M-21-31 Dashboard Documentation

The M-21-31 Dashboards are to show users where they are receiving live logs as they align with the M-21-31 requirements. There are two ways for this to map logs to M-21-31 requirements. Ocsf Mapping and custom query. This document will detail how to work with each.

OCSF Mapping:

There is a pure OCSF schema mapping. As OCSF is the standard at SentinelOne for all Marketplace apps, we have mapped all the OCSF types to a M-21-31 requirement, where it makes sense.

How this works:

All OCSF type_ids are listed in a file under /datatables/ocsf.csv. And all M-21-31 requirements (EL-0 only at the time of this document) are under /datatables/m2131.csv

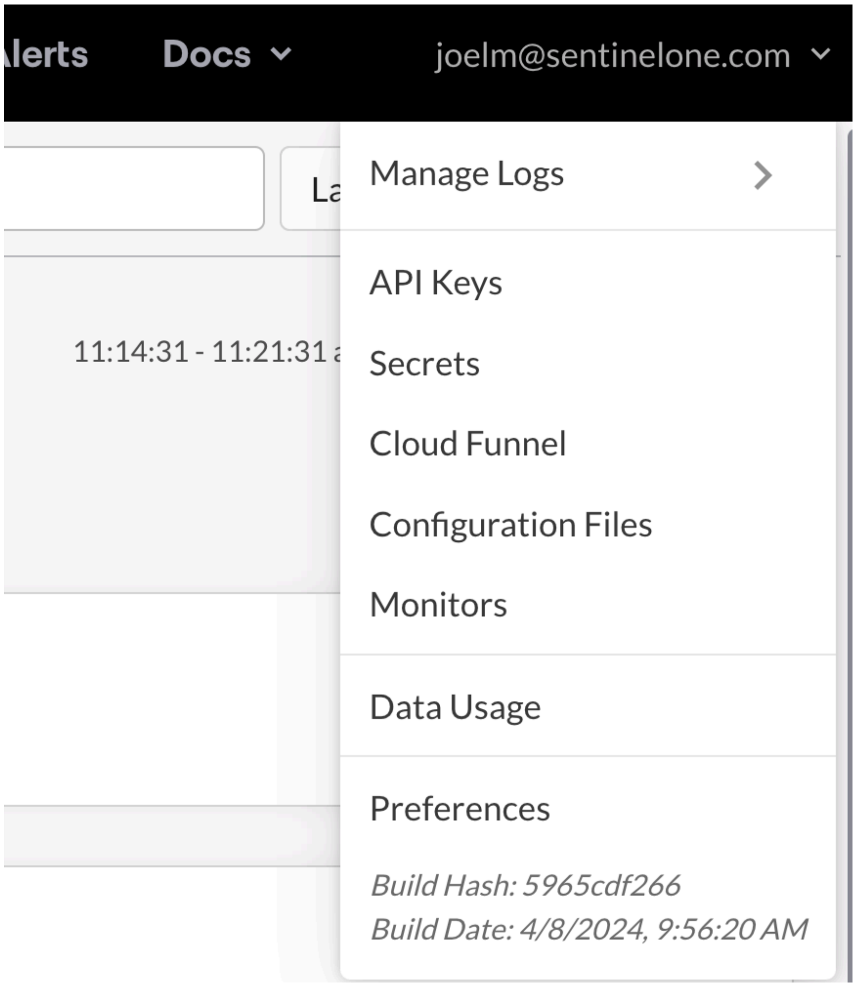
In the ocsf.csv, each type_id is listed and a m2131_fk maps to the primary key of the requirement in m2131.csv.

It is that simple.

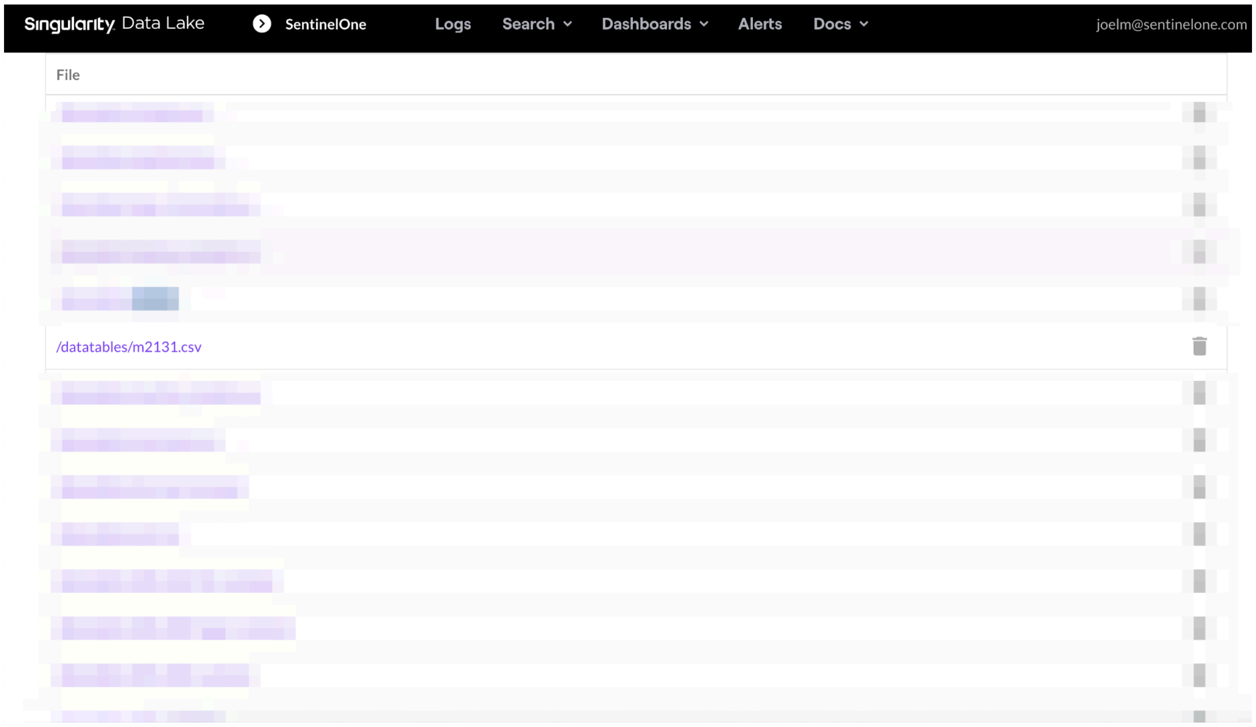
Adding a requirement

You can add a new requirement by doing the following steps

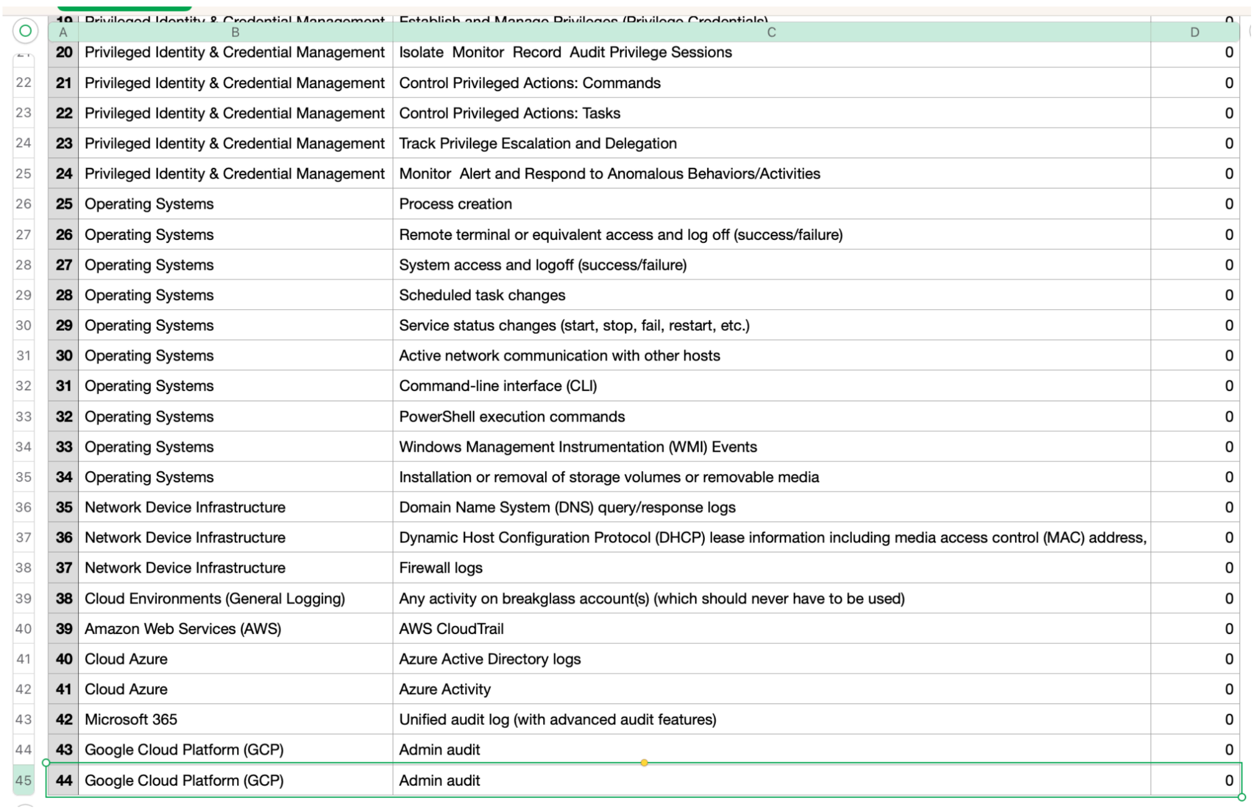
- Click your name in the top right corner.
- **Configuration Files**



-
- Cick **m2131.csv**



-
- It is easier to edit in an csv editor like excel, but you can edit the text from the app too.
- Add a new row with the following values
    - Id - sequential primary key you will use to id the row
    - Category - Category of the requirement. i.e IAM, Cloud, OS
    - Log - The name of the requirement. I.e “Track user logins”
    - Criticality - EL0 - EL4 as an integer.

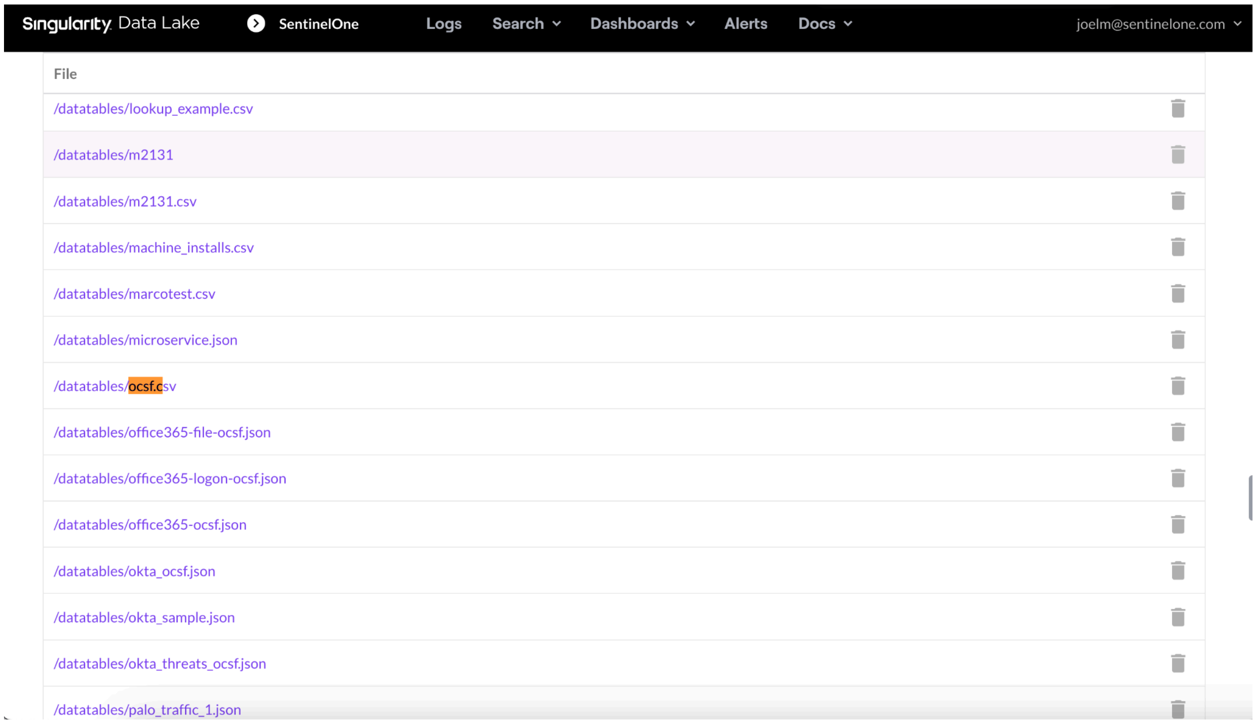


A	B	C	D
19	Privileged Identity & Credential Management	Establish and Manage Privileges (Privilege Credentials)	0
20	Privileged Identity & Credential Management	Isolate Monitor Record Audit Privilege Sessions	0
21	Privileged Identity & Credential Management	Control Privileged Actions: Commands	0
22	Privileged Identity & Credential Management	Control Privileged Actions: Tasks	0
23	Privileged Identity & Credential Management	Track Privilege Escalation and Delegation	0
24	Privileged Identity & Credential Management	Monitor Alert and Respond to Anomalous Behaviors/Activities	0
25	Operating Systems	Process creation	0
26	Operating Systems	Remote terminal or equivalent access and log off (success/failure)	0
27	Operating Systems	System access and logoff (success/failure)	0
28	Operating Systems	Scheduled task changes	0
29	Operating Systems	Service status changes (start, stop, fail, restart, etc.)	0
30	Operating Systems	Active network communication with other hosts	0
31	Operating Systems	Command-line interface (CLI)	0
32	Operating Systems	PowerShell execution commands	0
33	Operating Systems	Windows Management Instrumentation (WMI) Events	0
34	Operating Systems	Installation or removal of storage volumes or removable media	0
35	Network Device Infrastructure	Domain Name System (DNS) query/response logs	0
36	Network Device Infrastructure	Dynamic Host Configuration Protocol (DHCP) lease information including media access control (MAC) address,	0
37	Network Device Infrastructure	Firewall logs	0
38	Cloud Environments (General Logging)	Any activity on breakglass account(s) (which should never have to be used)	0
39	Amazon Web Services (AWS)	AWS CloudTrail	0
40	Cloud Azure	Azure Active Directory logs	0
41	Cloud Azure	Azure Activity	0
42	Microsoft 365	Unified audit log (with advanced audit features)	0
43	Google Cloud Platform (GCP)	Admin audit	0
44	Google Cloud Platform (GCP)	Admin audit	0

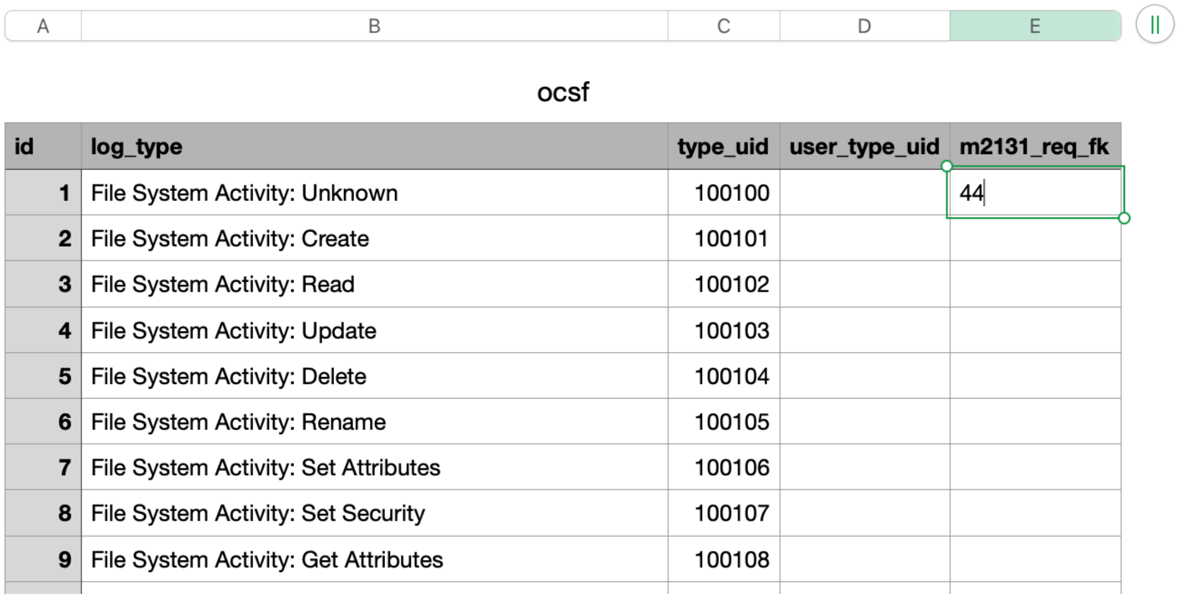
Map to an OCSF Req

The requirement you just created above can be mapped to an OCSF task_id.

1. Click your name in the top right > Config Files > Open ocsf.csv



File
/datatables/lookup_example.csv
/datatables/m2131
/datatables/m2131.csv
/datatables/machine_installs.csv
/datatables/marcotest.csv
/datatables/microservice.json
/datatables/ocsf.csv
/datatables/office365-file-ocsf.json
/datatables/office365-logon-ocsf.json
/datatables/office365-ocsf.json
/datatables/okta_ocsf.json
/datatables/okta_sample.json
/datatables/okta_threats_ocsf.json
/datatables/palo_traffic_1.json

2. It is easier to edit in an csv editor like excel, but you can edit the text from the app too.
3. Place the Id of the requirement you created above the m2131_req_fk column
   a. 

ocsf

id	log_type	type_uid	user_type_uid	m2131_req_fk
1	File System Activity: Unknown	100100		44
2	File System Activity: Create	100101		
3	File System Activity: Read	100102		
4	File System Activity: Update	100103		
5	File System Activity: Delete	100104		
6	File System Activity: Rename	100105		
7	File System Activity: Set Attributes	100106		
8	File System Activity: Set Security	100107		
9	File System Activity: Get Attributes	100108		

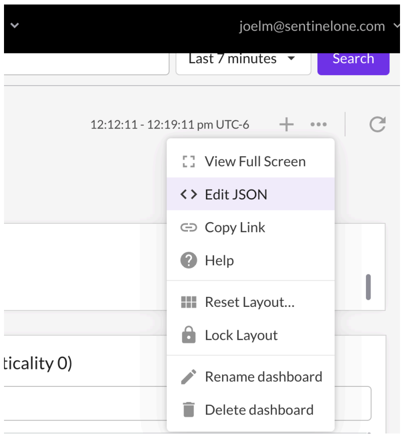
4. Save or export as csv and replace.

Custom Queries

How this works

Obviously, OCSF doesn't encompass everything in the M-21-31 requirements. There are also more complex ocsf commands you can do with custom filters, and for these edge cases or custom cases, this is where Customer Queries come into play. You can effectively map any requirement to a custom query.

- Custom queries are added on the dashboard json



- Custom Query. i.e powershell usage. The query is. `indicator.name = “powershell”)`
- Here is what a params looks like

-
  ```
  {
      parameters: [
      //Custom Queries
      //powershell
      { "name": "powershell",  "options": {"display": "hidden"}, "defaultValue":
  "indicator.name = 'PowershellExecution' OR src.process.name='powershell.exe'" }
  ,
      { "name": "powershell-requirement",   "options": {"display": "hidden"},
  "defaultValue": "32" }
  ]
  ```
- There are two parts:
  - Query: This param goes on the dashboard json so it can be referenced. This contains the custom query as the defaultValue
    - `powershell`
    - `{ "name": "powershell",  "options": {"display": "hidden"}, "defaultValue": "indicator.name = 'PowershellExecution' OR src.process.name='powershell.exe'" }`
  - Requirement: The requirement id from “Adding a new requirement” would go in the defaultValue

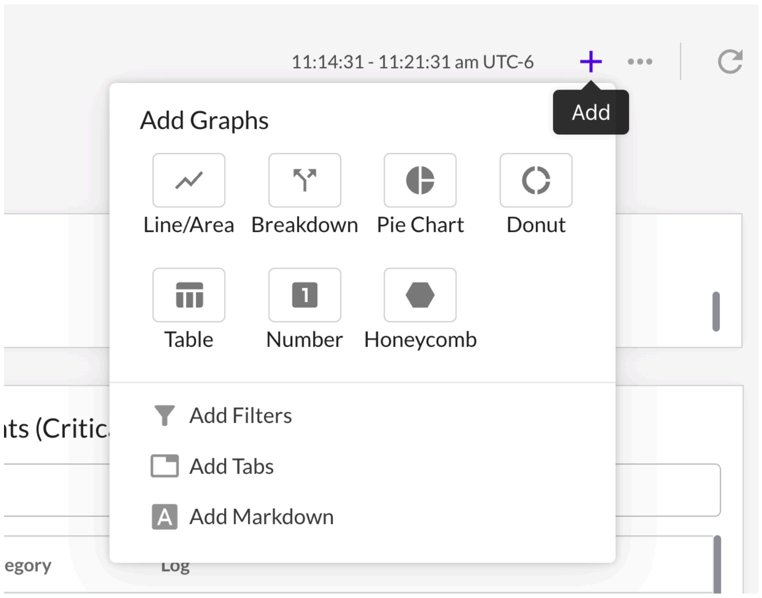
- `powershell-requrement`
- `{ "name": "powershell-requirement",   "options": {"display": "hidden"},  "defaultValue": "32" }`

Displaying

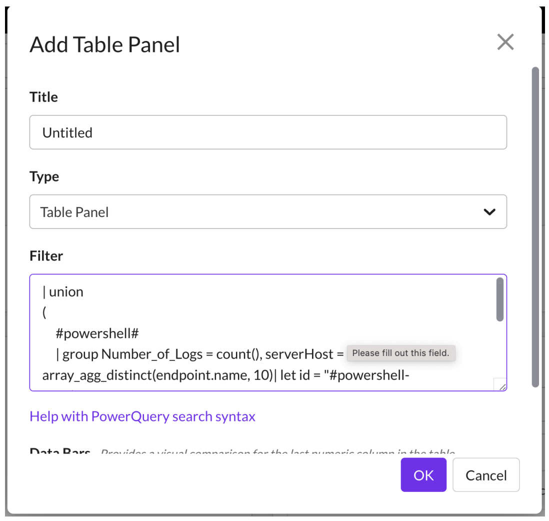
You can do whatever you want with these params on the dashboard. If you would like to align with the other custom queries, you can do either a table or a honeycomb chart.

Table

1. Add Graph
2. Add Table



3. Add **Table** panel

a. 

Add Table Panel

Title

Untitled

Type

Table Panel

Filter

```
| union
(
   #powershell#
   | group Number_of_Logs = count(), serverHost = 
array_agg_distinct(endpoint.name, 10)| let id = "#powershell-
```

Please fill out this field.

Help with PowerQuery search syntax

4. Paste the table union (limit 10 queries per) but you can have as many tables as you want.
   Here is an example for the Table Panel Filter:

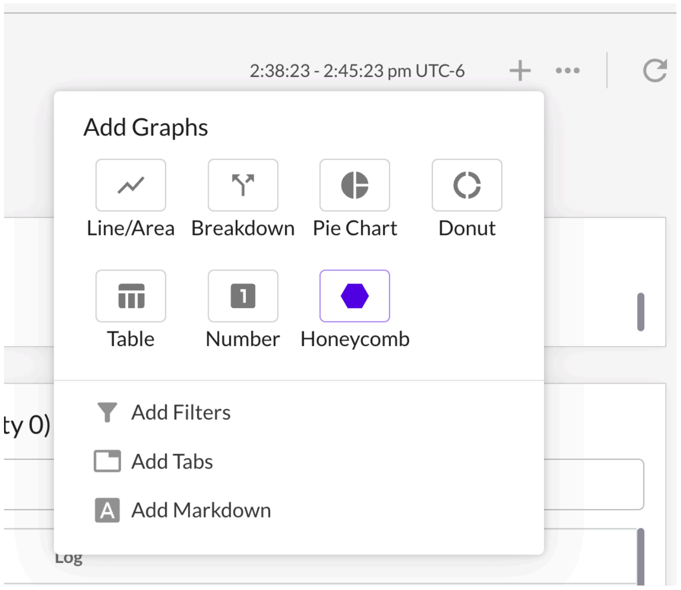
```
| union
(
     #powershell#
     | group Number_of_Logs = count(), serverHost = array_agg_distinct(endpoint.name,
10)| let id = "#powershell-requirement#"
     | #req_lookup#
),
(
     #cmd#
     | group Number_of_Logs = count(), serverHost = array_agg_distinct(endpoint.name,
10) | let id = "#cmd-requirement#"
     | #req_lookup#
)

| columns Number_of_Logs, Category, Log, type_uid = "null", "Host(s)" = serverHost
```

5. You are referencing the query and requirement with the #var# syntax. I.e #powershell#
6. You are just adding another union to the list. I.e

```
(
     #powershell#
     | group Number_of_Logs = count(), serverHost = array_agg_distinct(endpoint.name,
10)| let id = "#powershell-requirement#"
     | #req_lookup#
),
```

Honeycomb

1. You can also add this as a honeycomb chart. (10 allowed per custom query)



2.
3. The command is the same as before, the only difference is the final columns command.

```
| union
(
     #powershell#
     | group Number_of_Logs = count(), serverHost = array_agg_distinct(endpoint.name,
10)| let id = "#powershell-requirement#"
     | #req_lookup#
),
(
     #cmd#
     | group Number_of_Logs = count(), serverHost = array_agg_distinct(endpoint.name,
10) | let id = "#cmd-requirement#"
     | #req_lookup#
)

| columns  Category,  Log,  Number_of_Logs
```