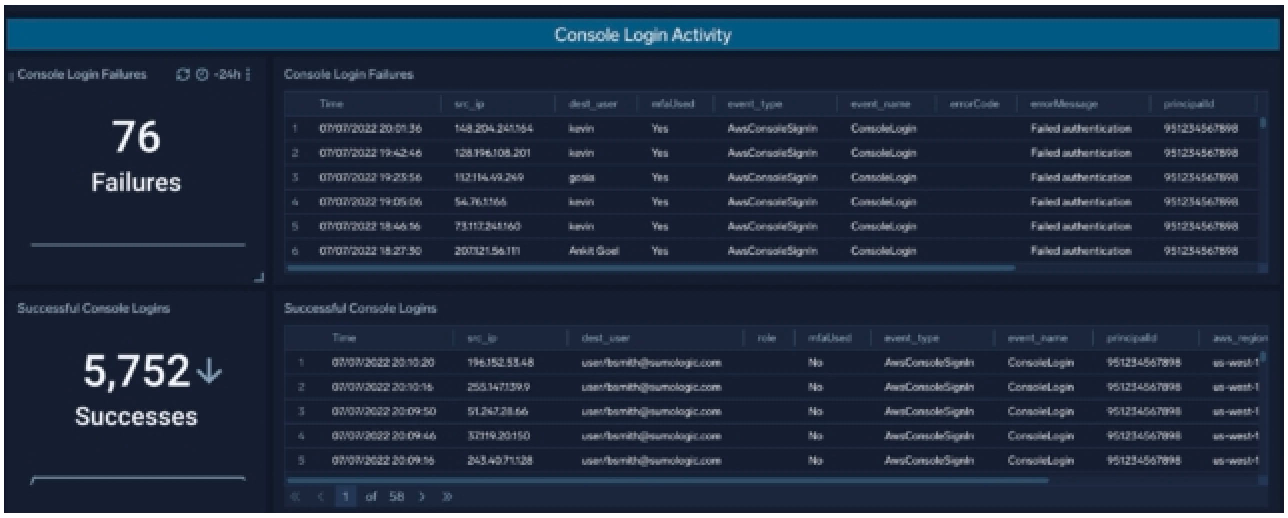
Panel Settings

Data

Statistics Type: Sum

Label

The remaining three panels have the same issue, so fix the same problem and refresh the dashboard. The data is now displayed correctly.


Console Login Activity

Console Login Failures -24h

76

Failures

Console Login Failures

	Time	src_ip	dest_user	mfaUsed	event_type	event_name	errorCode	errorMessage	principalId
1	07/07/2022 20:01:36	148.204.241.164	kevin	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
2	07/07/2022 19:42:46	128.196.108.201	kevin	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
3	07/07/2022 19:23:56	112.114.49.249	gosia	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
4	07/07/2022 19:05:06	54.76.1.166	kevin	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
5	07/07/2022 18:46:16	73.117.241.160	kevin	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
6	07/07/2022 18:27:30	207.121.56.111	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898

Successful Console Logins

5,752↓

Successes

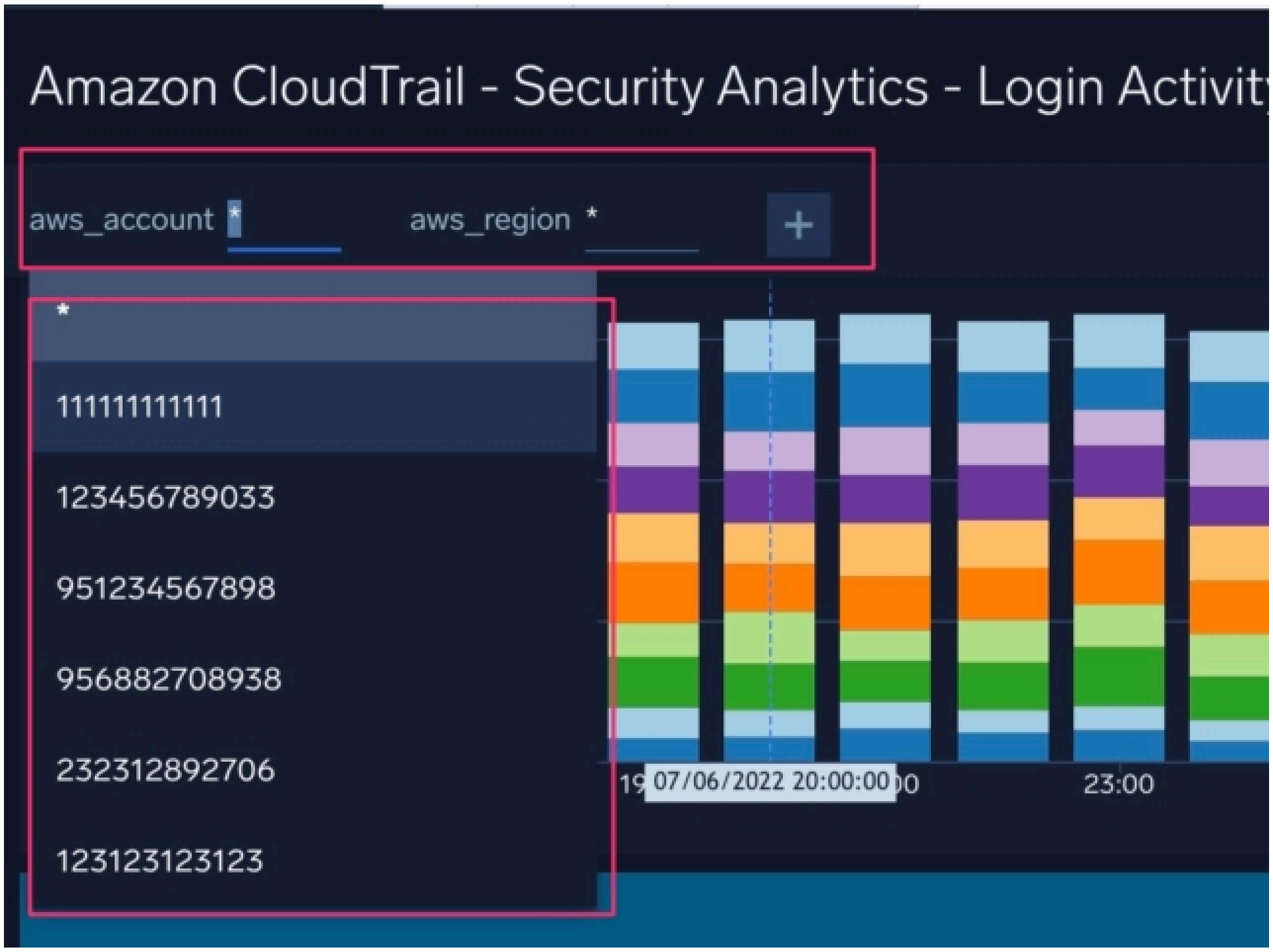
Successful Console Logins

	Time	src_ip	dest_user	role	mfaUsed	event_type	event_name	principalId	aws_region
1	07/07/2022 20:10:20	196.152.53.48	user/bsmith@sumologic.com		No	AwsConsoleSignIn	ConsoleLogin	951234567898	us-west-1
2	07/07/2022 20:10:16	255.147.139.9	user/bsmith@sumologic.com		No	AwsConsoleSignIn	ConsoleLogin	951234567898	us-west-1
3	07/07/2022 20:09:50	51.247.28.66	user/bsmith@sumologic.com		No	AwsConsoleSignIn	ConsoleLogin	951234567898	us-west-1
4	07/07/2022 20:09:46	37.119.20.150	user/bsmith@sumologic.com		No	AwsConsoleSignIn	ConsoleLogin	951234567898	us-west-1
5	07/07/2022 20:09:16	243.40.71.128	user/bsmith@sumologic.com		No	AwsConsoleSignIn	ConsoleLogin	951234567898	us-west-1

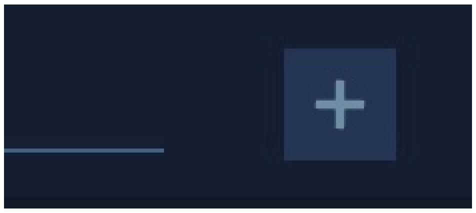
1 of 58

Get started with dashboard filters

Dashboard filters are available at the top of the dashboard, and account filtering is already available on this dashboard, so you can use these to filter to the accounts you want to see.



Unfortunately, there are no filters for individual users. So, let's create one ourselves.
To create a filter, click the + icon on the right side of the filter.



This will open an editor for creating the filter.

We will explain the items to be set.

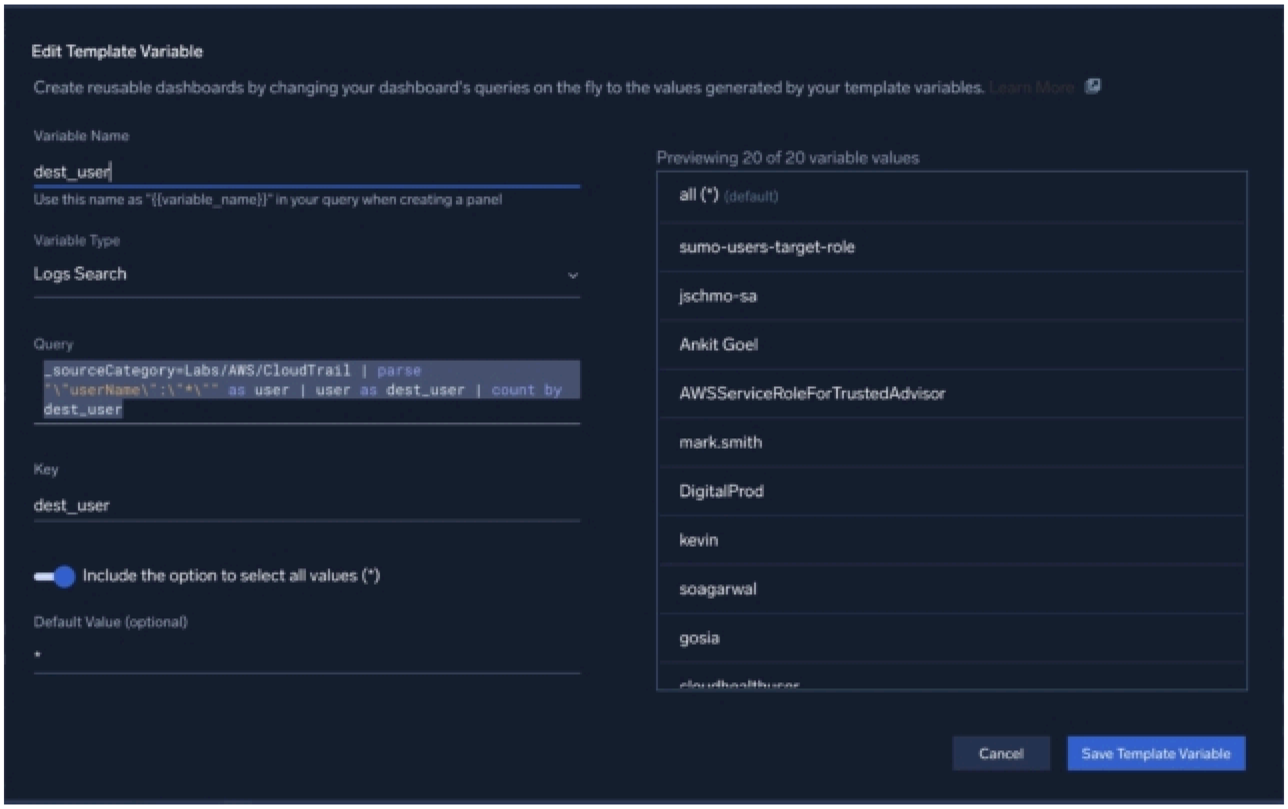
- `Variable Name`: Define a variable to link the filter with the filter conditions in the dashboard. You can filter by including the variable name set here as a condition in the dashboard query statement. The notation format is **{{variable name}} .**
- `Variable Type`: Specify the type of query in which the variable to be used for the filter will be used. Depending on the type you select, other settings will be slightly different.

- `Query`: This can be set when you select Logs Search as the Variable Type. You can write a query statement that can be selected as an auto-complete candidate when using a filter.
- `Key`: This can be set when you select Logs Search as the Variable Type. You can specify which key values output by the query will be used as filter candidates.
- `Include the option to select all values (\*)`: Determines whether **"*" (all)** can be selected as a filter condition .
- `Default Values (optional)`: Sets the default value filter condition.

Now let's fill in the values.

- `Variable Name`:`{{dest_user}}`
- `Variable Type`:`Logs Search`
- `Query`:`_sourceCategory=Labs/AWS/CloudTrail | parse "\"userName\":\"*\"" as user | user as dest_user | count by dest_user`
- `Key`:`dest_user`
- `Include the option to select all values (\*)`:`ON`
- `Default Values (optional)`:`\*`

Create a filter with **Save Template Variable .**



Edit Template Variable

Create reusable dashboards by changing your dashboard's queries on the fly to the values generated by your template variables. Learn More

Variable Name: dest_user

Use this name as "{{variable_name}}" in your query when creating a panel

Variable Type: Logs Search

Query:
```
_sourceCategory=Labs/AWS/CloudTrail | parse "\"userName\":\"*\"" as user | user as dest_user | count by dest_user
```

Key: dest_user

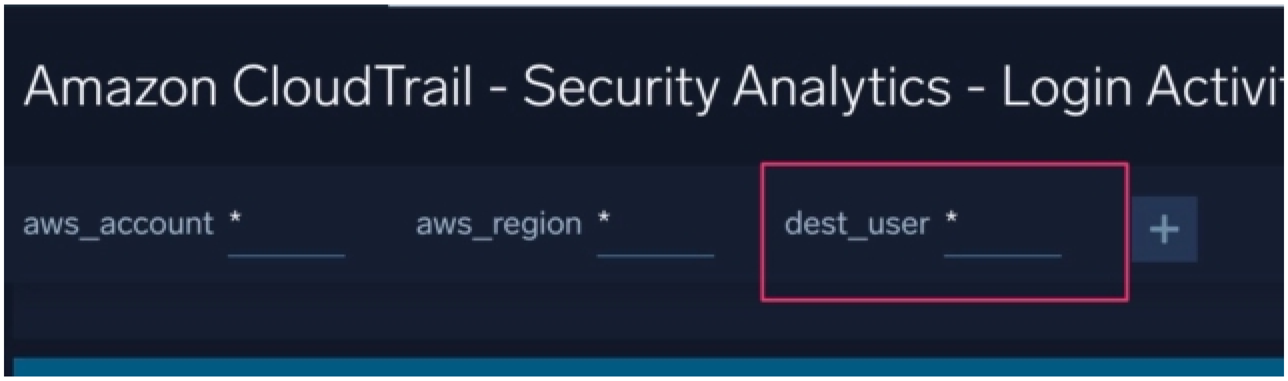
Include the option to select all values (*)

Default Value (optional): *

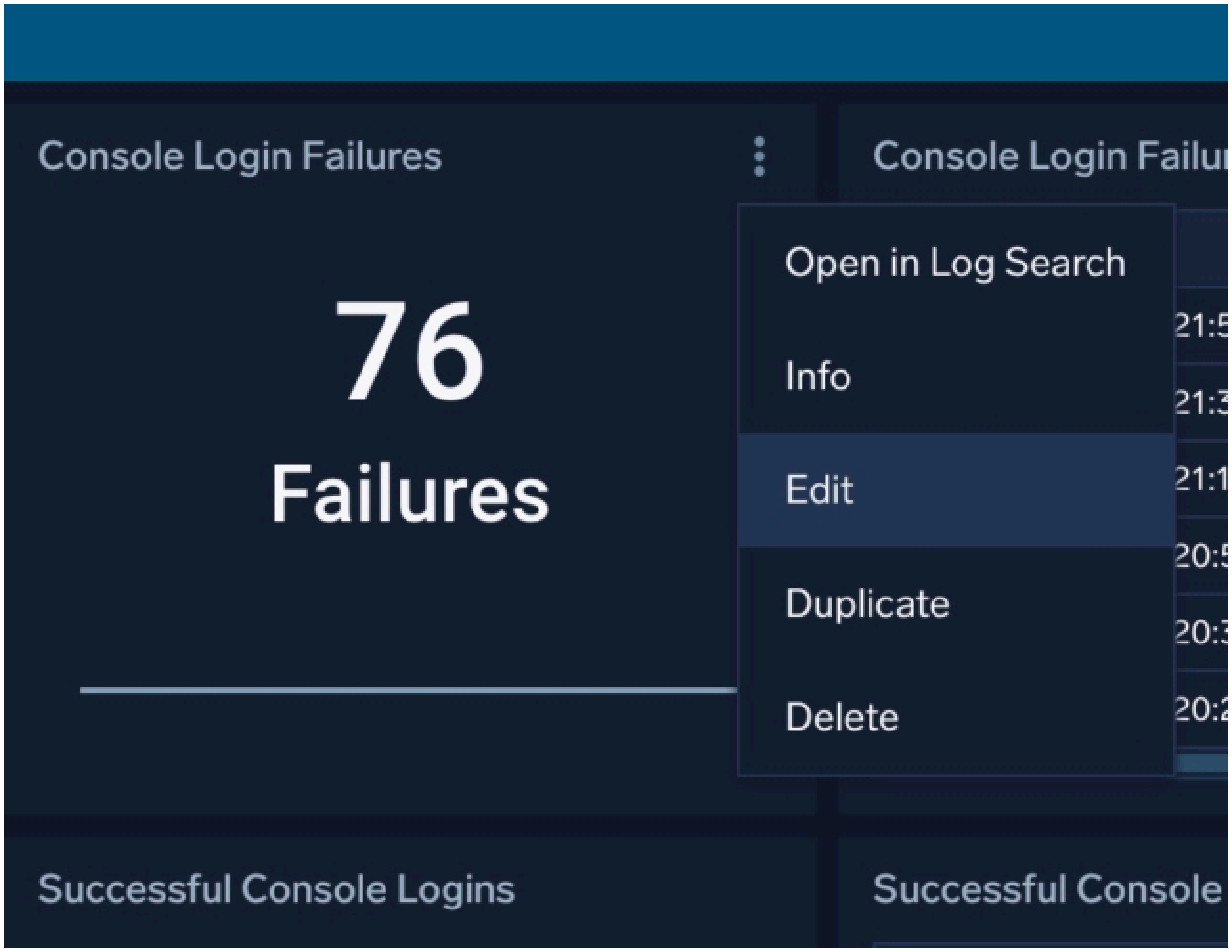
Previewing 20 of 20 variable values

all (*) (default)
sumo-users-target-role
jschmo-sa
Ankit Goel
AWSServiceRoleForTrustedAdvisor
mark.smith
DigitalProd
kevin
soagarwal
gosia
cloudhealthuser

Then the new filter was reflected on the screen.



Finally, embed the variable (Variable Name) that will be the filter condition in the dashboard query statement. Select Edit on the panel.

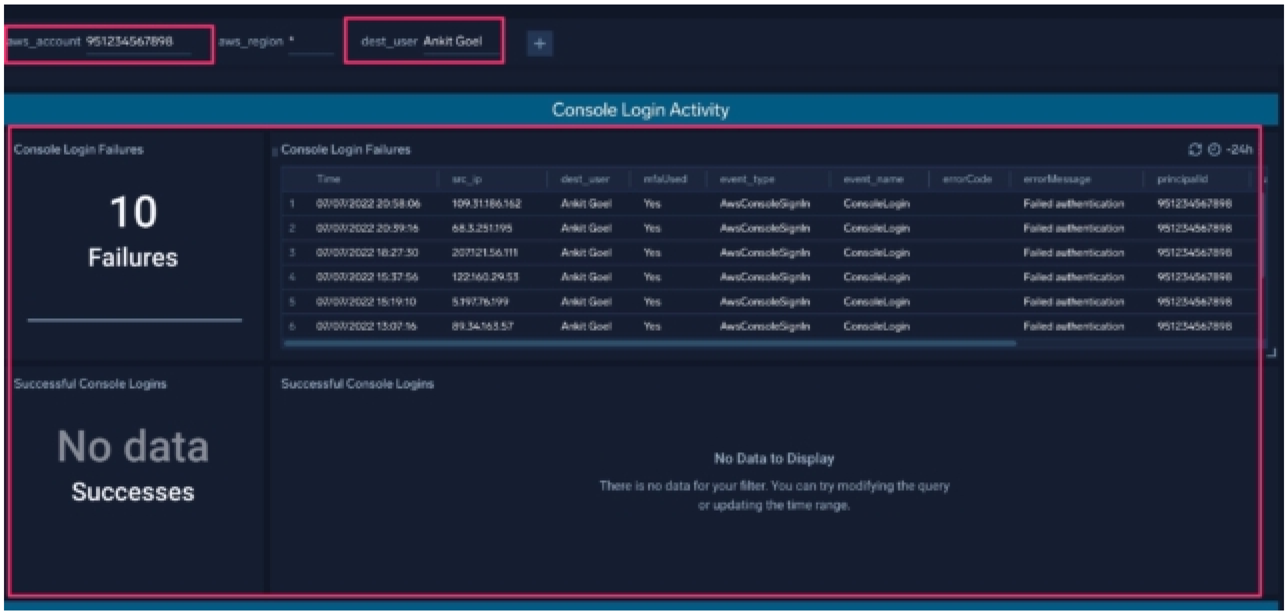


The where statement on line 18 will be the filter condition. At the end of this, we will add a new variable condition to filter by user, using AND. (`and dest_user matches "{{dest_user}}"`)

```
_sourceCategory=Labs/AWS/CloudTrail ConsoleLogin AwsConsoleSignIn Failure
| json "awsRegion", "recipientAccountId" as aws_region, recipient_acc_id nodrop
| parse "\"eventSource\":\"*\"" as event_source nodrop
| parse "\"eventName\":\"*\"" as event_name nodrop
| parse "\"eventType\":\"*\"" as event_Type nodrop
| parse "\"awsRegion\":\"*\"" as aws_Region nodrop
| parse "\"sourceIPAddress\":\"*\"" as source_ipaddress nodrop
| parse "\"userName\":\"*\"" as user nodrop
| parse "\"errorMessage\":\"*\"" as errorMessage nodrop
| parse "\"errorCode\":\"*\"" as errorCode nodrop
| parse "\"principalId\":\"*\"" as principalId nodrop
| parse "\"MFAUsed\":\"*\"" as mfaUsed nodrop
| json "responseElements.ConsoleLogin" as loginResult nodrop
| parse field=errorMessage " Error Code: *; Request ID" as errorCode2 nodrop
| parse "\"accountId\":\"*\"" as accountId nodrop
| if (isEmpty(errorCode), errorCode2, errorCode) as errorCode
| source_ipaddress as src_ip | user as dest_user
| where recipient_acc_id matches "{{aws_account}}" and aws_region matches "
{{aws_region}}" and dest_user matches "{{dest_user}}"
| where event_name="ConsoleLogin" and event_type="AwsConsoleSignIn" and
loginResult="Failure"
| fields src_ip, dest_user, mfaUsed, event_Type, event_name, errorCode,
errorMessage, principalId, aws_region, source_ipaddress, accountId
| timeslice 15m
| count as eventCount by _timeslice
| sort _timeslice
```

Apply this to other panels in the same way to update your dashboard.

Finally, you can now use filters to filter your dashboard by account or user.



	Time	src_ip	dest_user	mfaUsed	event_type	event_name	errorCode	errorMessage	principalId
1	07/07/2022 20:58:06	109.31.186.162	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
2	07/07/2022 20:39:16	68.3.251.195	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
3	07/07/2022 18:27:30	207.121.56.111	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
4	07/07/2022 15:37:56	122.160.29.53	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
5	07/07/2022 15:19:10	5.197.76.199	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898
6	07/07/2022 13:07:16	89.34.163.57	Ankit Goel	Yes	AwsConsoleSignIn	ConsoleLogin		Failed authentication	951234567898

summary

In this way, we were able to modify the dashboard to the appropriate query and add additional filtering functionality. The filters in the New Dashboard are very easy to use and highly functional, so we expect they will be useful in a variety of situations.
We hope you will find this useful and help strengthen your security through log analysis.