Honeypot-Based Threat Detection: Real-World Attack Analysis with Azure Sentinel:

Prepared by: Yasif Farook
Date of Creation: 10/03/2025

Table of Contents

1.0 Project Overview

This project involves setting up a honeypot *virtual machine* (VM) to detect and analyse real Remote Desktop Protocol (RDP) attack attempts, including brute-force attacks, using Azure Sentinel. Unlike simulated attack scenarios, this honeypot was actively targeted by real-world attackers, providing genuine security event data for analysis. The project demonstrates the application of SOC (Security Operations Center) methodologies in monitoring, detecting, and responding to unauthorised login attempts.

Key Activities:

- Deployment of a Windows Server VM as a honeypot to attract real attack attempts.
- Real-time monitoring of RDP brute-force attempts using Windows Event Viewer.
- Integration with Azure Log Analytics Workspace and Sentinel to aggregate, store, and analyse security event data.
- Threat investigation using Kusto Query Language (KQL) for querying security logs.
- Geolocation enrichment to map attack origins and identify high-risk regions.
- Visualisation of attack trends using Sentinel's Map feature for real-time monitoring.

1.1 Tools and Technologies

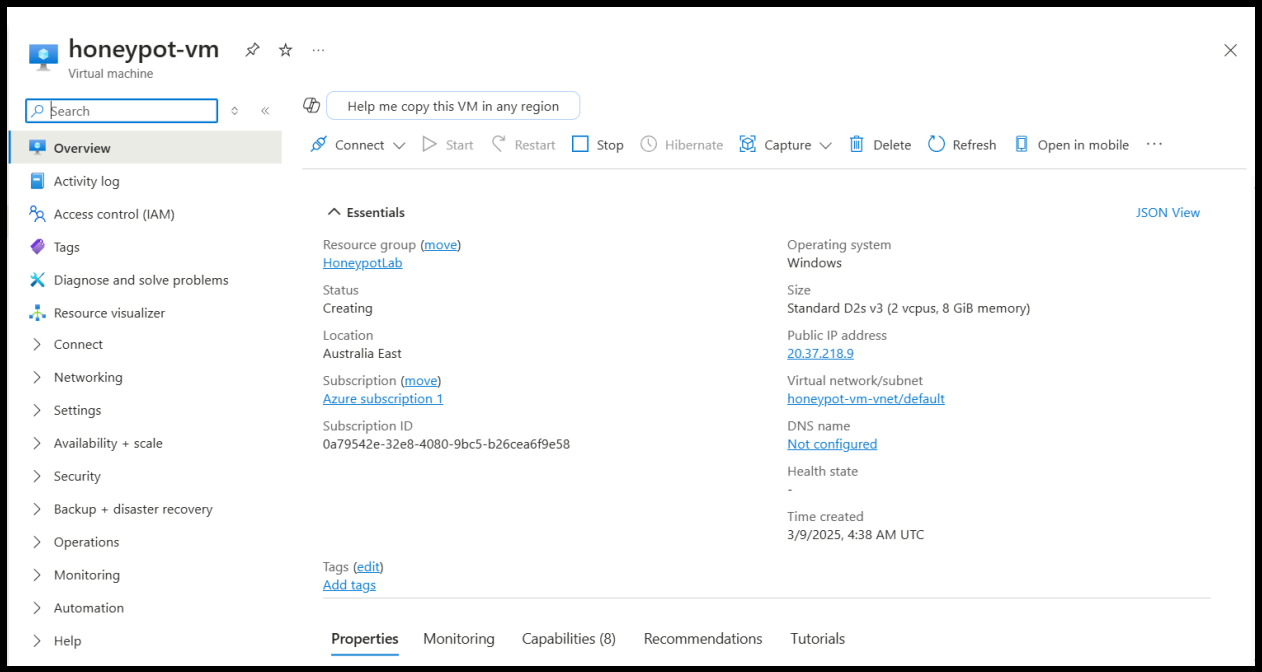
Tool	Purpose
Microsoft Azure	Hosts the honeypot VM and manages cloud-based security infrastructure.
Windows Server (Honeypot VM)	Acts as a decoy to attract and log unauthorised login attempts.
Event Viewer	Captures security event logs, including failed

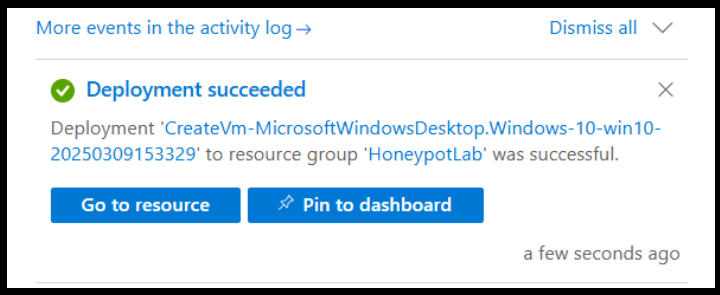
	login attempts.
Azure Log Analytics	Collects, stores, and enables querying of event logs.
Azure Sentinel (SIEM)	Provides real-time threat detection, alerting, and visualization of attacks.
Kusto Query Language (KQL)	Enables detailed log analysis and correlation of security data.
IP Geolocation API	Enriches logs with location-based intelligence.

2.0 Setting Up Relevant Machines (VM, Log Analytics Workspace)

2.1 Setting Up the Windows Server Virtual Machine

The VM is set up within the Azure environment. The purpose of this machine is to act as a decoy system, deliberately configured to attract RDP brute force attempts.

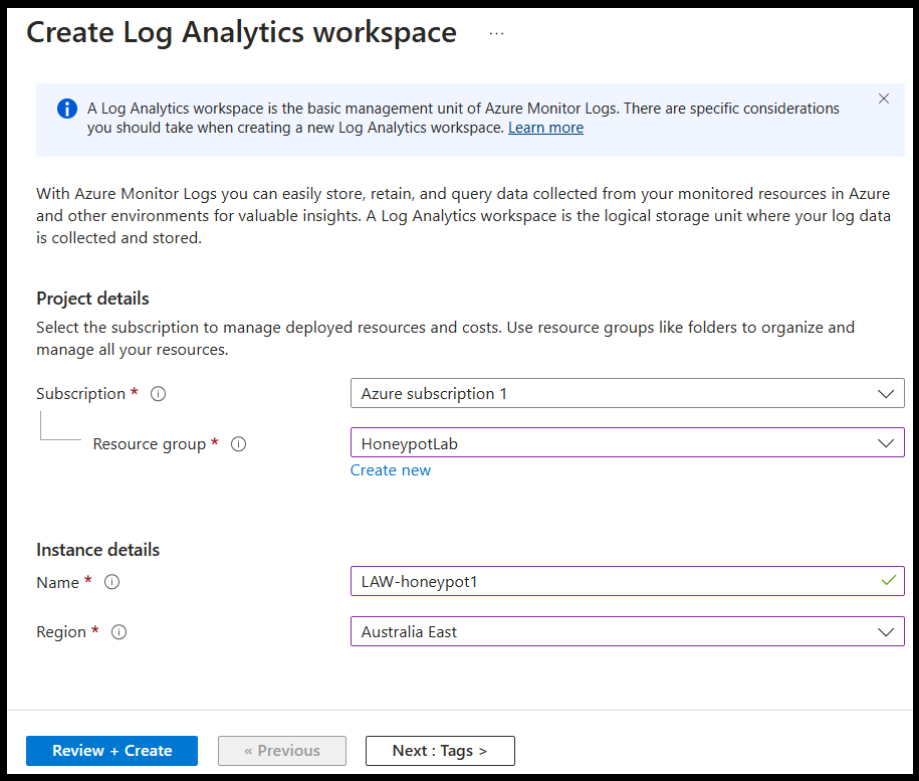




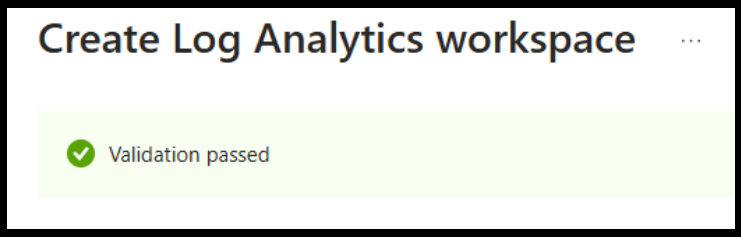
The Windows Server VM serves as a honeypot (hence the name *"honeypot-vm"*), to attract login attempts, which will then be logged and monitored under the **Log Analytics Workspace**, enabling real-time detection and analysis of activity. This could include failed login attempts or potential brute-force attacks. Unlike typical systems, the honeypot will be left intentionally vulnerable to lure real-world attackers into initiating login attempts, which would then be captured and analysed.

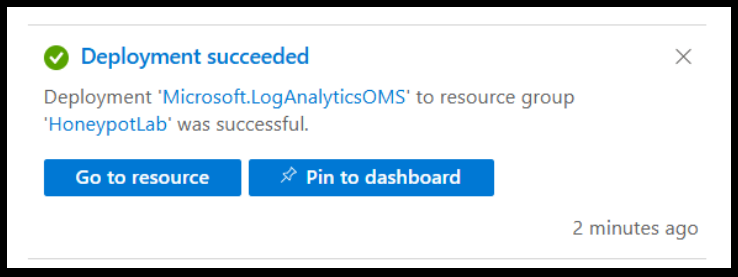
2.2 Creating a Log Analytics Workspace under the VM

The Log Analytics Workspace "*LAW-honeypot1*" was created in the Australia East region. The workspace is responsible for collecting and storing all logs generated by the honeypot VM, enabling effective monitoring and querying for security events. By associating this workspace with the honeypot VM, logs related to login attempts and security events are centrally stored for analysis.

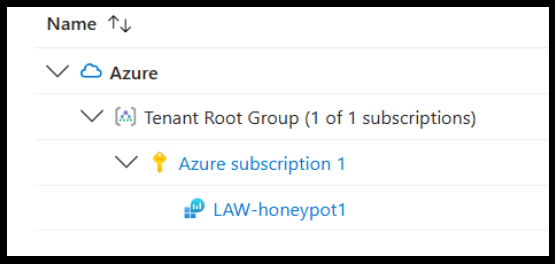


Workspace successfully created:





Full breakdown of the structure of the storage of the honeypot VM and its corresponding log analytics workspace, stored in the Azure cloud:

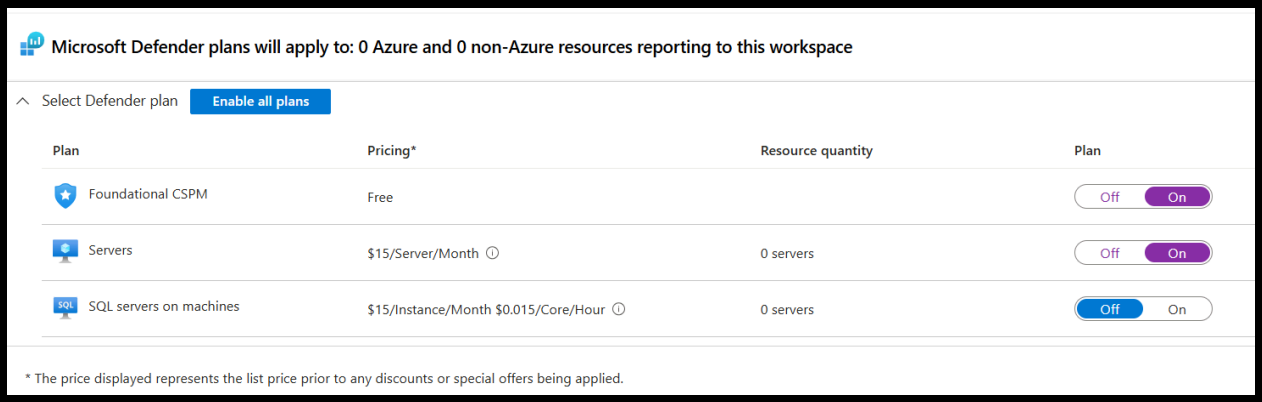


2.3 Configuring Security Settings

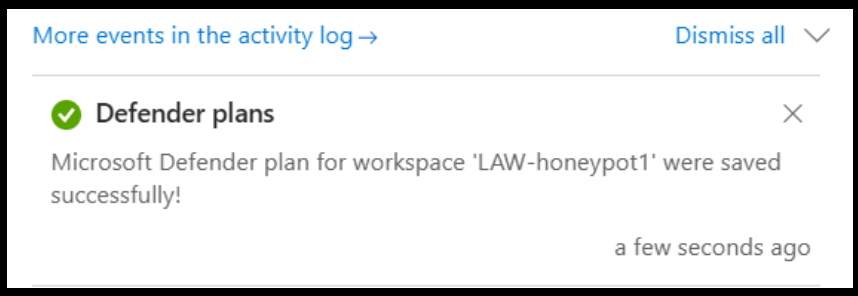
Microsoft Defender is enabled for ***LAW-honeypot1*** to ensure that all relevant security events, such as login attempts, are captured. The settings ensure that both successful and failed login attempts are logged in *Event Viewer* for further investigation.

This configuration is critical for accurately identifying attack patterns.

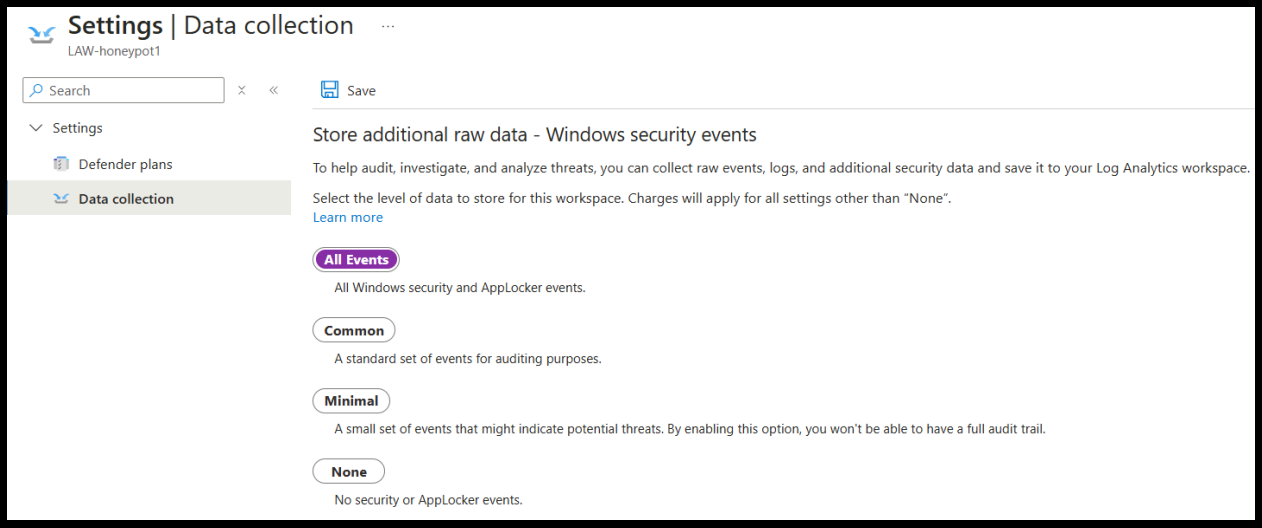
- Correctly configured *Microsoft Defender* settings for *LAW-honeypot1*



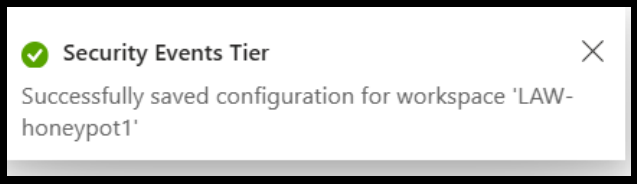
- *Microsoft Defender* Settings successfully saved:



By checking ‘All Events’, the VM’s Event Viewer will display all log activity.



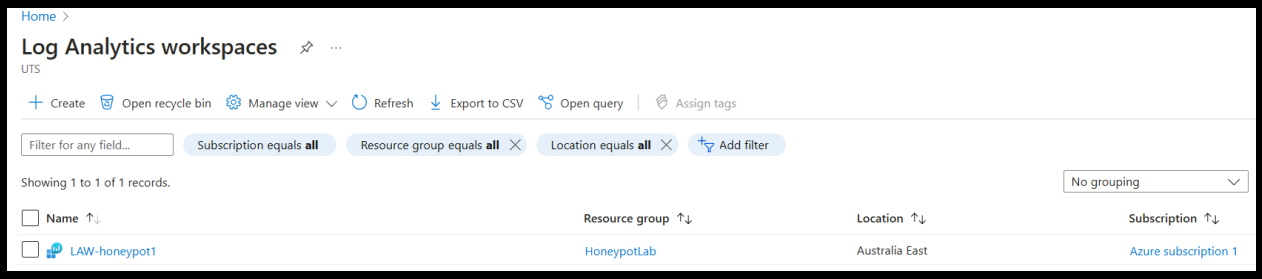
Successful configuration of *LAW-honeypot1*:



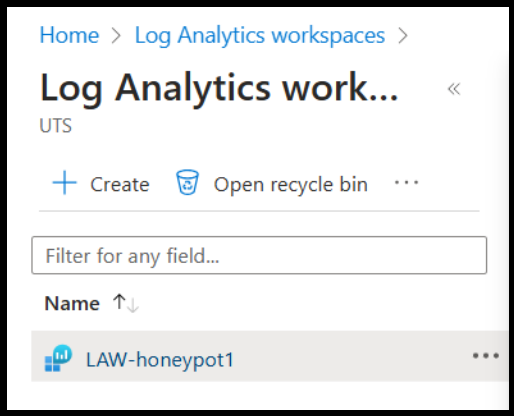
2.4 Connecting *honeypot-vm* to its Workspace

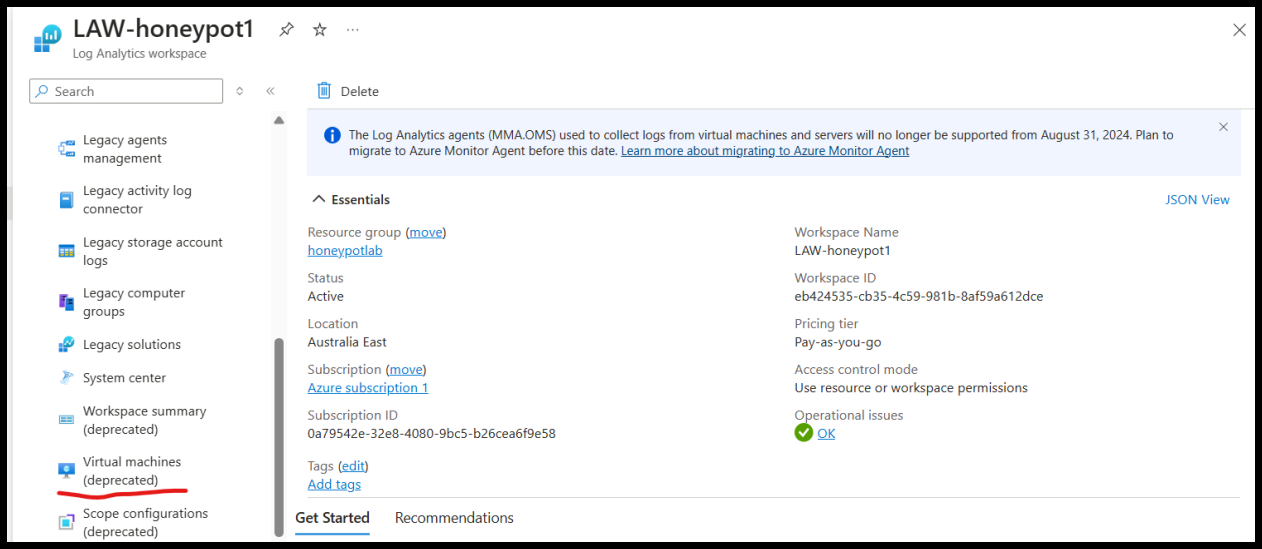
Once connected to the workspace, the honeypot VM will begin transmitting event logs in real-time. The connection between the VM and the workspace is crucial to ensure that all activity, including failed login attempts, is continuously monitored and captured for analysis in Azure Sentinel.

- Navigate to **Log Analytics Workspaces:**

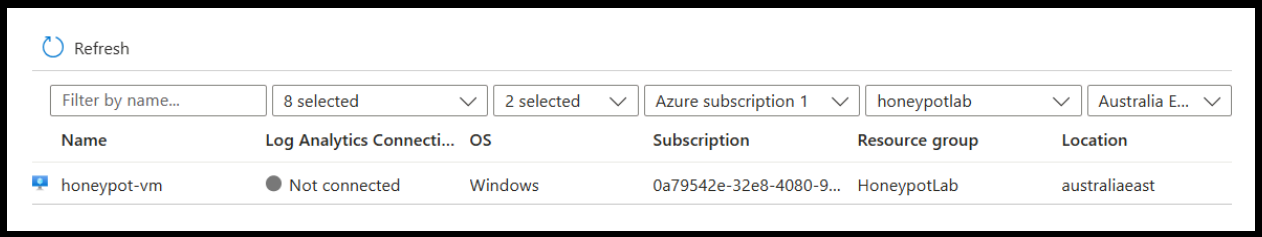


- Select ***LAW-honeypot1***

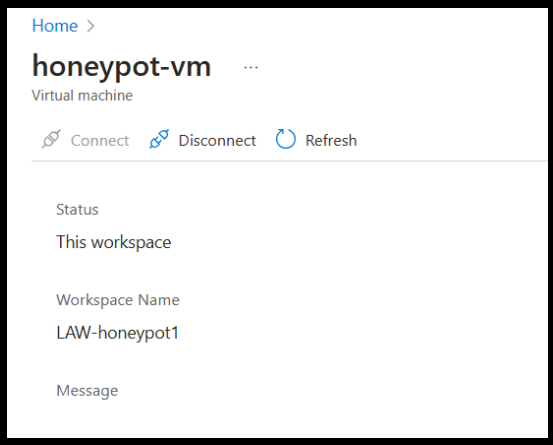




- Connection status displayed



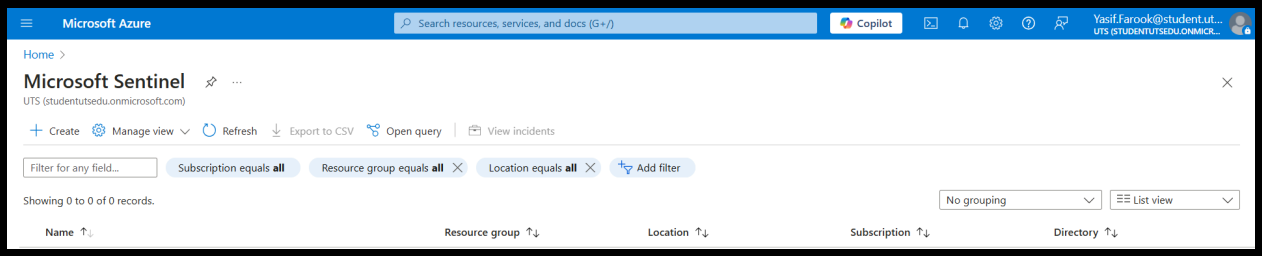
Connections may take a moment to finalise, but once connected, the *'Connect'* option will grey out, and the option to disconnect the virtual machine will appear.



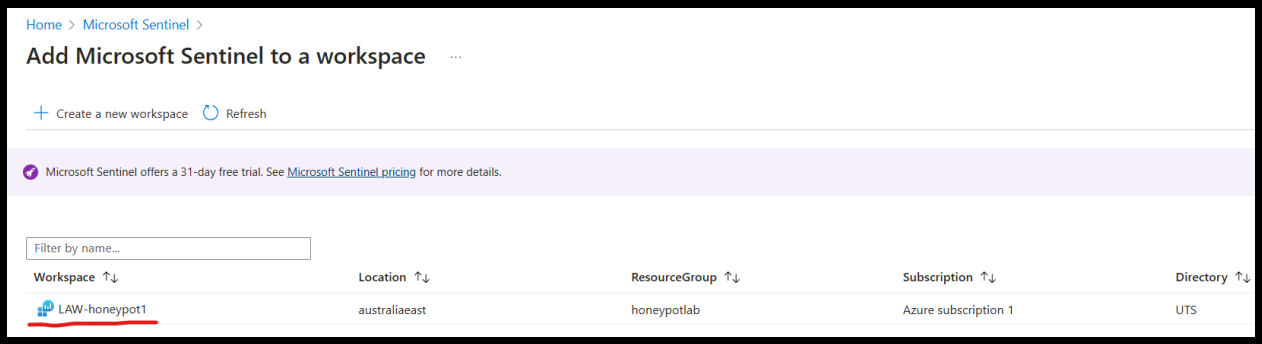
3.0 Integrating Sentinel to the Workspace

Microsoft Sentinel is the SIEM tool that will log all event data, as well as generate real-time alerts of activity relating to the honeypot VM. Integrating a cloud-based solution for a SIEM tool allows for greater ease of access, with its advanced capabilities in automated threat detection and alerting allow for immediate identification of suspicious activity within the honeypot environment.

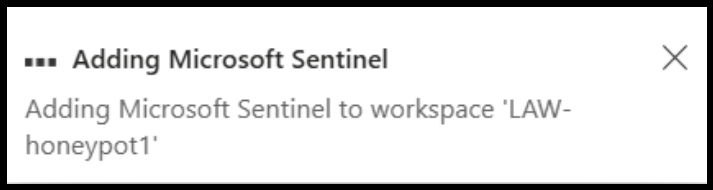
3.1 Navigating to Microsoft Sentinel



3.2 Adding Sentinel to the *LAW-honeypot1* Workspace



The following message should appear once the system begins integrating Sentinel:

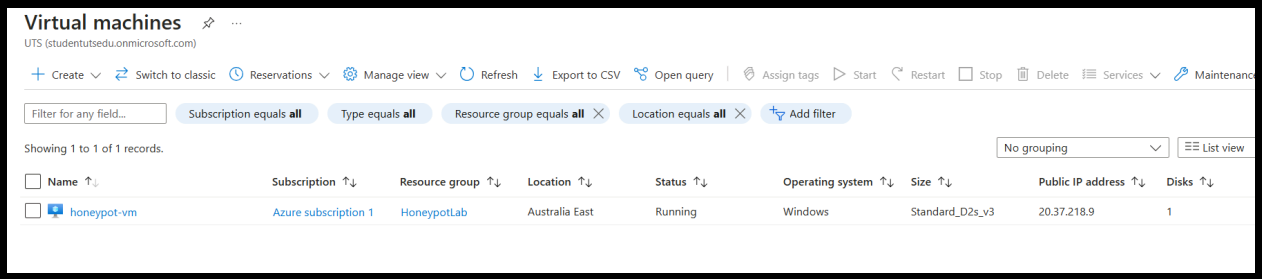


This step significantly enhances the security monitoring capability by leveraging Sentinel's detection rules and AI-driven insights.

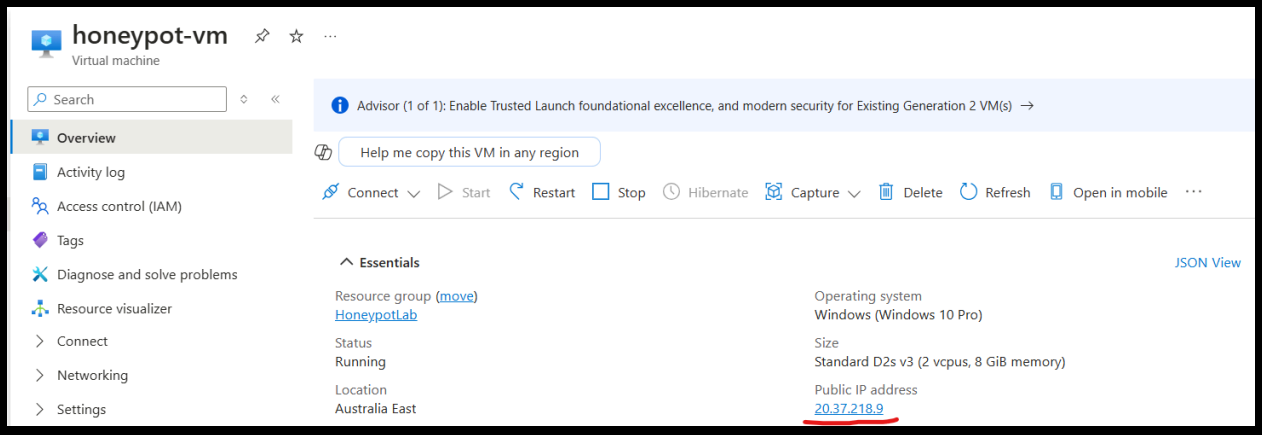
4.0 Starting the Virtual Machine

4.1 Navigating to Virtual Machines in Azure

Before opening up the virtual machine, use the local host to navigate to the list of machines created on Azure.

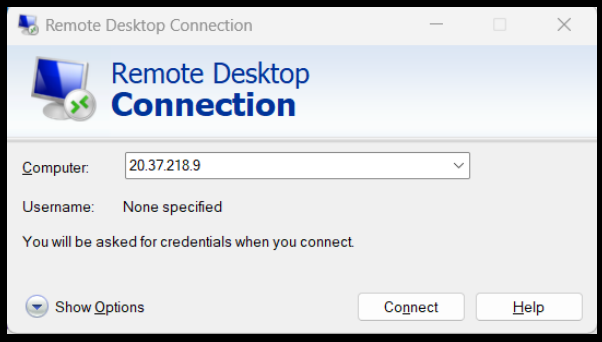


Using the details listed on *honeypot-vm*, a connection to the VM can be established. The public IP address below is listed as *20.37.218.9*. This is the address that will be used to connect to the VM.

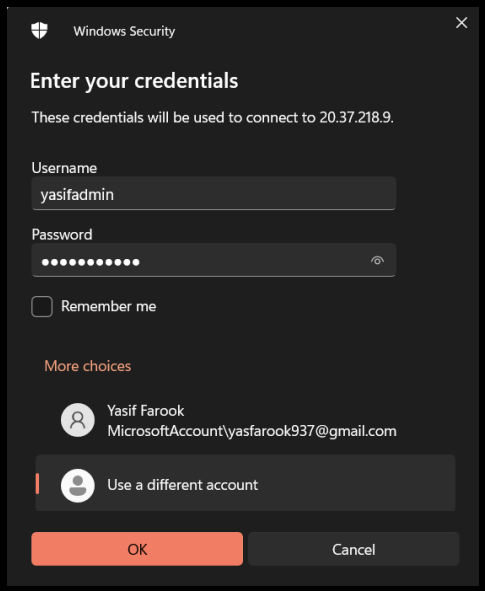


4.2 Connecting to the Virtual Machine

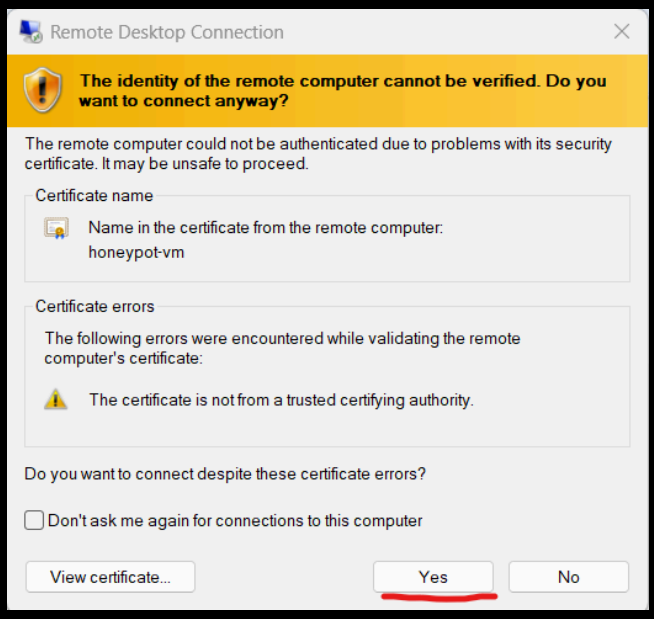
1. Open Remote Desktop Connection on local device, and connect using the Public IP address of the virtual machine:



2. Login using Azure credentials (same credentials used to create the VM and workspace):

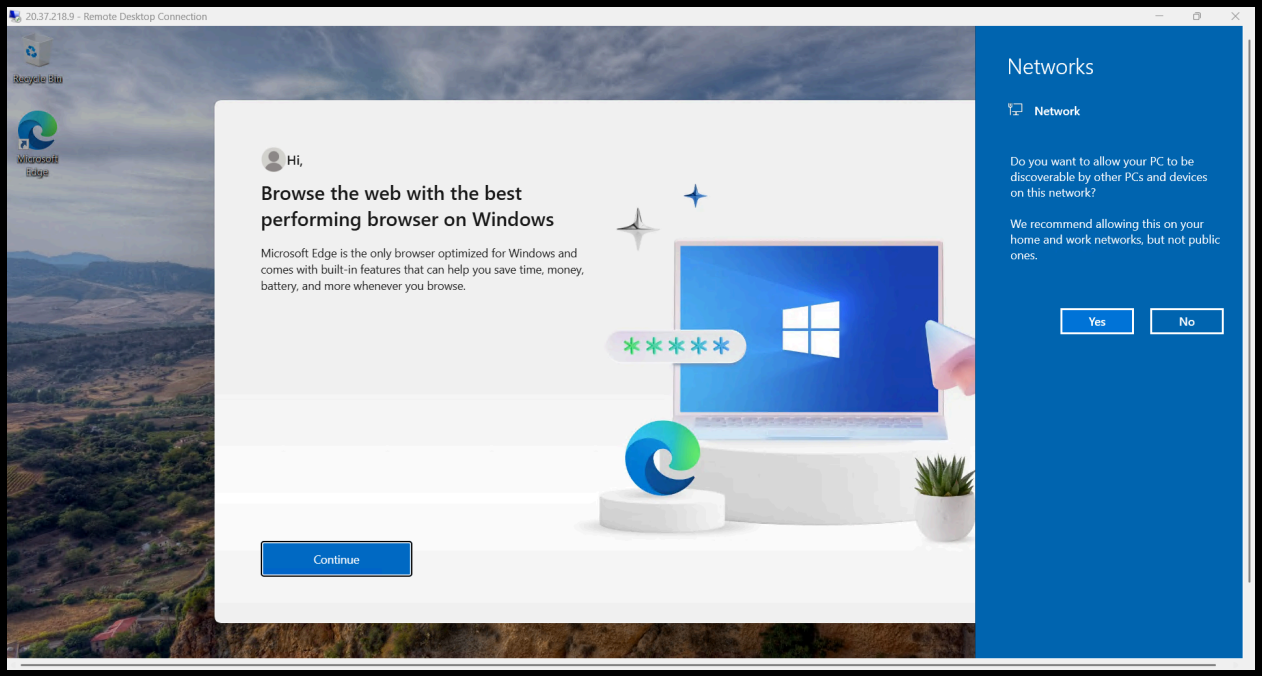


3. Accept Certificate to continue connecting:



4. Successfully connected:

Inside the Virtual Machine (IP Address displayed on the top left of the screen):

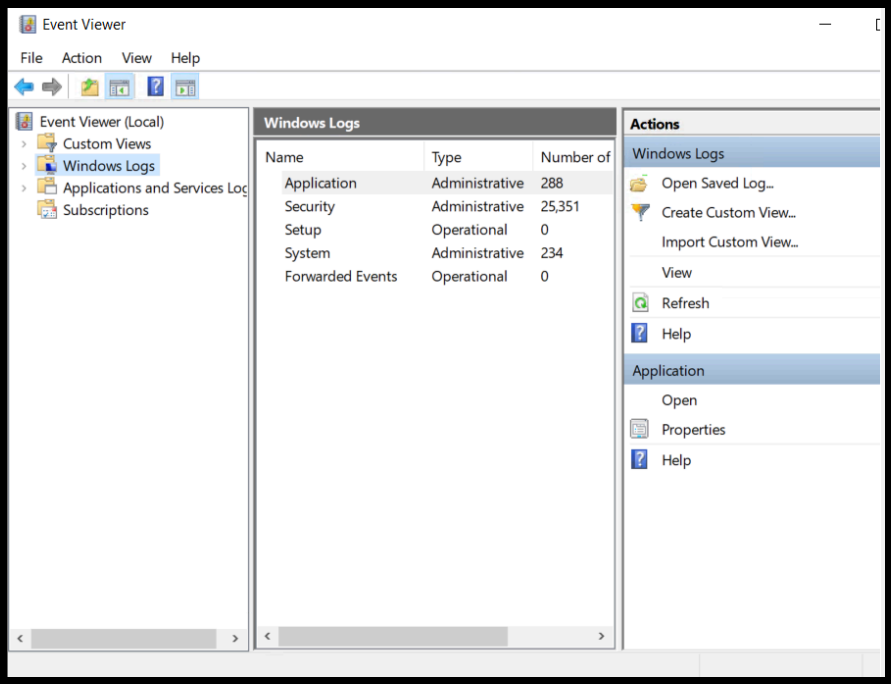


5.0 Monitoring Inside the VM

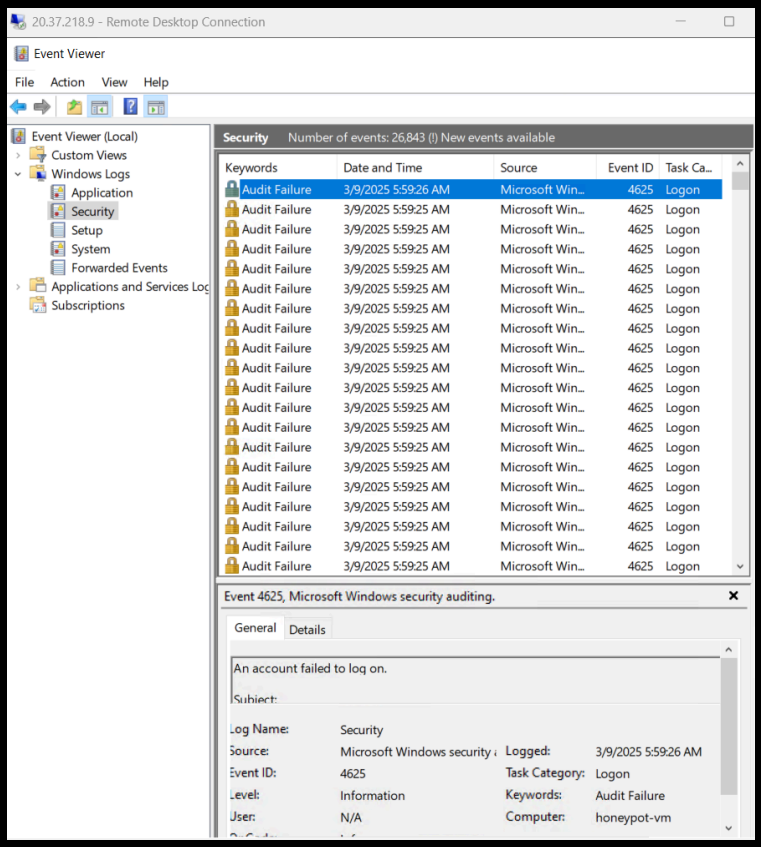
5.1 EventViewer

Inside the Virtual Machine for *20.37.218.9*, open EventViewer.

EventViewer inside the VM:

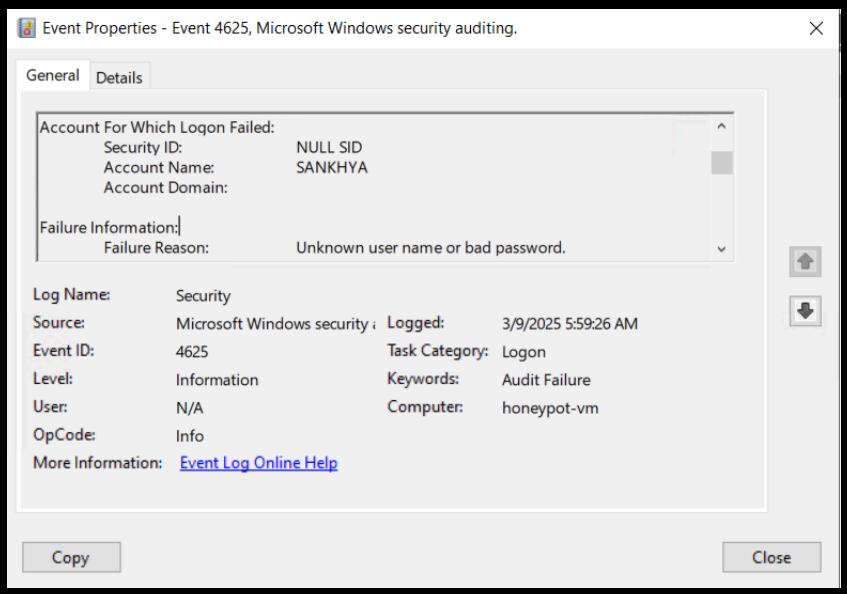


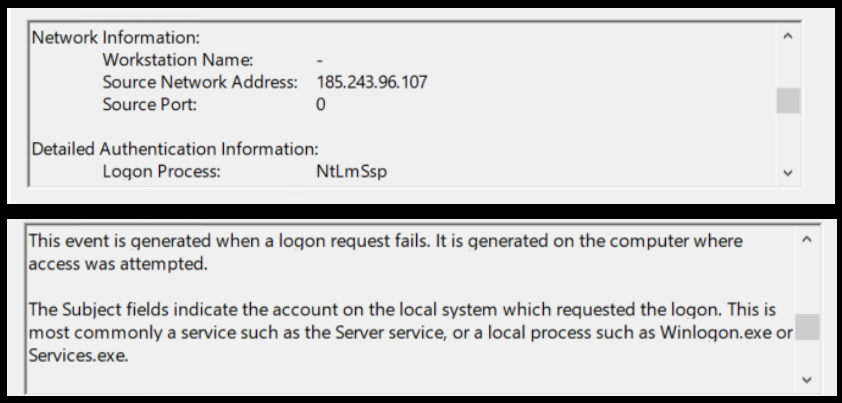
Check security events on this VM via navigating to Windows Logs > Security:



5.1.1 Monitoring Security Events Inside EventViewer

- Inside Security Event ID 4625, occurring at 3/9/2025 5:59:26 AM:

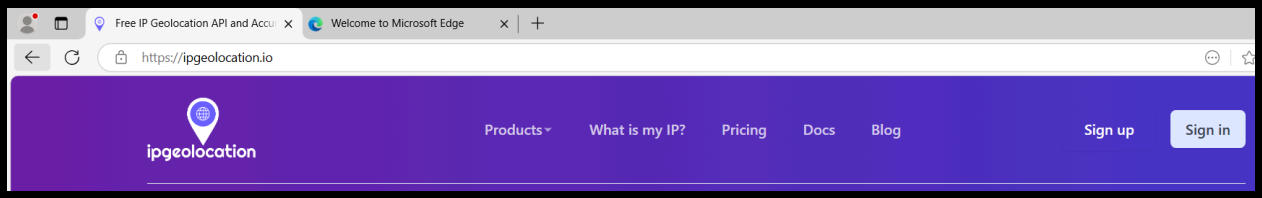




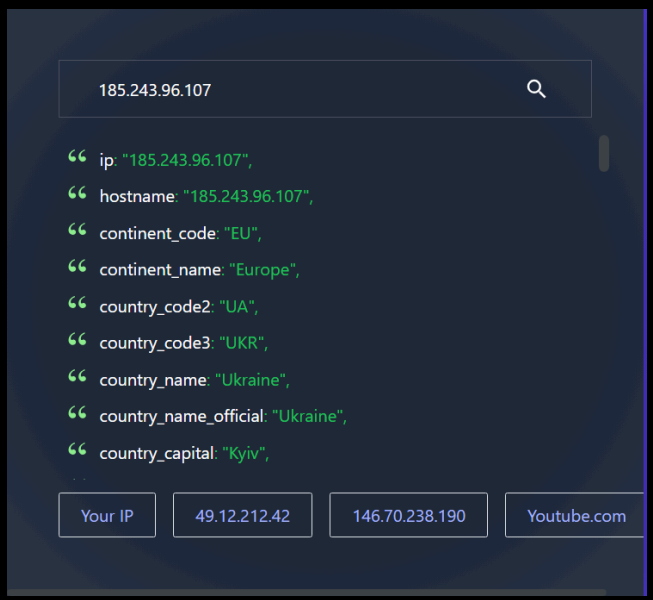
The alert suggests a failed login attempt. This is confirmed by tagging the event with the keyword “Audit Failure”, and categorising the task as an attempted logon.

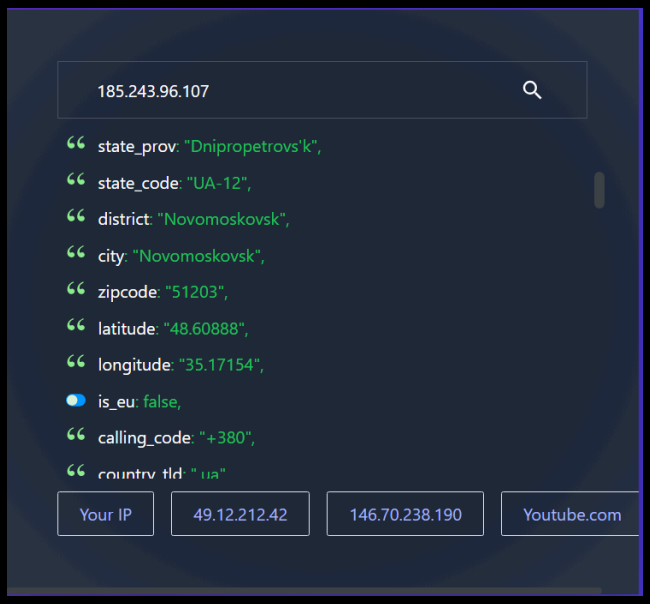
5.1.2 Integrating Geolocation API

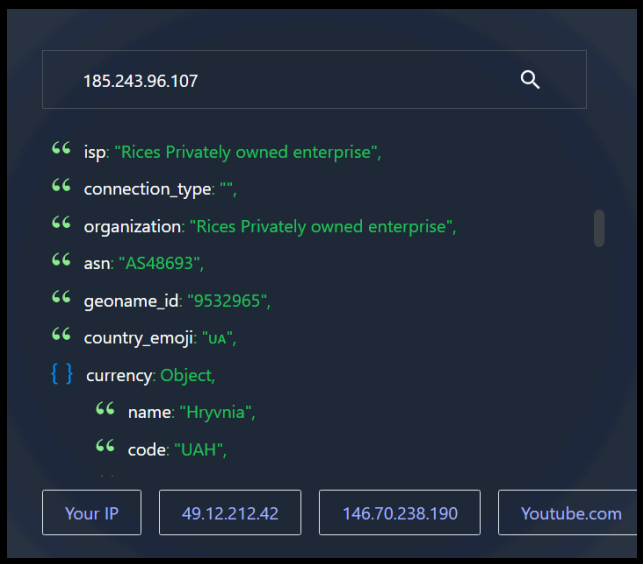
More investigations behind the failed login attempt can be made. An IP Geolocation API, can use the IP address behind the attempt, to reveal geological details such as country, state/province, city as well as coordinates. For example, for the user SANKHYA, from the failed logon above.



- Results returned inside the API:

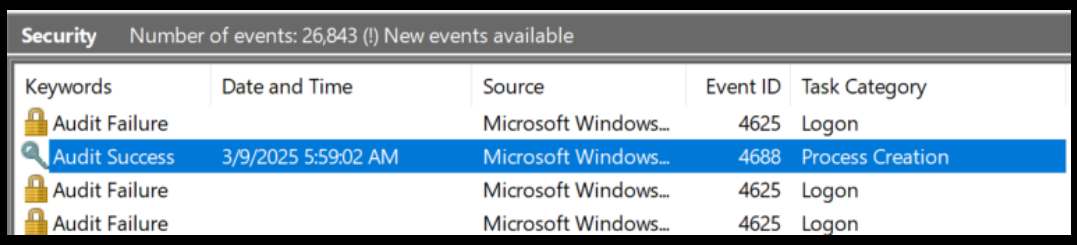




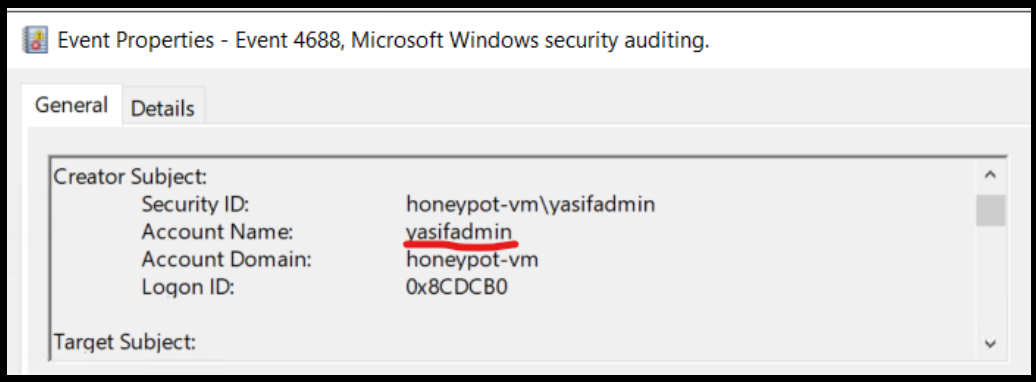


5.1.3 Monitoring More Events Inside EventViewer

- More details on Event ID 4688, occurring at 3/9/2025 5:59:02 AM.

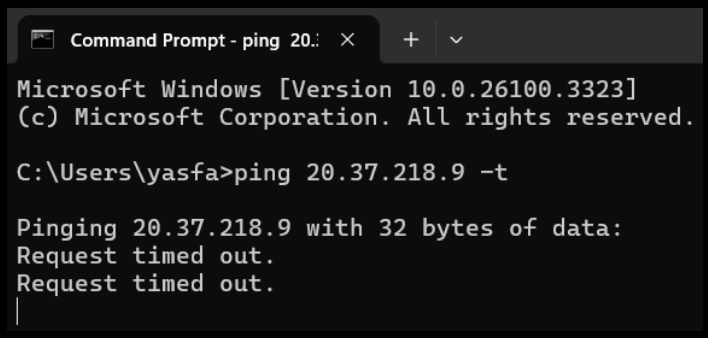


Alert provides contextual information behind the process, that verifies the user behind creating the process (myself):

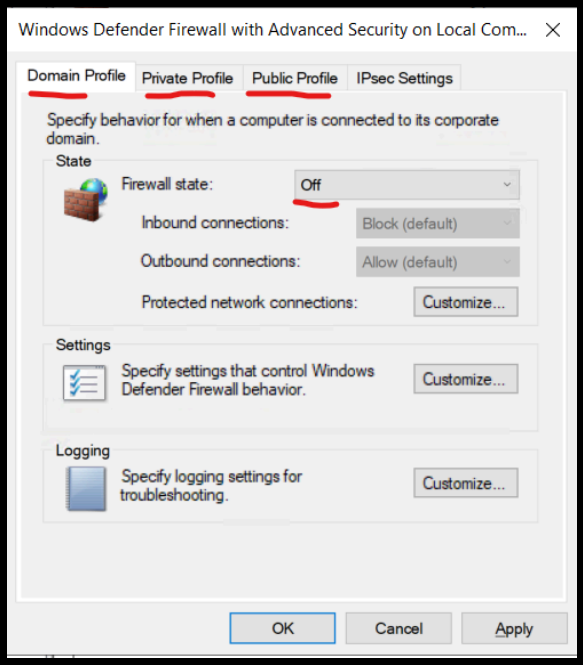


5.2 Configuring the VM Firewall

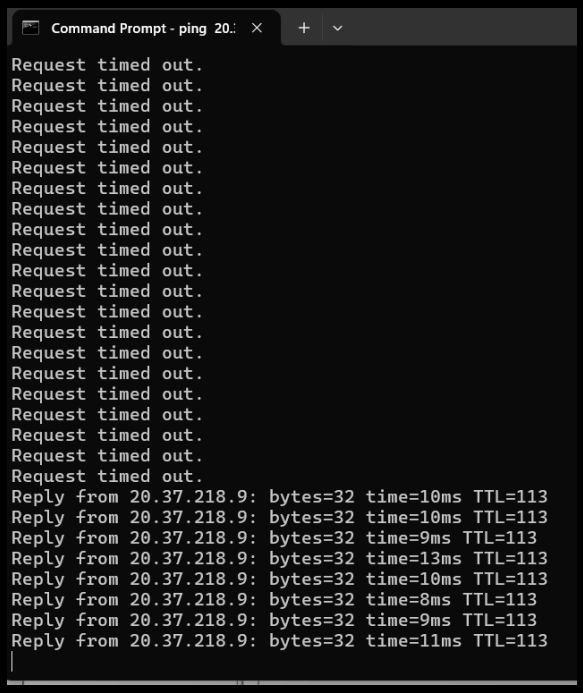
When attempting to ping the virtual machine from the local device, requests time out (due to the VM's firewall settings)



Configure the firewall inside the VM to negate this. Start by entering 'wf.msc' on the start menu of the VM. On Firewall properties, turn the Firewall state to Off on Domain, Private and Public Profiles.

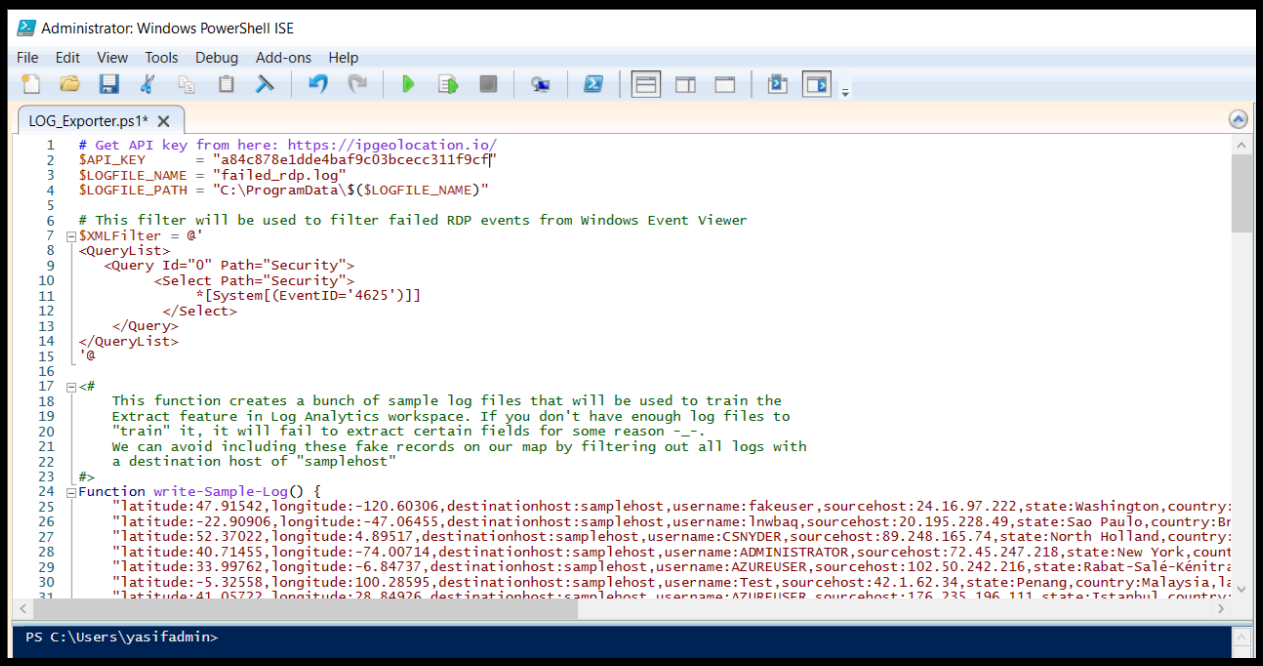


Pings are now successful:

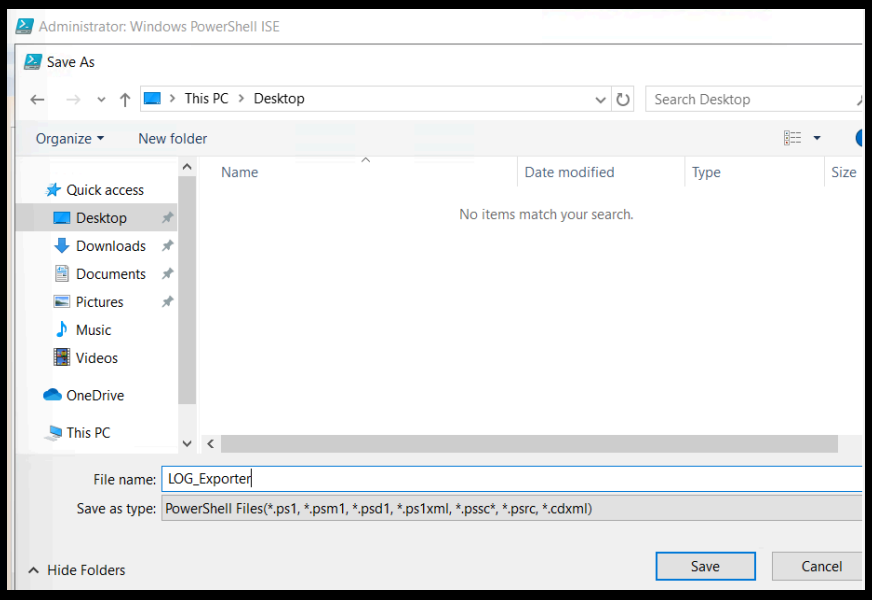


5.3 Generating Logs Inside the VM

Generate logs using open-source PowerShell Script.



- Saving script:



The purpose of the script is to run a loop that initiates continuous event monitoring. This relates to any activity regarding the *honeypot-vm*, such as failed/successful login attempts, as well as connecting the VM to the Azure Cloud (as done in this project)

LOG_Exporter.ps1*

```
1   # Get API key from here: https://ipgeolocation.io/
2   $API_KEY      = "a84c878e1dde4baf9c03bcecc311f9cf"
3   $LOGFILE_NAME = "failed_rdp.log"
4   $LOGFILE_PATH = "C:\ProgramData\$($LOGFILE_NAME)"
5
6   # This filter will be used to filter failed RDP events from Windows Event Viewer
7  $XMLFilter = ...
16
17 <#...#>
24 Function write-Sample-Log() {...}
37
38  # This block of code will create the log file if it doesn't already exist
39 if ((Test-Path $LOGFILE_PATH) -eq $false) {...}
43
44  # Infinite Loop that keeps checking the Event Viewer logs.
45  while ($true)
46 {...}
```

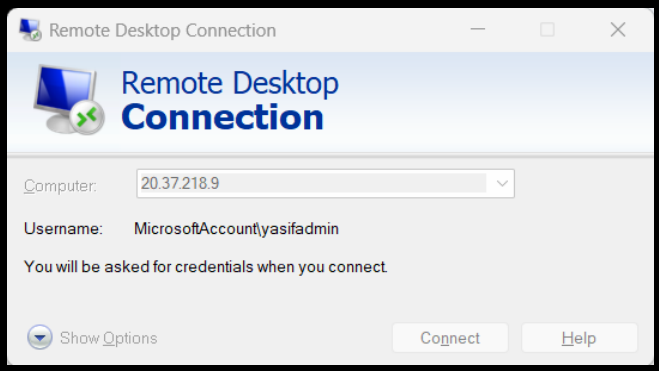
5.3.1 Running Script

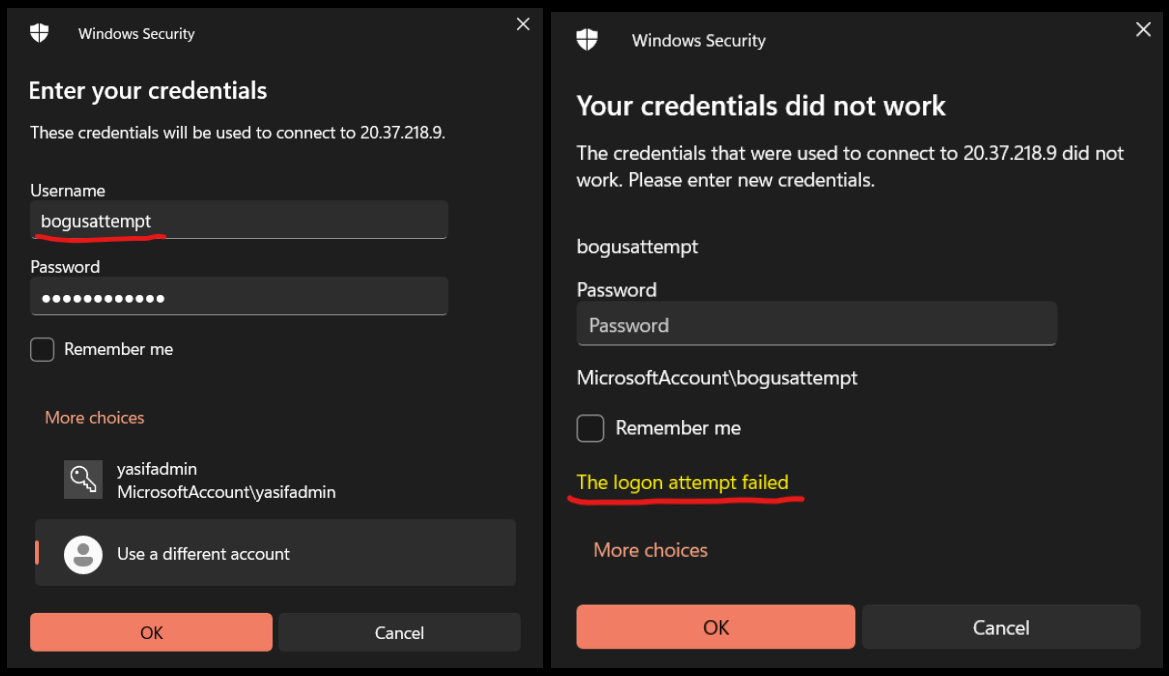
- Running the script, waiting for activity to be recorded and logged:

```
    Directory: C:\ProgramData

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   7:05 AM              0 failed_rdp.log

PS C:\Users\yasifadmin> C:\Users\yasifadmin\Desktop\LOG_Exporter.ps1
```

5.3.2 Verifying Event Monitoring

To verify activity is being correctly logged and script is running, attempt a failed login to the VM, using a bogus username and password from another instance of Remote Desktop Connection. This again, can be launched from the local host. See below for confirmation that the attempt was correctly monitored and logged.

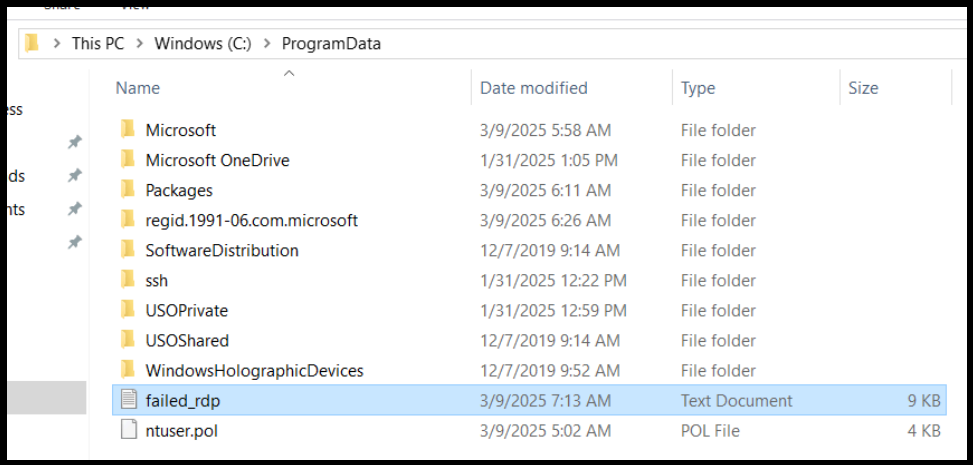




5.3.3 Preparing the Log File

The Powershell script aims to generate a recurring *.log* file that prints out all activity logged on EventViewer, and the Powershell terminal.

Failed_rdp.log file created upon running the script. This file keeps track of all event activity regarding the VM:



Upon inspecting the file, the intentional failed login attempt using bogus credentials can be verified:

```
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:CHUCK,sourcehost:92.63.197.9,state:Nor
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
latitude:52.37022,longitude:4.89517,destinationhost:honeypot-vm,username:bogusattempt,sourcehost:61.68.142.168,
```

Login attempts have appeared on the terminal:

LOG_Exporter.ps1

```
$API_KEY      = "a84c878e1dde4baf9c03bcecc311f9cf"
$LOGFILE_NAME = "failed_rdp.log"
$LOGFILE_PATH = "C:\ProgramData\$($LOGFILE_NAME)"

# This filter will be used to filter failed RDP events from Windows Event Viewer
$XMLFilter = ...

<#...#>
Function write-Sample-Log() {...}

# This block of code will create the log file if it doesn't already exist
if ((Test-Path $LOGFILE_PATH) -eq $false) {...}

# Infinite Loop that keeps checking the Event Viewer logs.
while ($true)
{...}
```

```
.169,timestamp:2025-03-09 06:32:01
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:ROSELY,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:32:00
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:ACTECK,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:58
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:SITI,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:56
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:A,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:54
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:SEVGI,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:53
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:CAHIT,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:51
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:BASAK,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:49
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:GIZEM,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:48
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:SEMIH,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:46
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:MUHASEBE6,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:44
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:MAHMUT,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:42
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:UGUR,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:40
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:RECEP,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:38
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:ECZANE,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:36
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:KUDRET,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:34
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:ERCAN,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:32
latitude:52.13263,longitude:5.29127,destinationhost:honeypot-vm,username:OZGUR,sourcehost:185.156.73.169,state:North Holland,label:Netherlands - 185.156.73.169,timestamp:2025-03-09 06:31:30
```

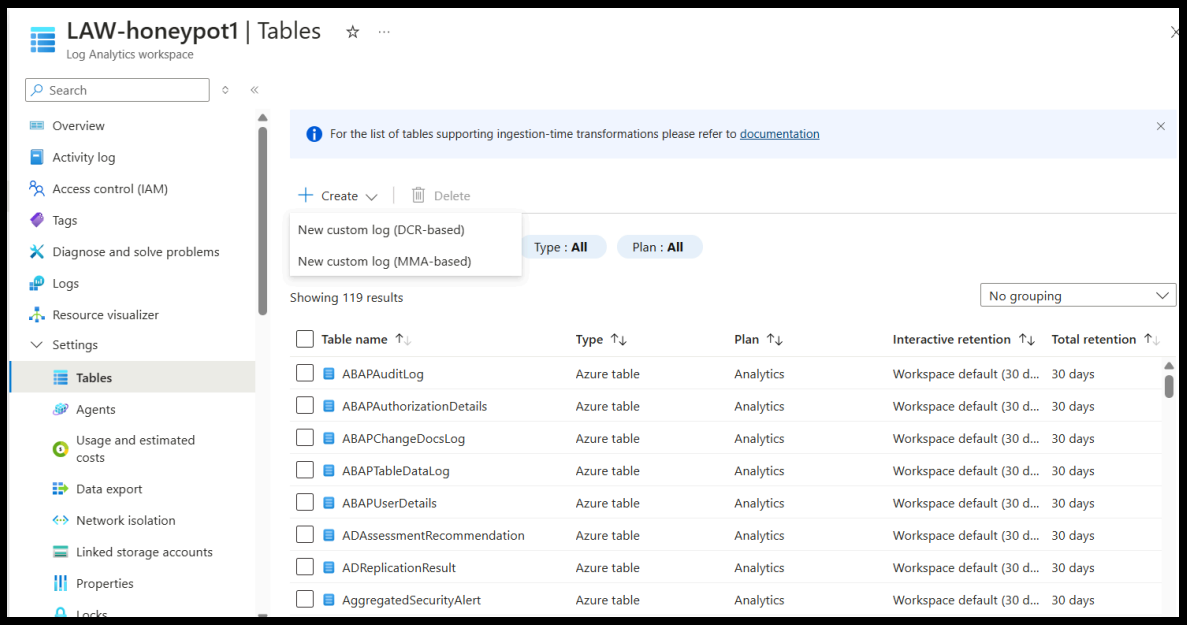
6.0 Upgrading Logs

With the use of Azure, the logs generated by the Powershell script can be optimised to allow for greater efficiency, as well as more in-depth analysis and viewing.

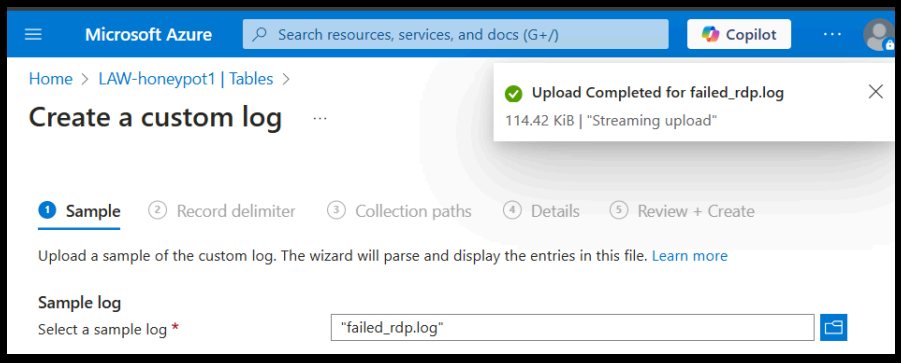
6.1 Creating a custom log in Azure

The following steps are completed on the local machine:

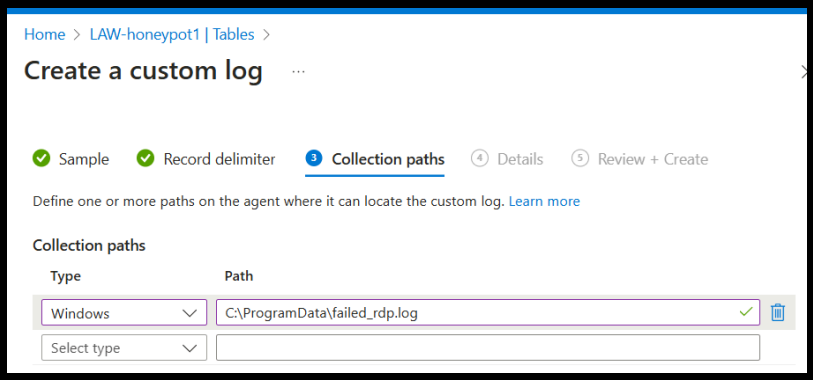
1. Navigate to **Tables** in Azure:

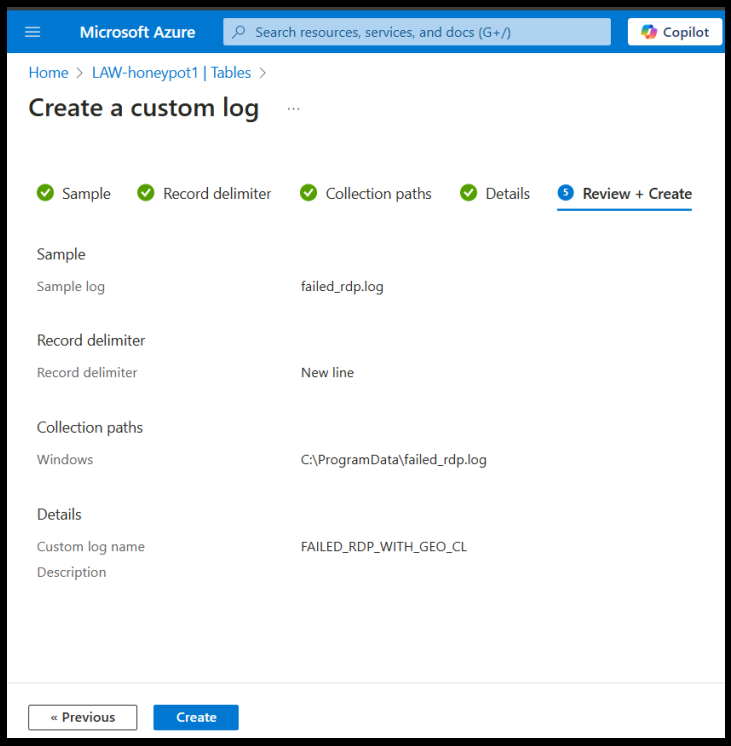


2. Navigate to **Create a custom log**:

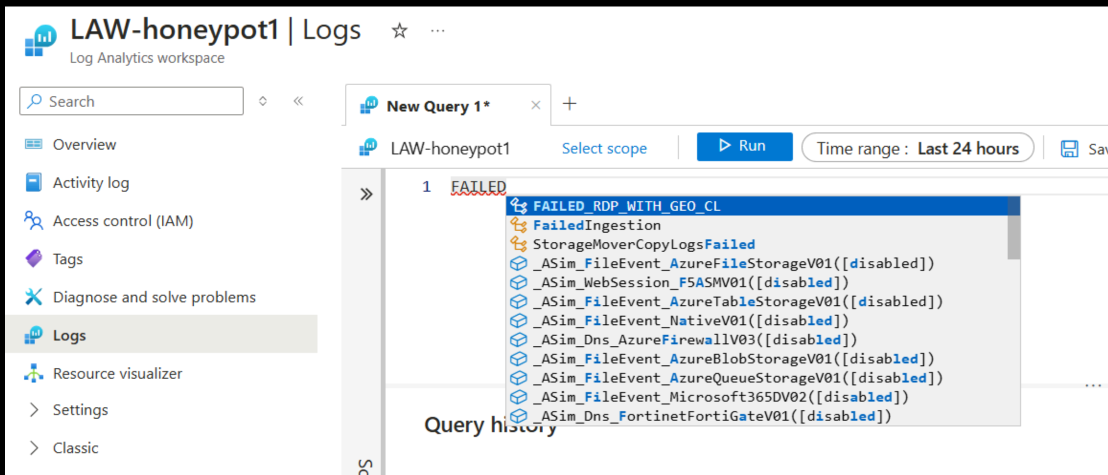


3. Set the correct path to the log file that the Powershell script generates. This path can be found on the file library, back in the VM.

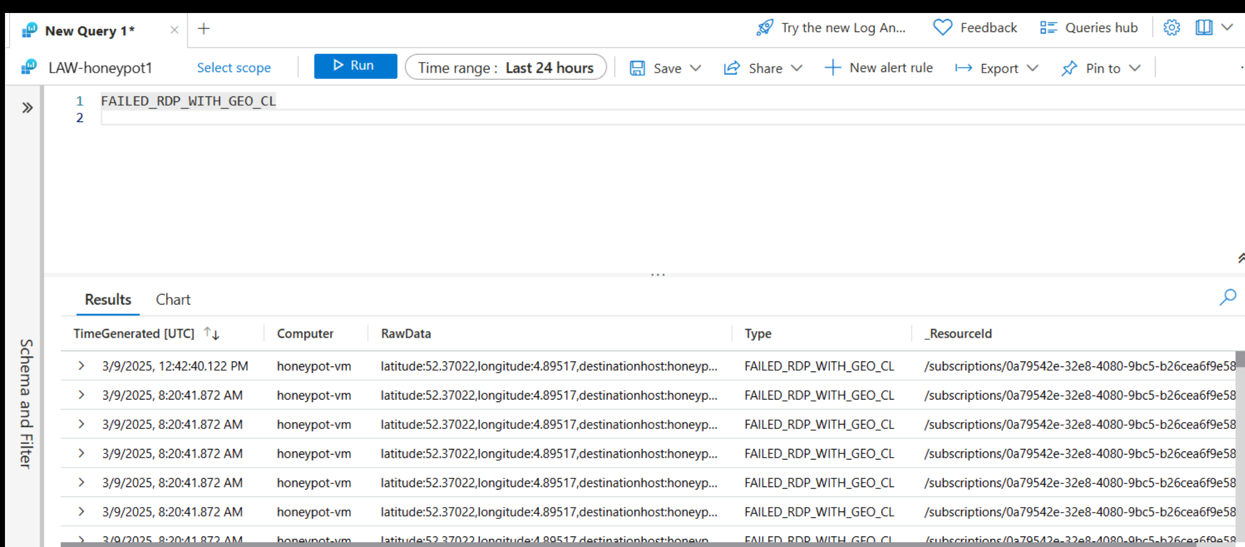




Custom logs can be found in the **Logs** section:

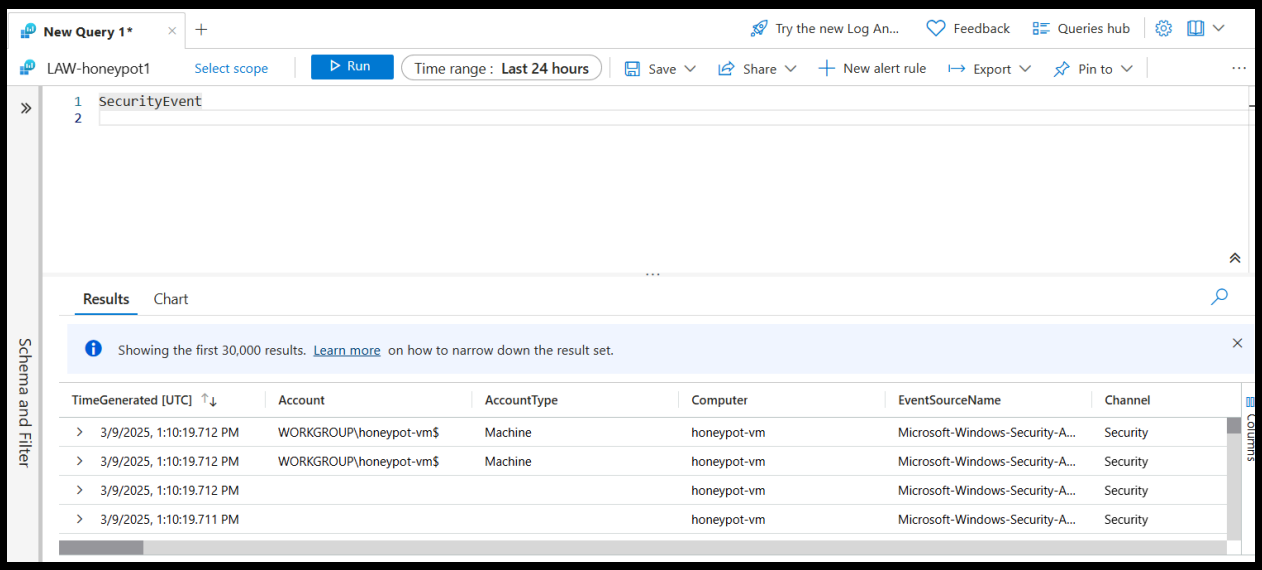


4. Run the selected log:

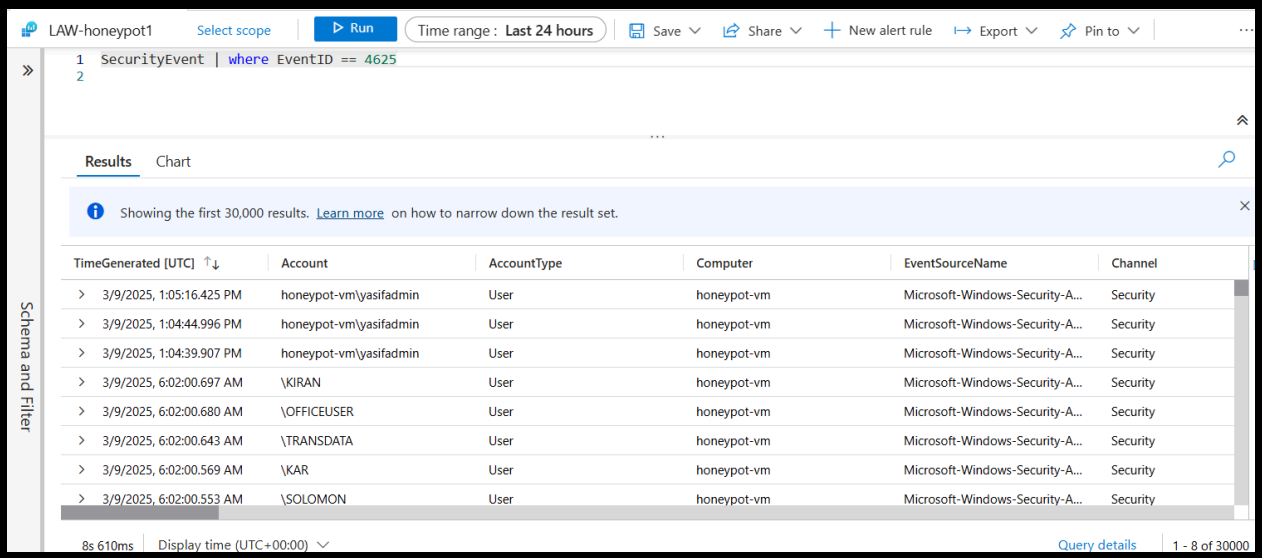


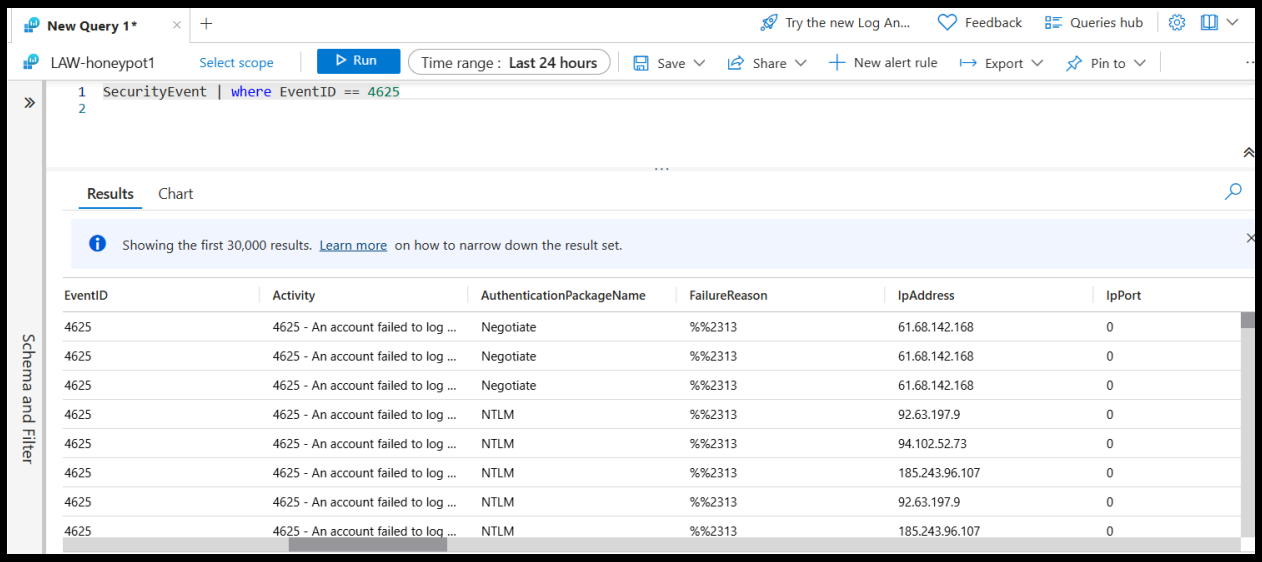
6.1.1 Testing Other Queries

- Run ***SecurityEvent***. This brings the Windows Event Viewer into the Log Analytics Workspace.

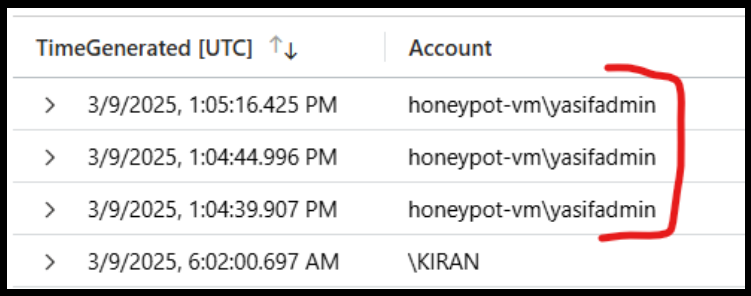


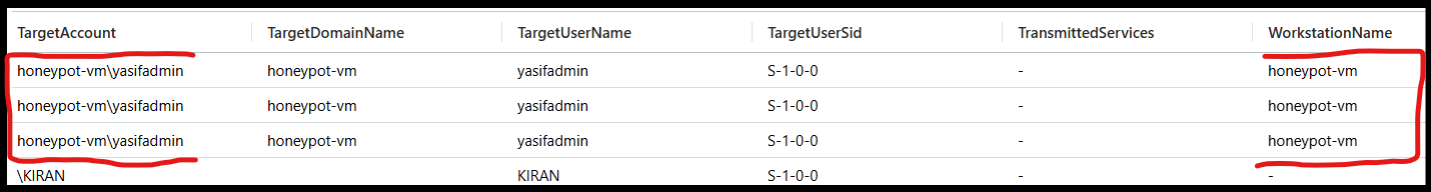
- Reviewing all failed RDP logs (events with EventId 4625):





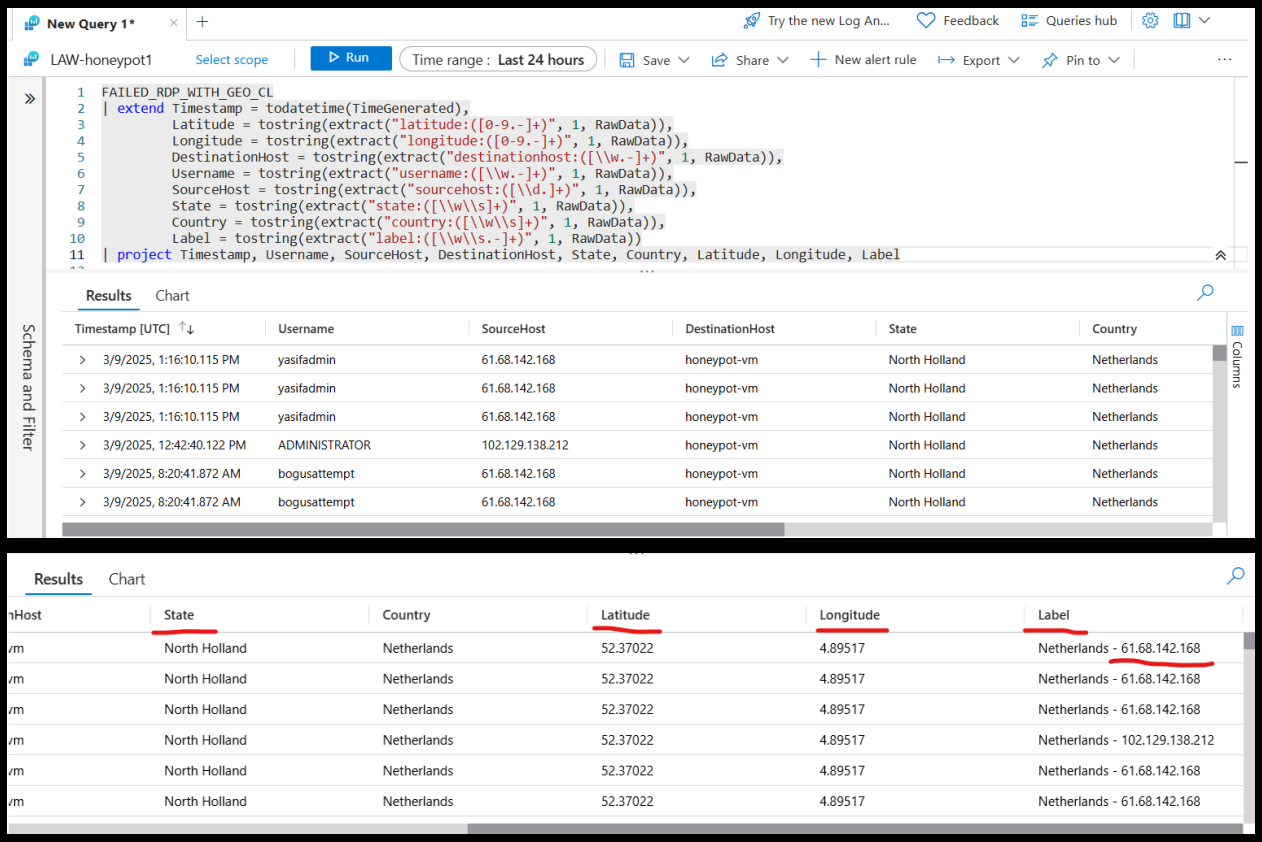
The 3 most recent events (at 1:05 PM), that occurred 7 hours after the previous recorded event (at 6:02 AM), are related to logon attempts made from the admin user themselves (*honeypot-vm\yasifadmin*), who attempted to remember their password without logging into their Google Password Manager. This is further supported by the fact that the logged times align with the admin user attempting to login after not using their laptop (which contained the local host and VM) for 7 hours.





6.1.2 Adding More Fields to Logs for Greater Details

The query below aims to output the log data with greater structure, with finer details such as State, Country and longitude & latitude, in separate fields. This can allow for even greater log analysis and filtering, as the previous query (based on the initial Powershell script), displayed the newly filtered data in a long string format under *EventSourceName*



6.1.3 Breakdown of Updated Fields:

Added elements to the query to extract specific information (Latitude, Longitude, Username, etc.) from the *RawData* field, which stores everything as a long string in a key-value format.

- ***extend*** Clause: The extend keyword is used to create new columns from the existing data. In this case, it creates new fields by extracting values from the RawData string.

- ***Timestamp***: Converted the TimeGenerated field into a readable date and time format using todatetime().

- For each field (Latitude, Longitude, etc.), used *extract()* to extract values from RawData based on specific patterns (like latitude:, longitude:, etc.).

- ***extract()*** Function: This function helps pull out values from the RawData string. The general format is: ***extract("pattern", group_number, RawData)***

  In this structure:

Pattern: A regular expression (regex) that describes the structure of the data that is to be extracted.
Group Number: Specifies which part of the match to return. In this case, 1, as the actual value after the colon (e.g., after latitude:) should be returned.
RawData: The field data is being extracted from.

- *project* Clause*:* Once the required fields have been extracted (Latitude, Longitude, Username, etc.), use the project clause to display only the fields needed.

  | project Timestamp, Username, SourceHost, DestinationHost, State, Country, Latitude, Longitude, Label

  - *project*: Tells the query to display only the columns specified.
  - **Timestamp**: The time of the event.
  - **Username**: The username associated with the event.
  - **SourceHost**: The source IP address (or hostname).
  - **DestinationHost**: The target host (where the login attempt was made).
  - **State** and **Country**: The location of the source.
  - **Latitude** and **Longitude**: The geographical coordinates of the source.
  - **Label**: A label summarising the event.

This update query with separated fields makes analysis and filtering more effective and easier. Specific fields such as *longitude* can be pulled out of the log.

Full query:

```
FAILED_RDP_WITH_GEO_CL
| extend Timestamp = todatetime(TimeGenerated),
    Latitude = tostring(extract("latitude:([0-9.-]+)", 1, RawData)),
    Longitude = tostring(extract("longitude:([0-9.-]+)", 1, RawData)),
    DestinationHost = tostring(extract("destinationhost:([\\w.-]+)", 1, RawData)),
    Username = tostring(extract("username:([\\w.-]+)", 1, RawData)),
    SourceHost = tostring(extract("sourcehost:([\\d.]+)", 1, RawData)),
    State = tostring(extract("state:([\\w\\s]+)", 1, RawData)),
    Country = tostring(extract("country:([\\w\\s]+)", 1, RawData)),
    Label = tostring(extract("label:([\\w\\s.-]+)", 1, RawData))
| project Timestamp, Username, SourceHost, DestinationHost, State, Country, Latitude,
Longitude, Label
```

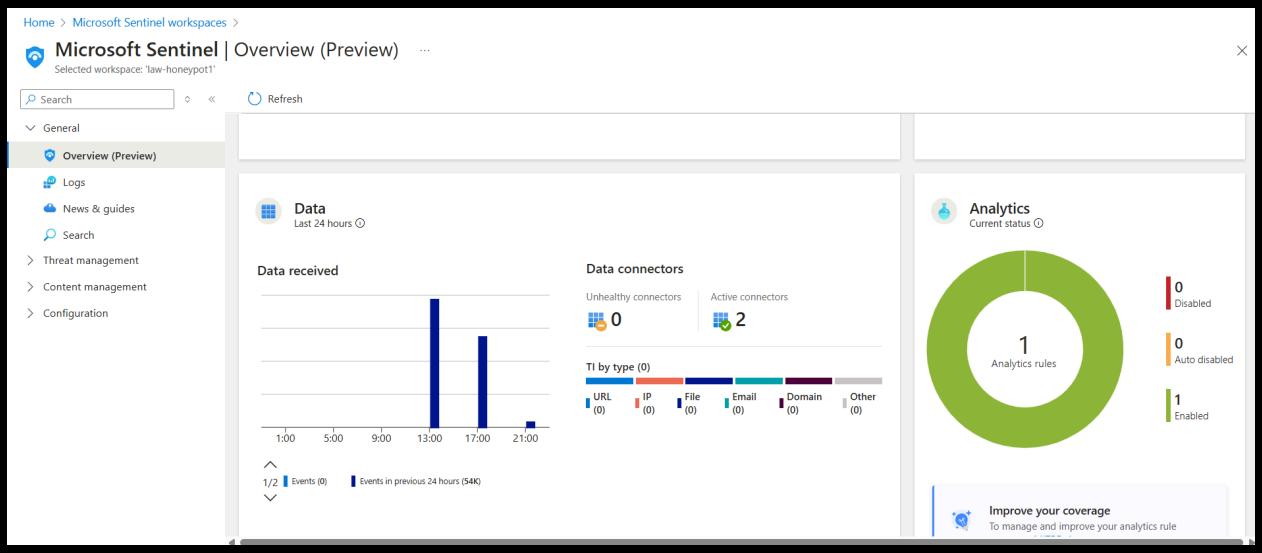
7.0 Visualising Attack Patterns with Azure Sentinel Map

Following on from the previous steps, a map representing log data can be generated.

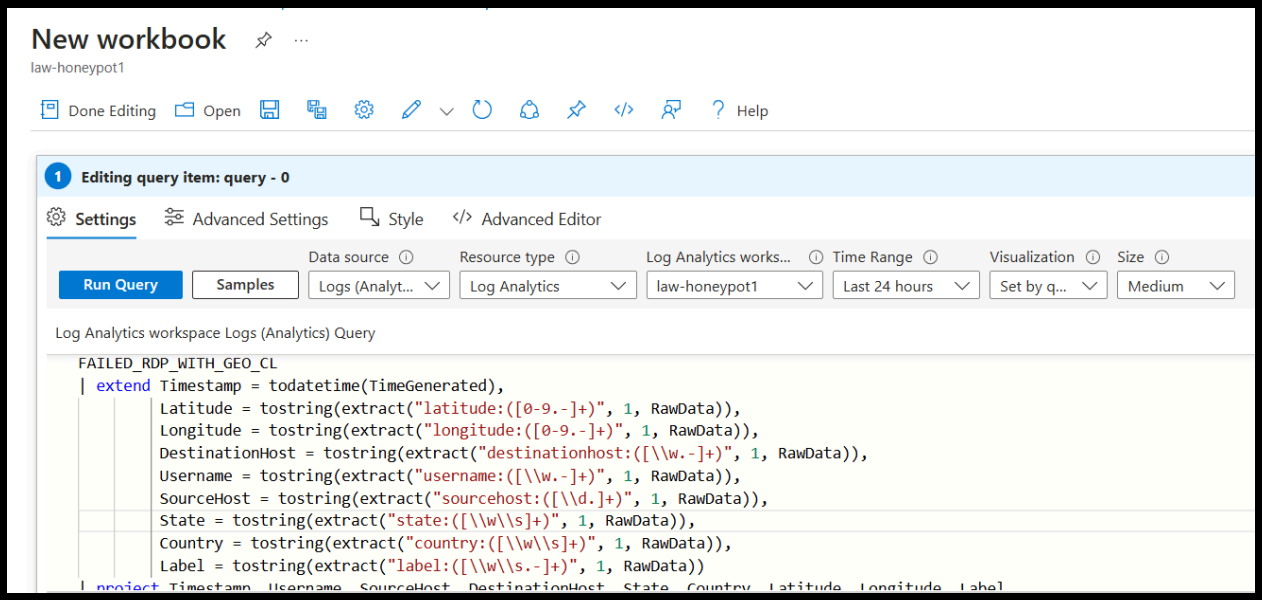
This map will enable real-time monitoring of events, with geographic location-based alerts. The mapping of events can be set by either country, as well as longitude & latitude.

7.1 Integrating Microsoft Sentinel

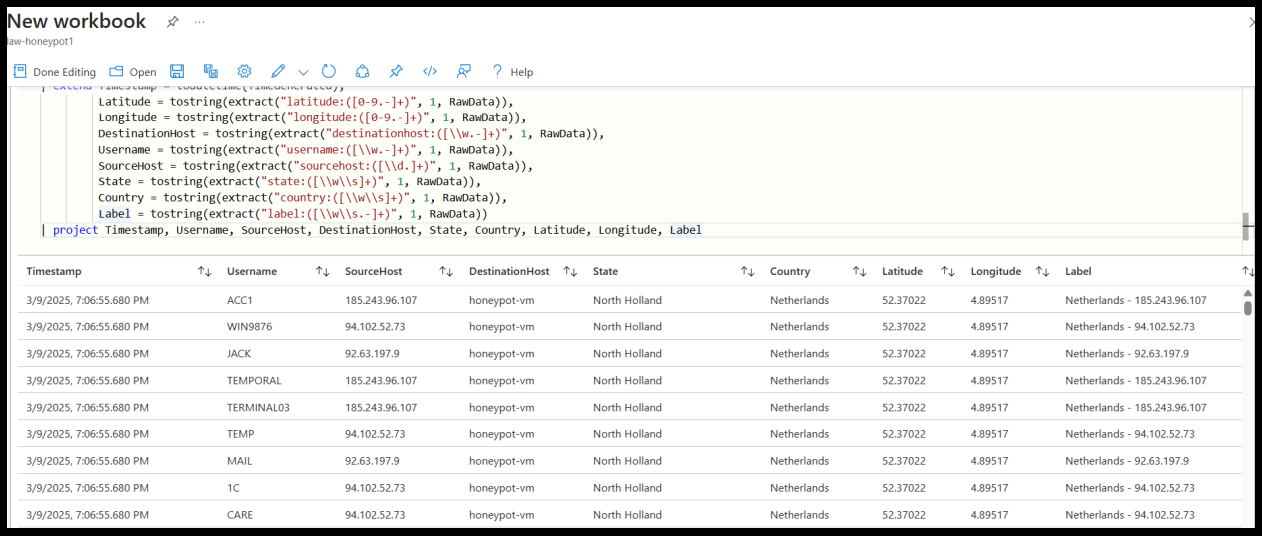
Navigate to *Microsoft Sentinel* (found via the search bar):



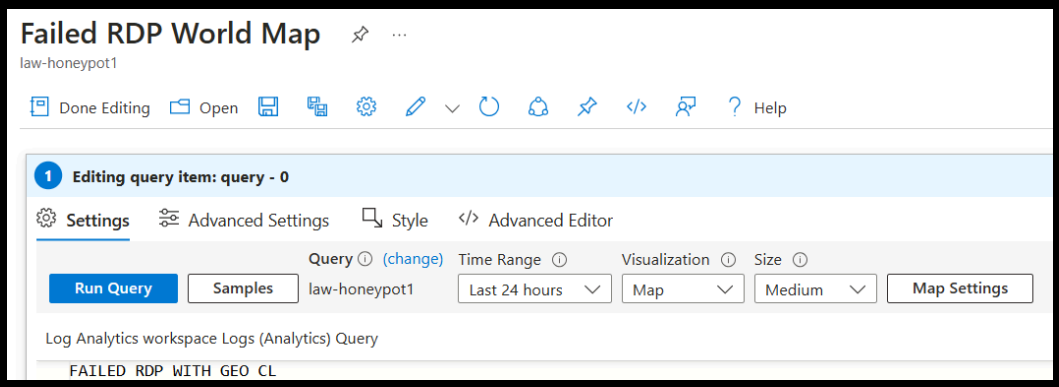
- Adding previous detailed query to new Workbook:



- Running Query:

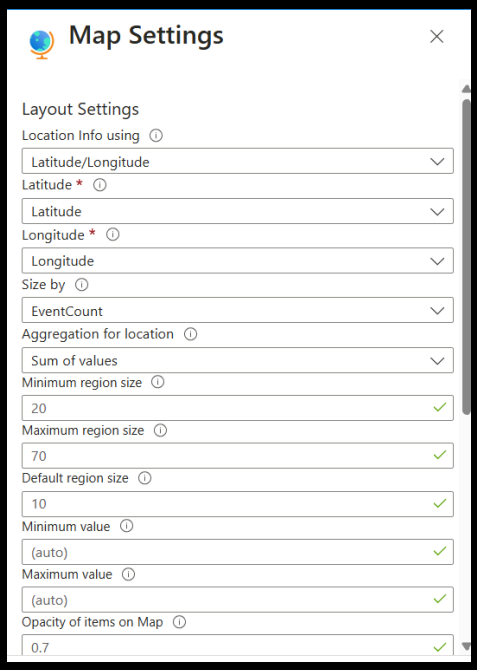


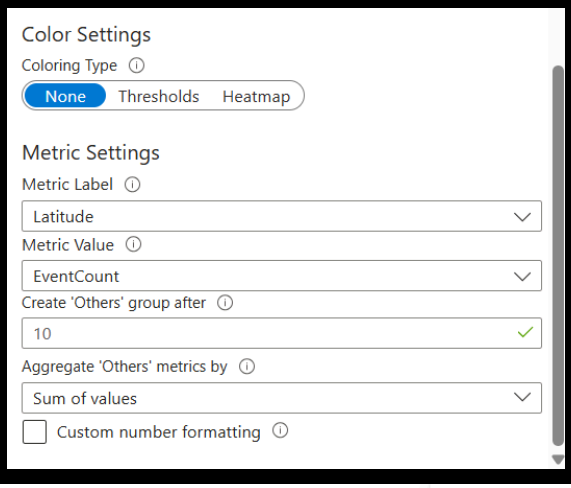
To view the output as a map instead of grid table, switch the **Visualisation** to **Map**.

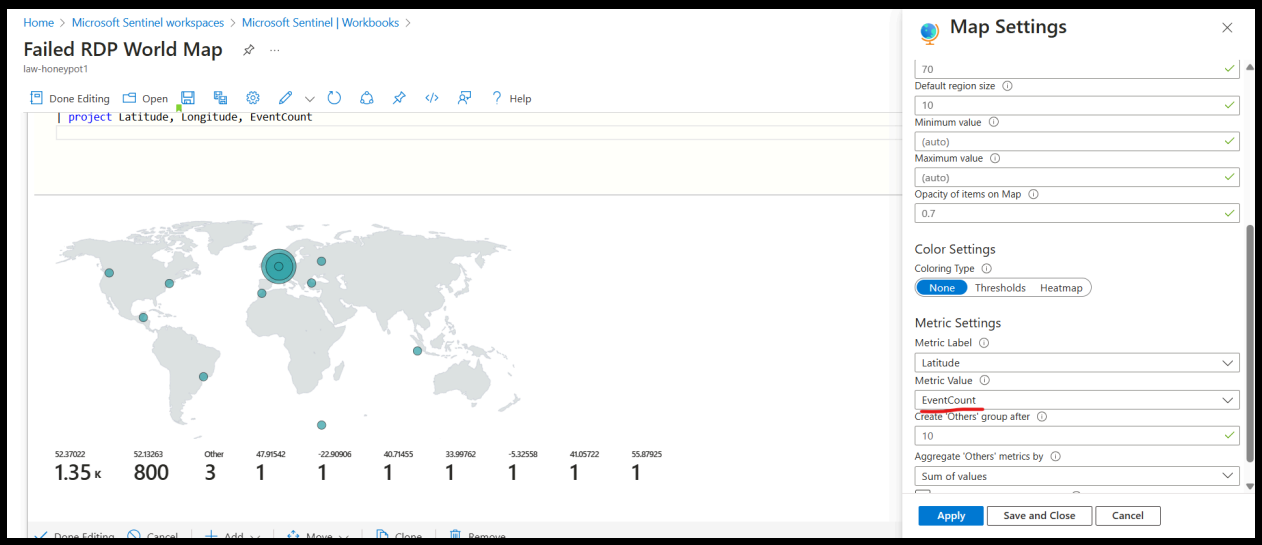


7.2 Latitude & Longitude-Based Map

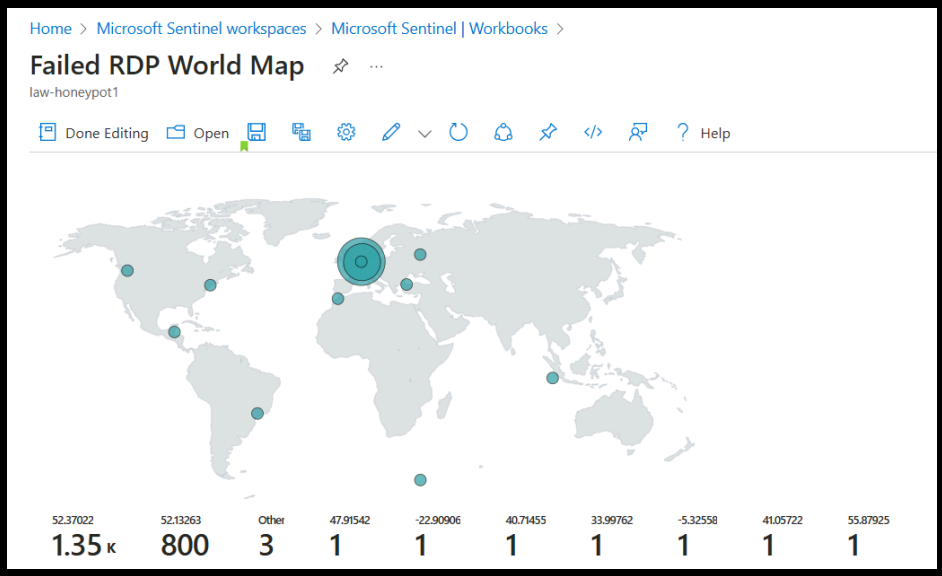
- Adjusting Map settings to accurately account for latitude and longitude as coordinates:





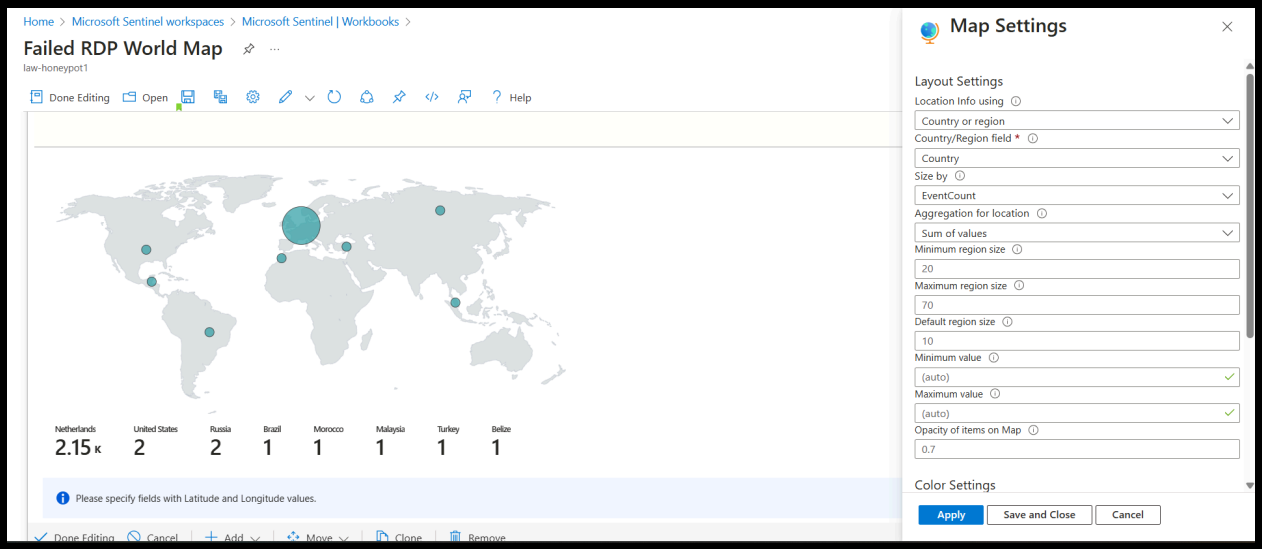


Global Map View, displaying all log activity, and ready to register future event activity (filtering by Latitude):

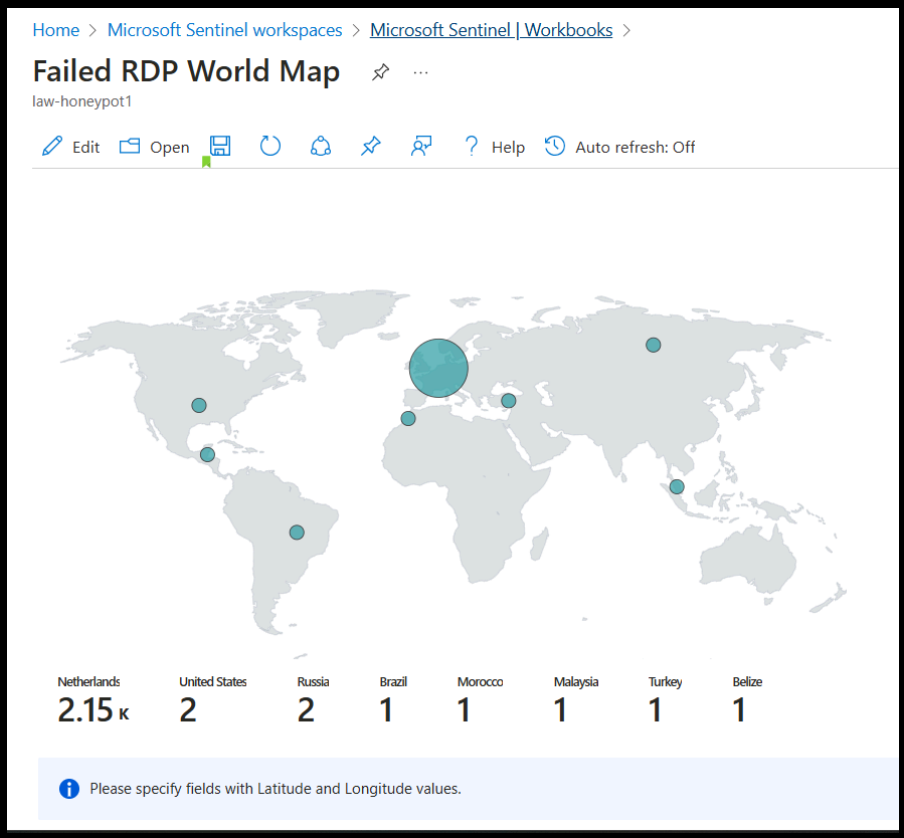


7.3 Country-Based Map

- Global Map settings to filter by Country:



Global Map View, displaying all log activity, and ready to register future event activity (filtering by country):



8.0 Results and Analysis

Observed Patterns:

- The map visualisation highlights clusters of failed logins from specific locations.

- Attack Trends: A large number of failed attempts from certain countries may indicate a brute-force campaign.

The integration of Sentinel with the honeypot VM provided real-time alerts and detailed analysis of attack patterns. The visual map confirmed the presence of multiple failed login attempts originating from various global regions, signaling potential automated attack campaigns. The project highlighted the effectiveness of honeypots in attracting real-world attackers and demonstrated the ability to use Sentinel to detect and analyse real attack data in a production-like environment.

9.0 Challenges and Solutions

Challenge	Solution
Incomplete logs in Event Viewer	Configured advanced logging policies to capture all security events.
No geolocation data for attacks	Integrated IP Geolocation API to enrich logs with country and coordinates.
Sentinel map not displaying data	Adjusted metric settings and verified correct latitude/longitude formatting.

10.0 Project Summary

This project successfully demonstrated how a honeypot can be leveraged to detect and analyse real-world RDP brute-force attacks. By integrating Azure Sentinel, custom KQL queries, and geolocation enrichment, the system effectively tracked and visualized attack trends. The insights gained highlight the importance of proactive threat detection and SOC monitoring in defending against cyber threats.

Future Improvements:

- Automate attack response by integrating Sentinel playbooks.
- Correlate logs across multiple honeypots for enhanced threat intelligence.
- Expand monitoring to additional attack vectors beyond RDP brute force.