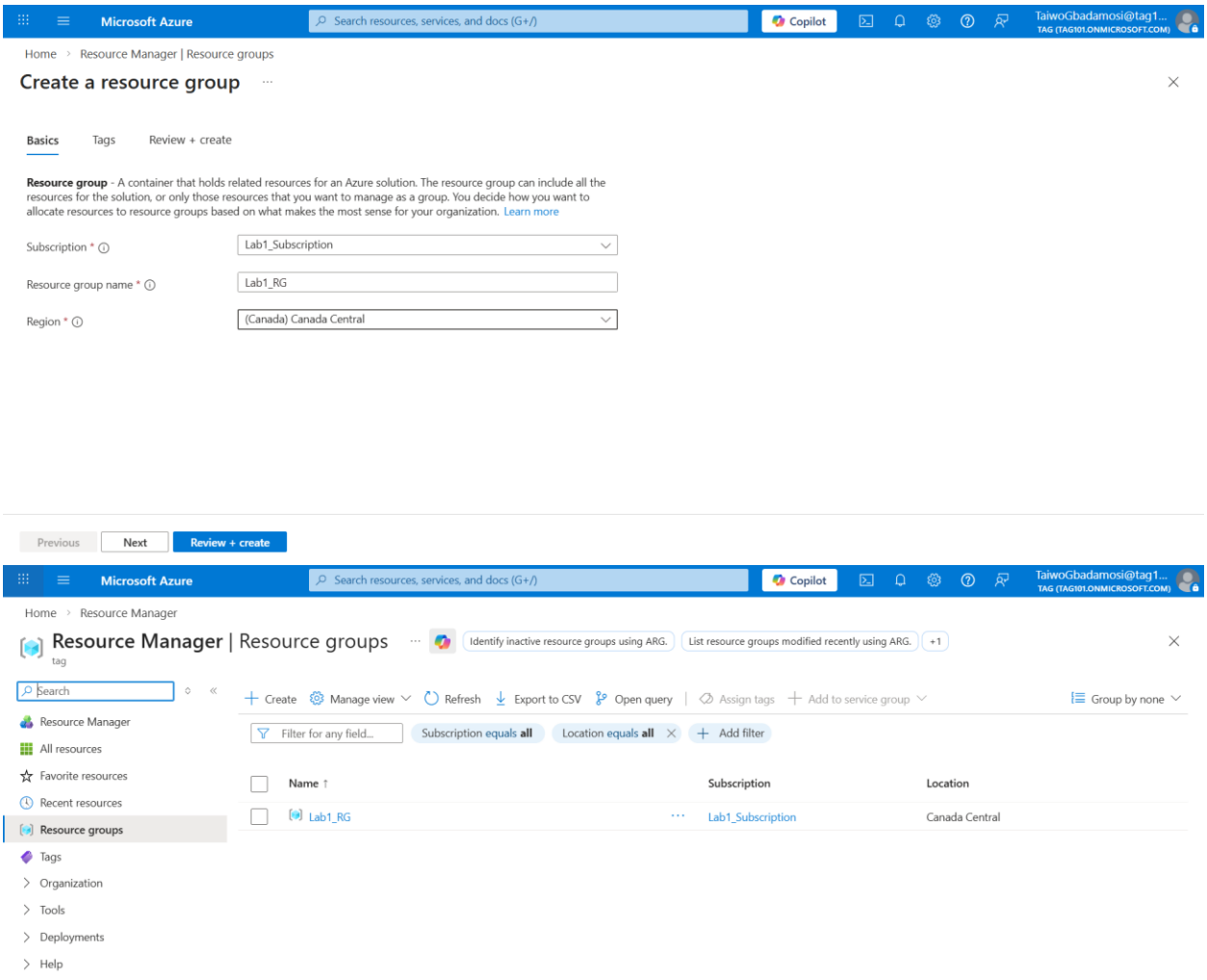
Project Title: Hybrid Cloud SIEM/SOAR Deployment, Log Ingestion & Threat Detection

1. Project Overview & Objectives

The objective of this lab was to simulate a real-world enterprise security environment. The focus was on centralizing log data from disparate sources, analyzing logs, threat hunting, creating high-fidelity alerts for common attack vectors (Brute Force), and automating the initial steps of the Incident Response lifecycle.

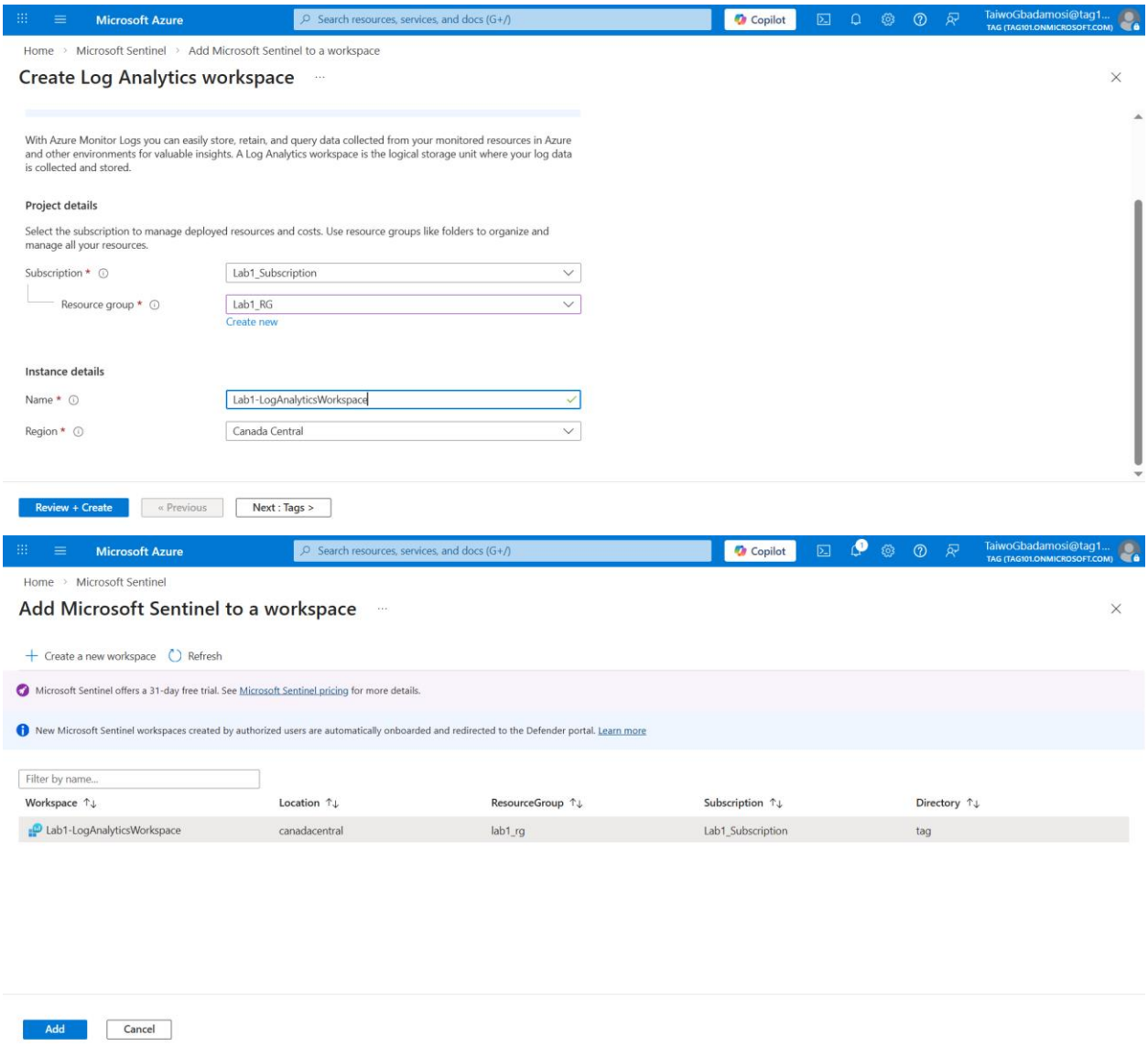
2. Environment Setup & Foundations

Resource Management: Established a centralized Azure Resource Group to manage lifecycle and billing.



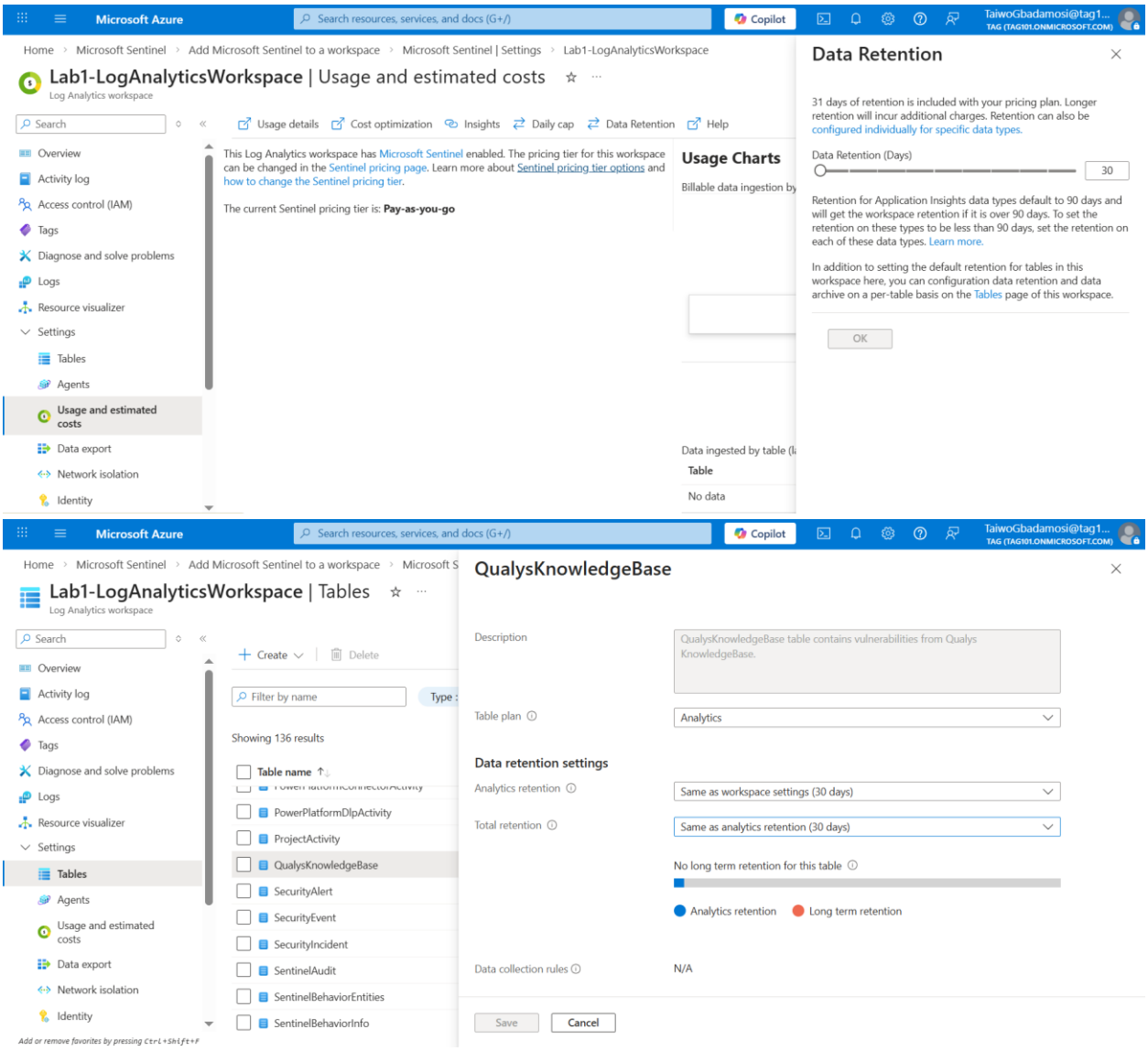
created resource group in Azure for Lab Resources management

Centralized Engine: Provisioned Microsoft Sentinel and Azure Log Analytics Workspace.



Deployed Log Analytics Workspace integrated with Microsoft Sentinel for centralized log ingestion and security telemetry analysis

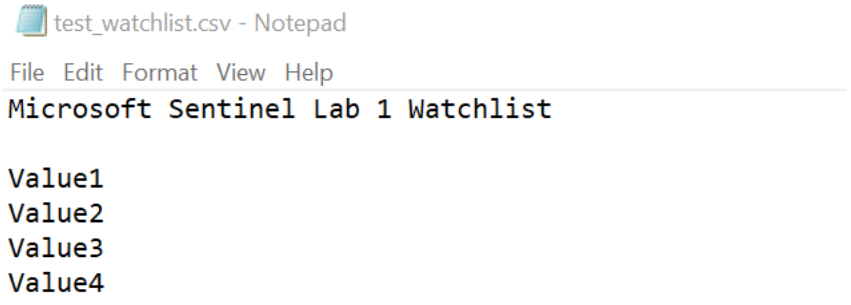
Retention Policy: Configured workspace settings to balance compliance requirements with cost-effectiveness.



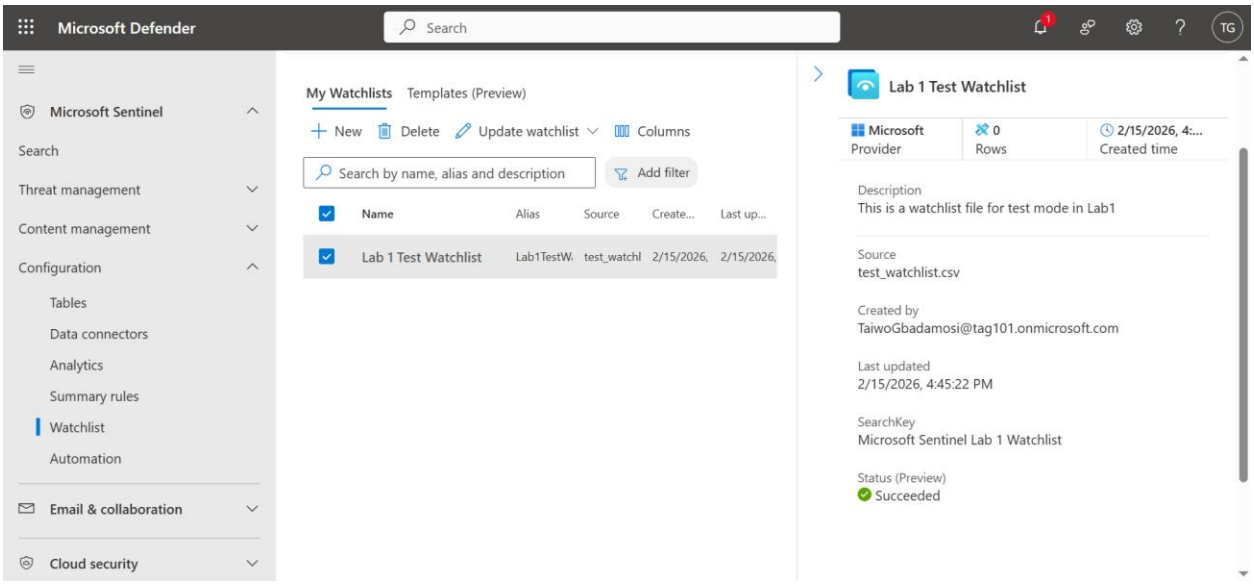
Confiured Data Retention policy in Workspace setting for data lifecycle management optimization

3. Threat Intelligence & Governance

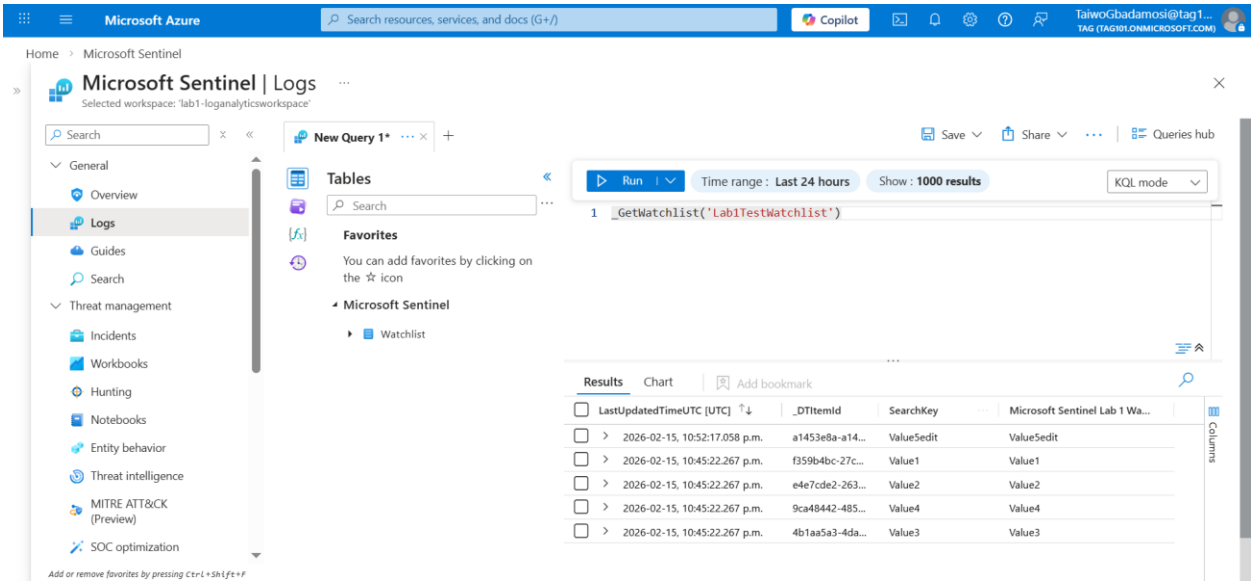
Before ingesting logs, I established a Lab test watchlist that could serve as a known-bad baseline in an actual environment



Notepad generated Test Watchlist

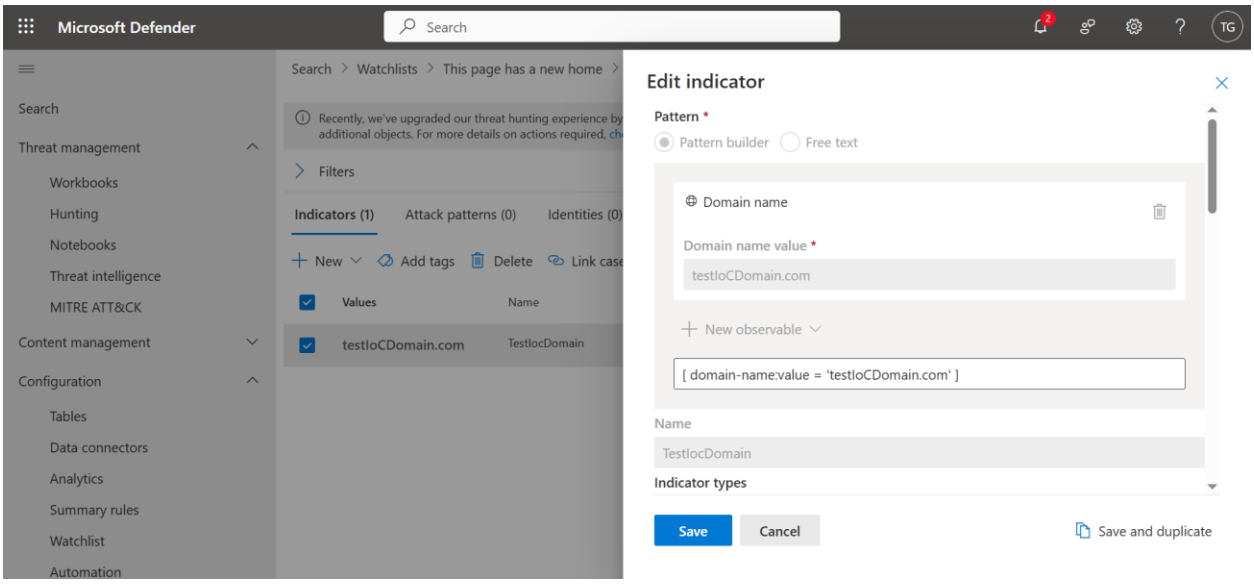


Test watchlist deployed in Azure

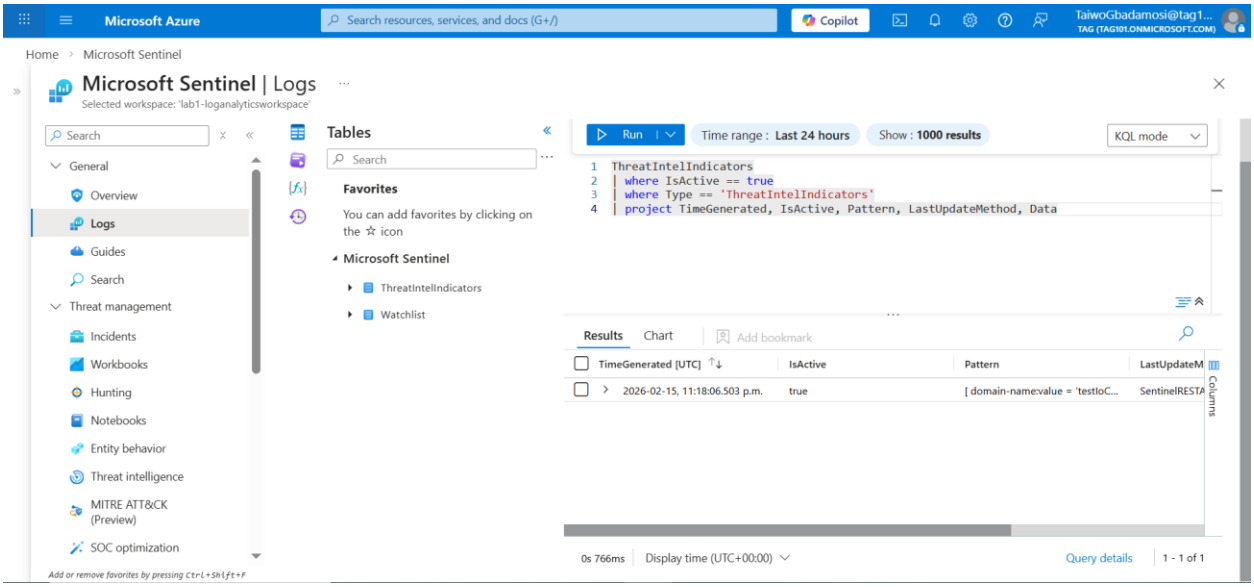


Azure watchlist integrated to Microsoft Sentinel Logs

IoC Management: Manually created a test Threat Indicator object which could ensure the SIEM is primed to detect known malicious fingerprints in an actual environment.



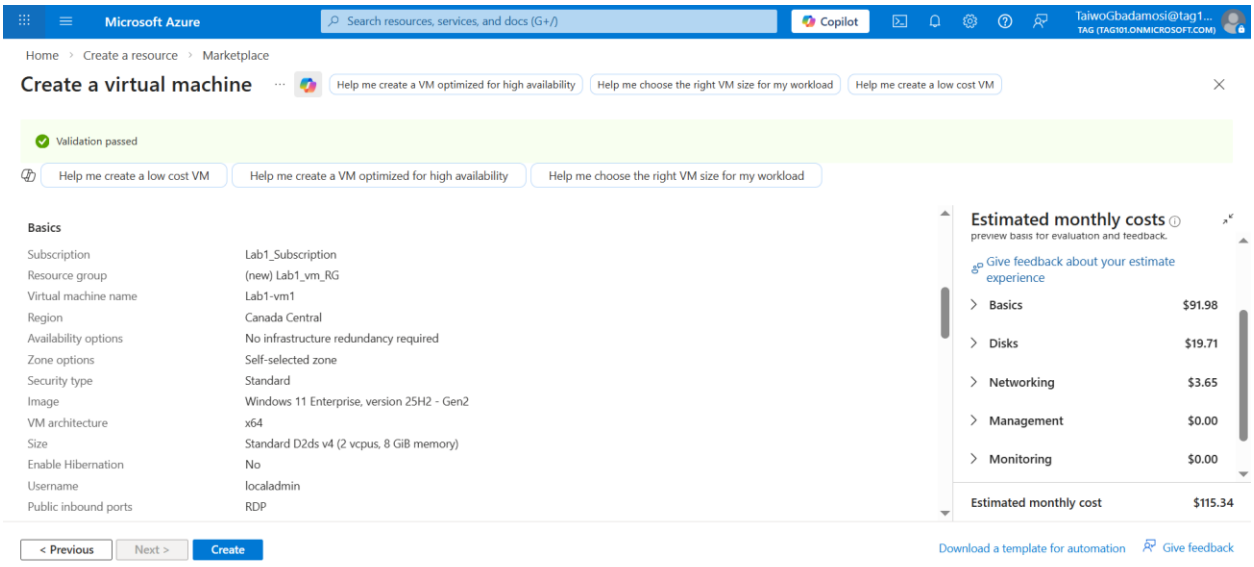
Deploying a test Indicator object



Test Indicator object enriched in Microsoft Sentinel

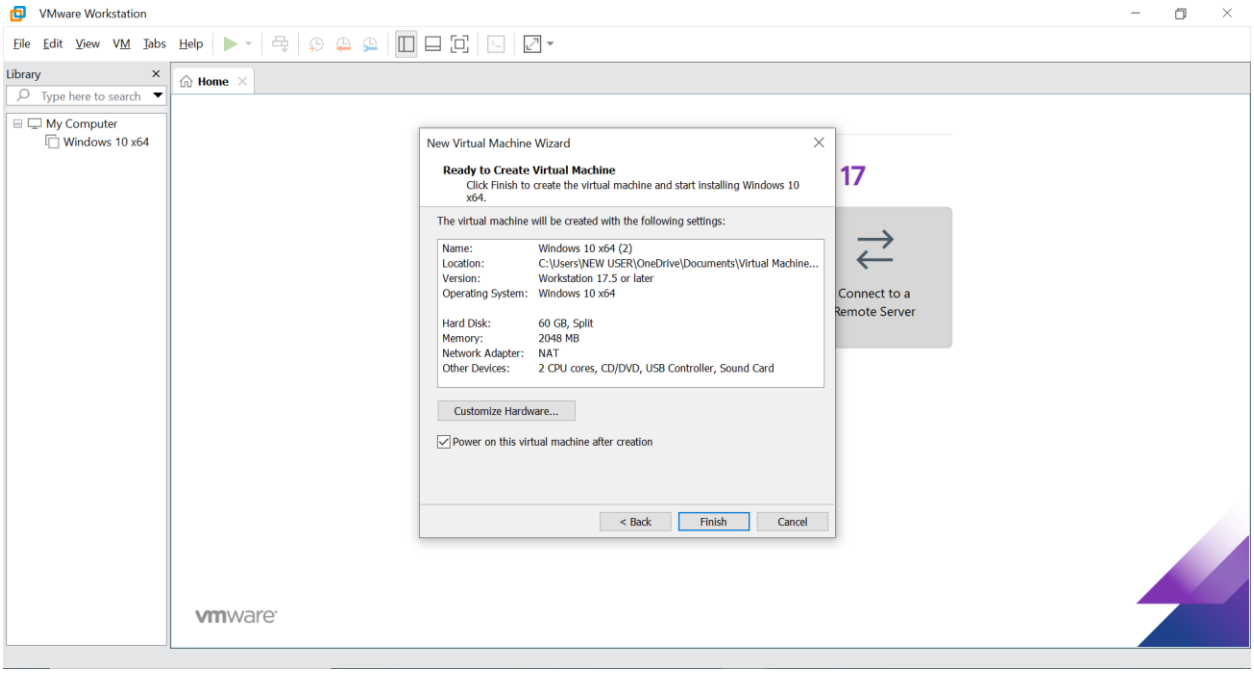
4. Hybrid Log Ingestion (The "Azure Arc" Implementation)

This phase focused on the Security Reach of the SIEM.
Azure VM: Standard deployment of a Windows VM in Azure.

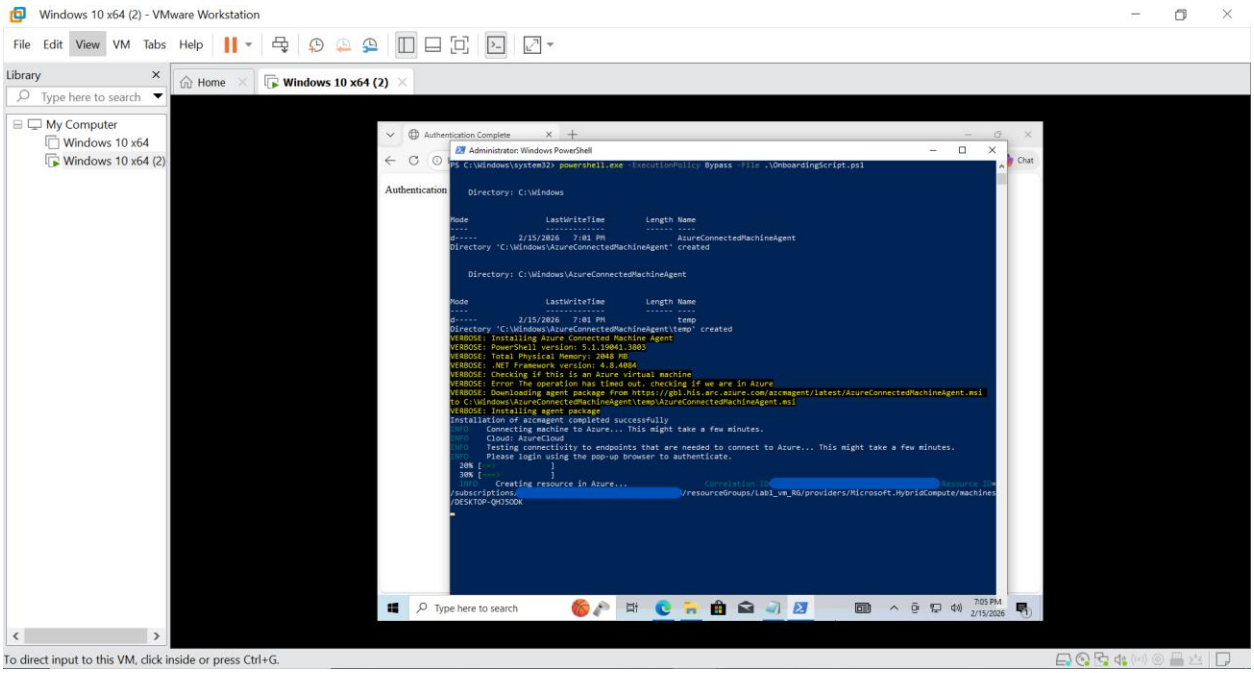


Deployment of Windows 11 vm in Azure

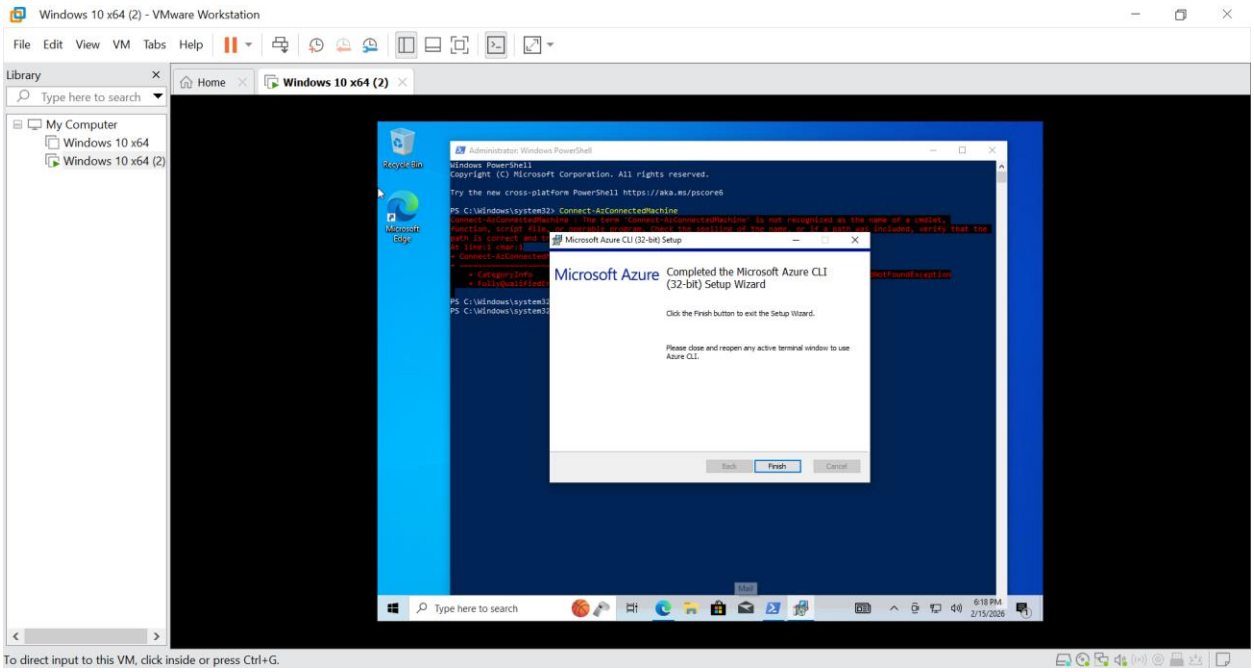
VMware Integration: Deployed a local windows 10 machine and generated an onboarding script via Azure Arc to connect local machine to Azure.



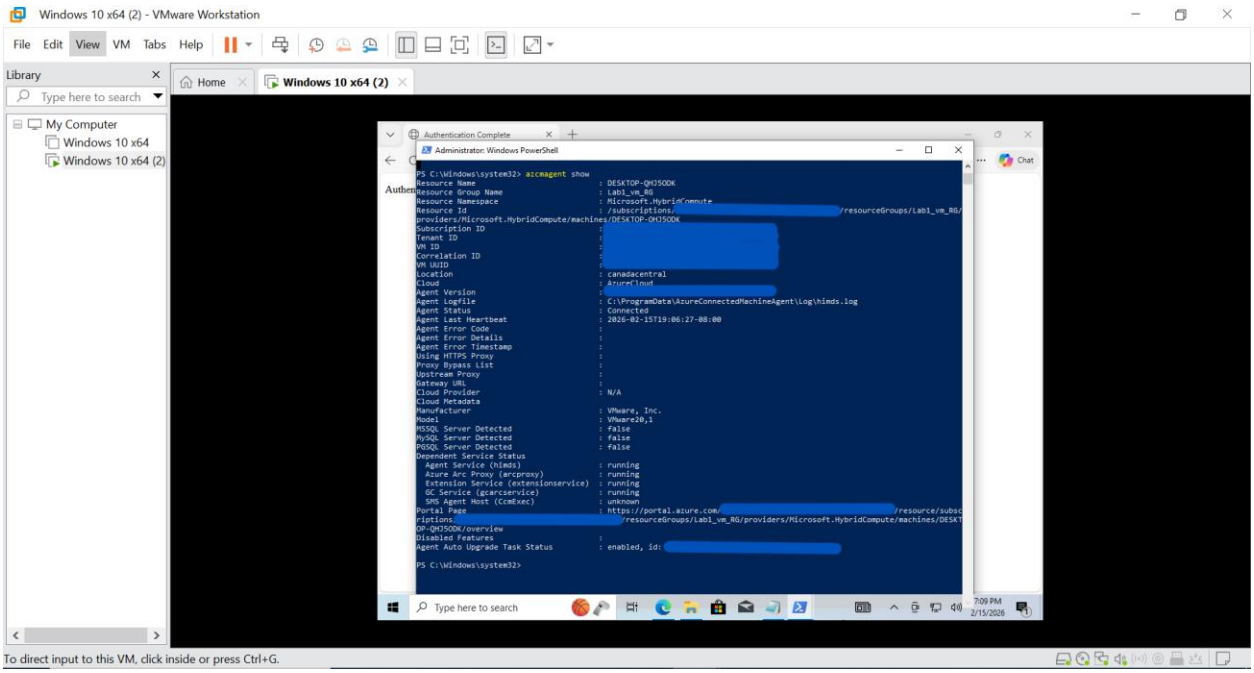
Deployment of Windows 10 vm in vmware



Onboarding Windows vm to azure arc 1

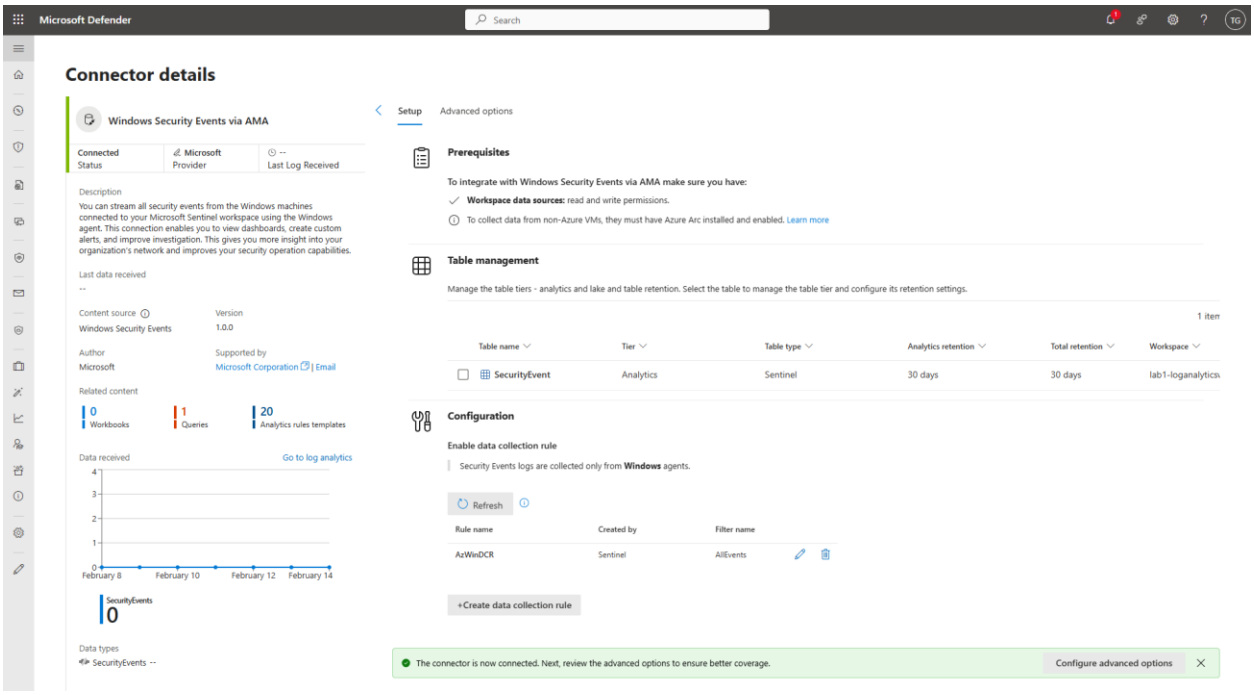


Onboarding Windows vm to azure arc 2

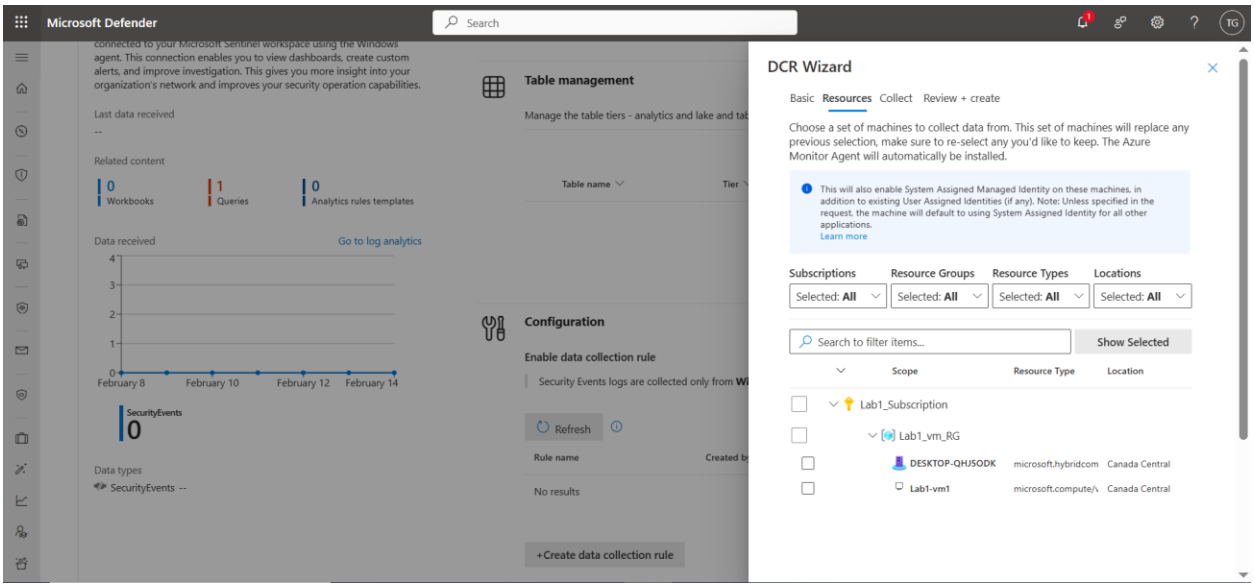


'Azcmagent show' generating azure arc onboarded info

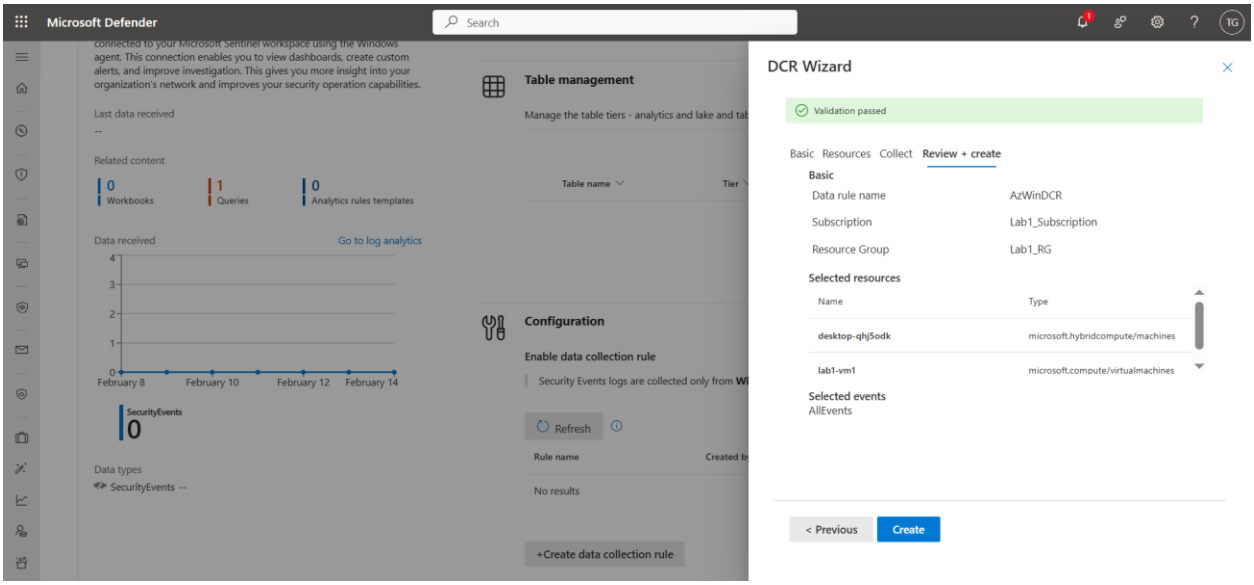
AMA & DCR: I utilized the Azure Monitor Agent (AMA) and created a Data Collection Rule (AzWinDCR). This ensured that Windows Security Events were streamed directly to the Sentinel Workspace.



Installing and configuring Windows Security Events data collector via Azure Monitoring Agent



Configuring data collection rule for specific resource group (containing both Azure and non-Azure machines)



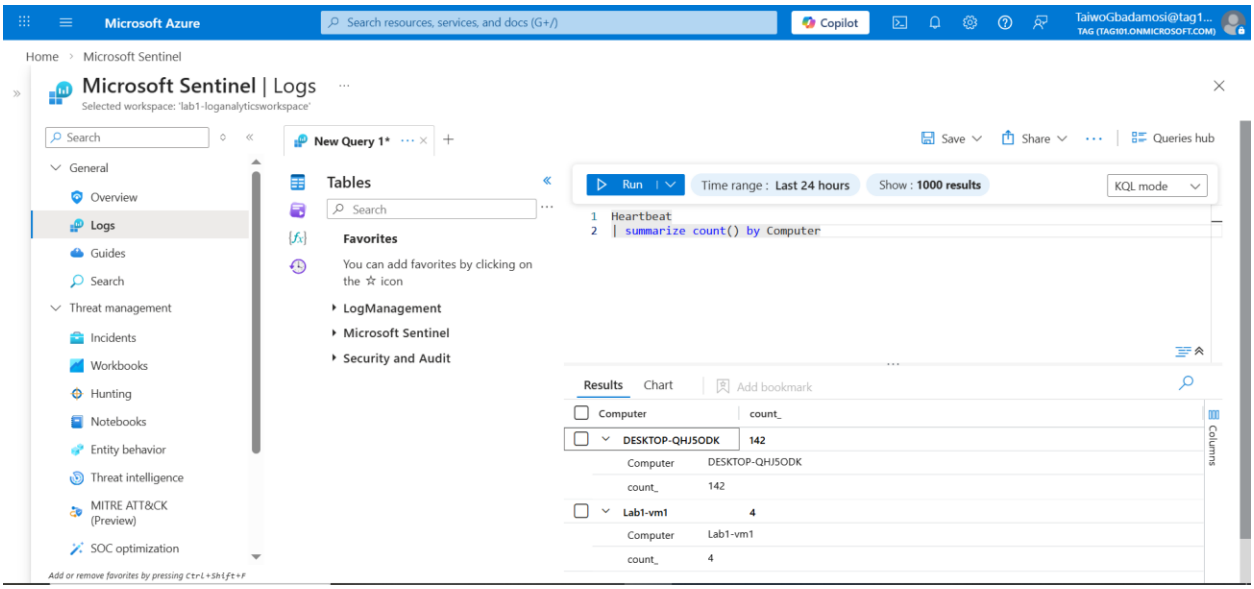
Creating Data collection rule "AzWinDCR" to ingest specific Windows security event logs

5. Detection & KQL Validation

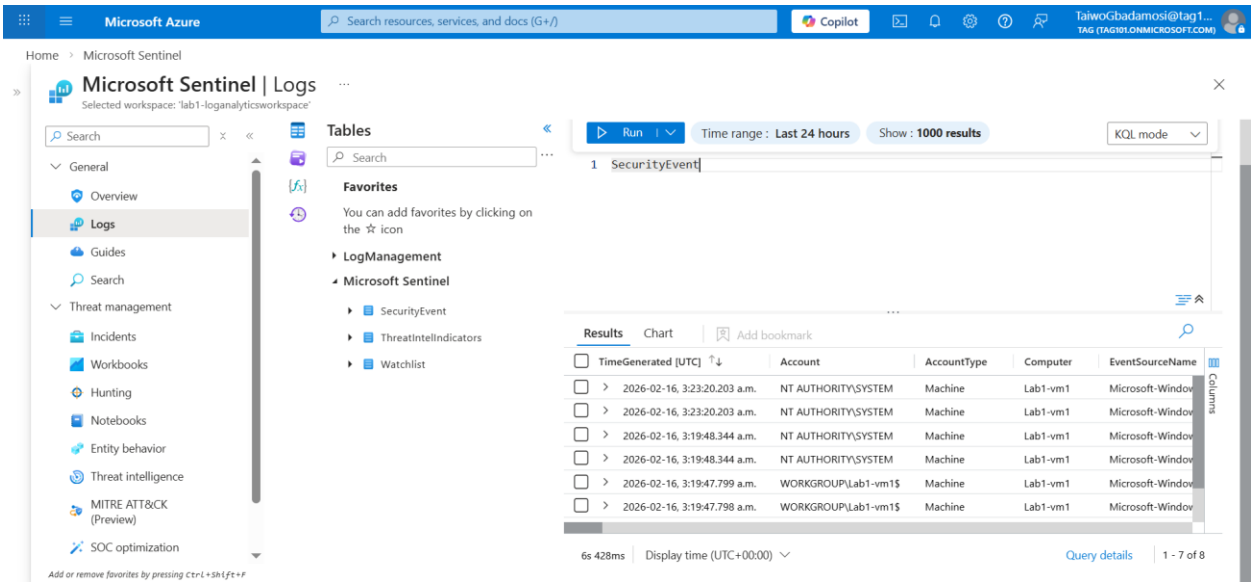
To ensure logs were flowing correctly, I executed heartbeats and sample queries.

Verification Query: Heartbeat | summarize Count() by Computer

Event Validation: Confirmed that Windows Event ID 4625 (Failed Logons) was populating the SecurityEvent table.



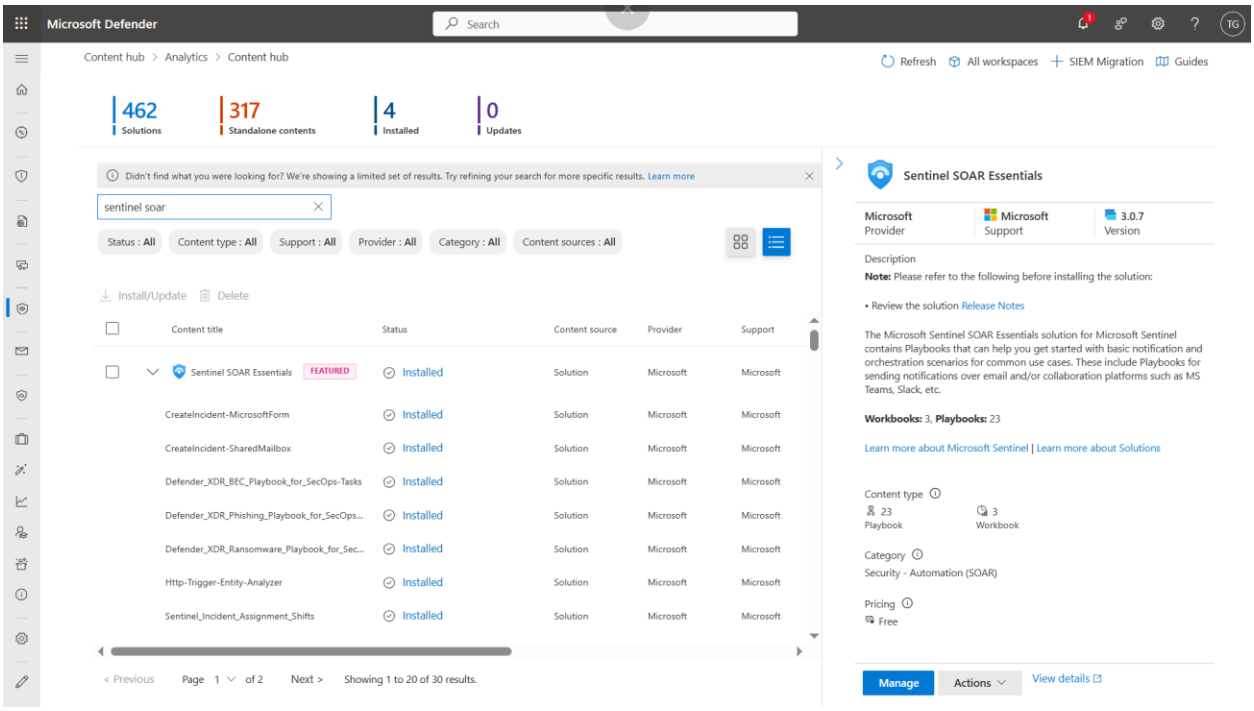
Windows security event logs integrated to Microsoft Sentinel showing heartbeat of both devices



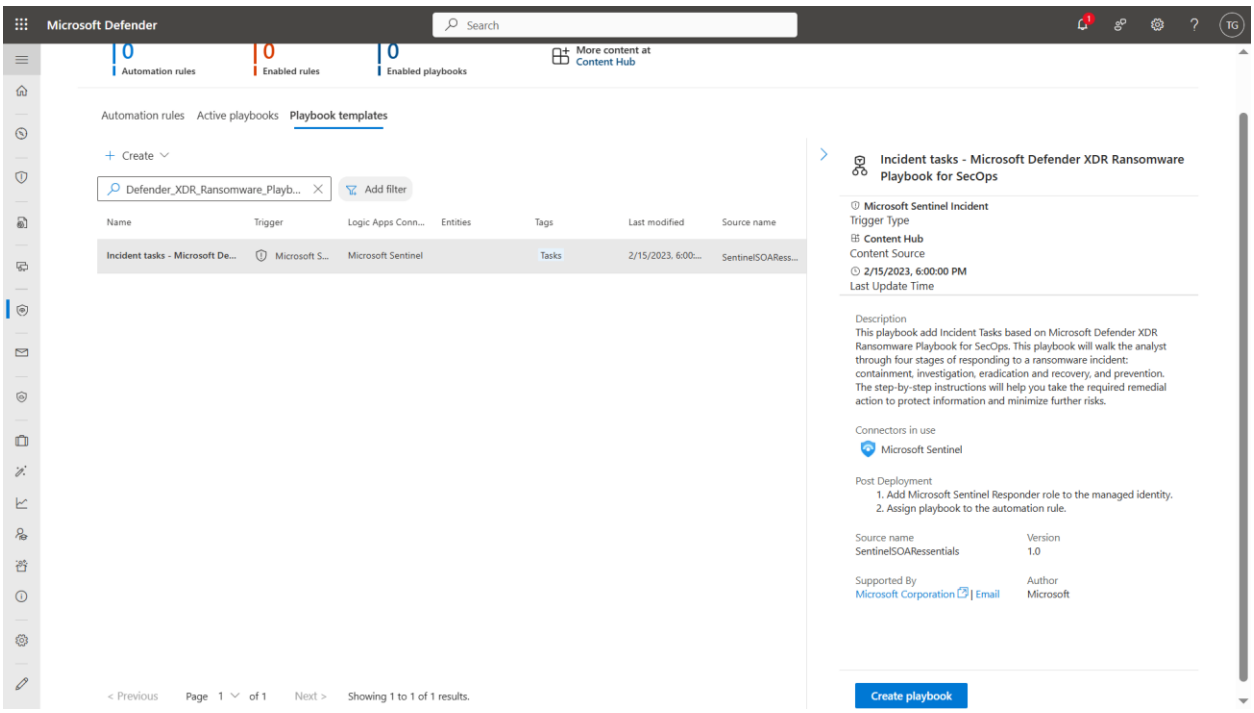
Query 'SecurityEvent' log integrated from Windows machines

6. SOAR: Automation & Response Orchestration

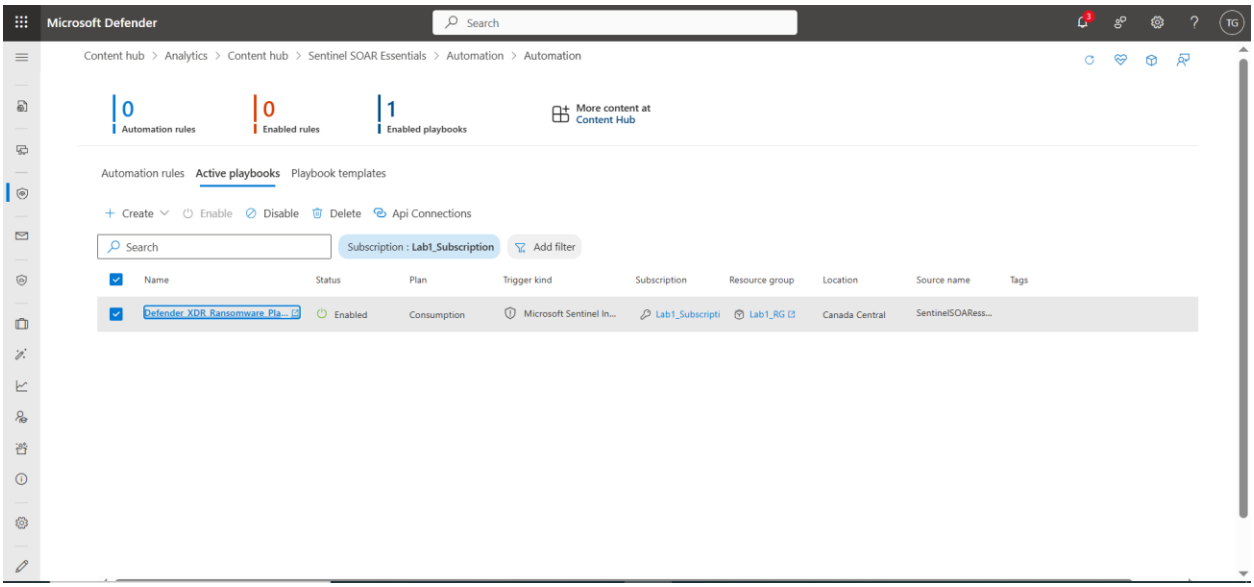
I implemented SOAR (Security Orchestration, Automation, and Response) to reduce Mean Time to Respond (MTTR).



Configuring Sentinel SOAR Essentials data connector to utilize SOAR templates

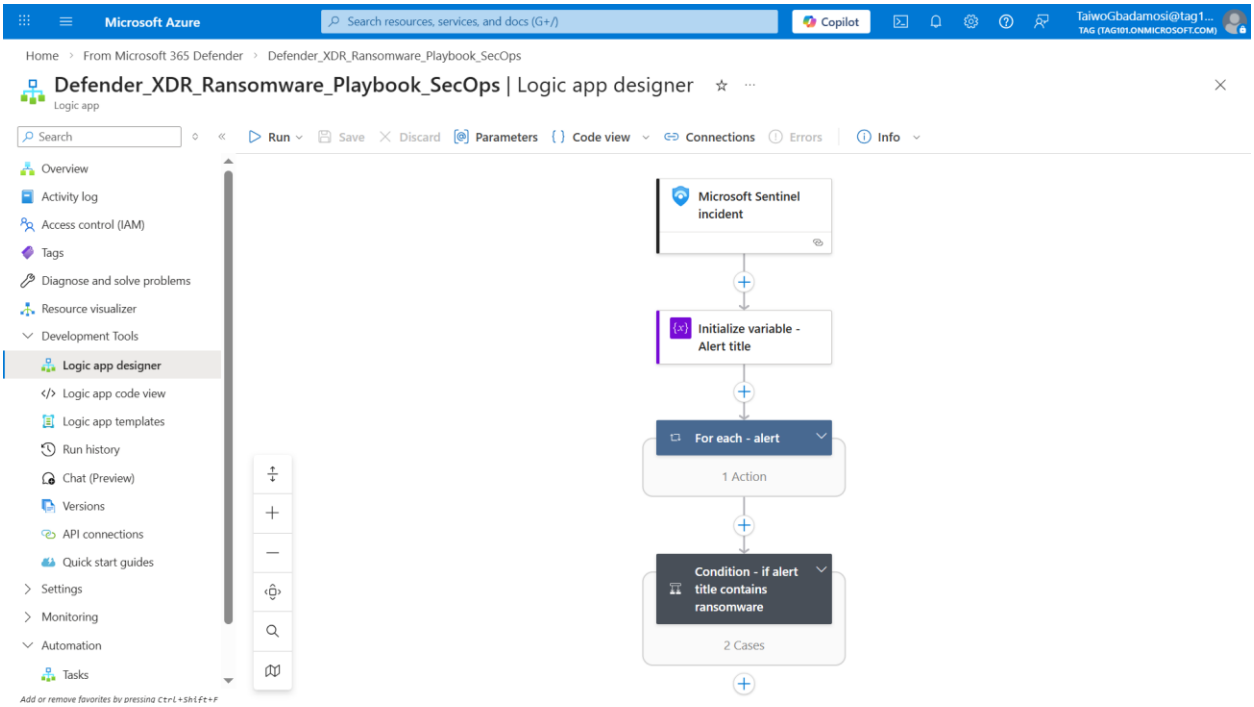


Configured Test Playbook utlilzing SOAR template



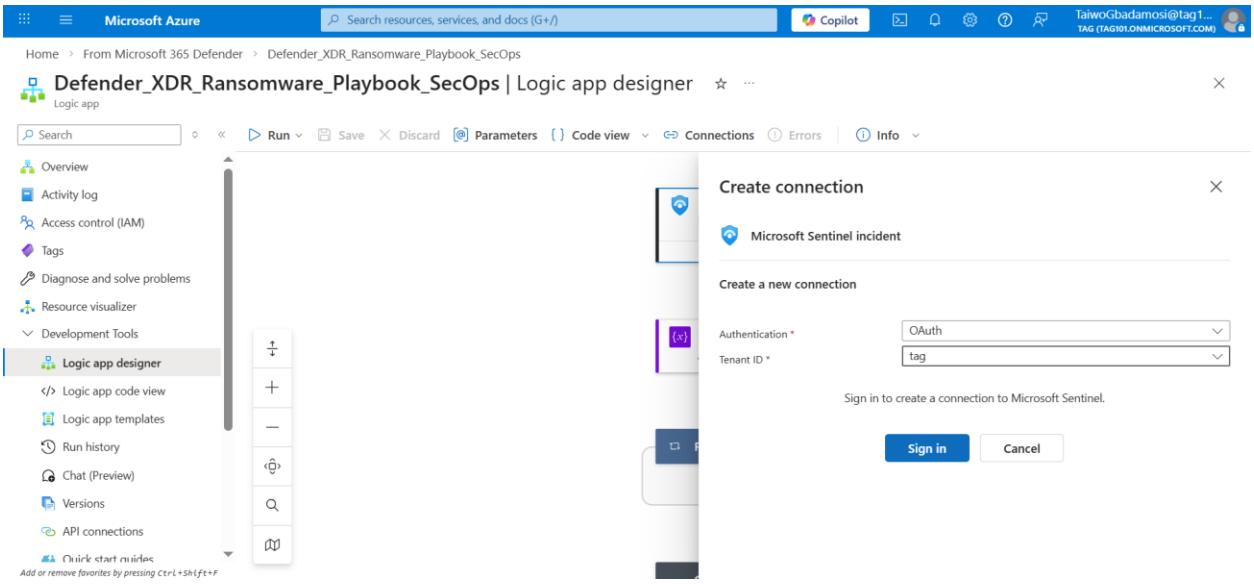
Centralized repository of active SOAR playbooks for automated incident response

Playbook Development: Utilized the Sentinel SOAR Essentials template to create a Logic App.

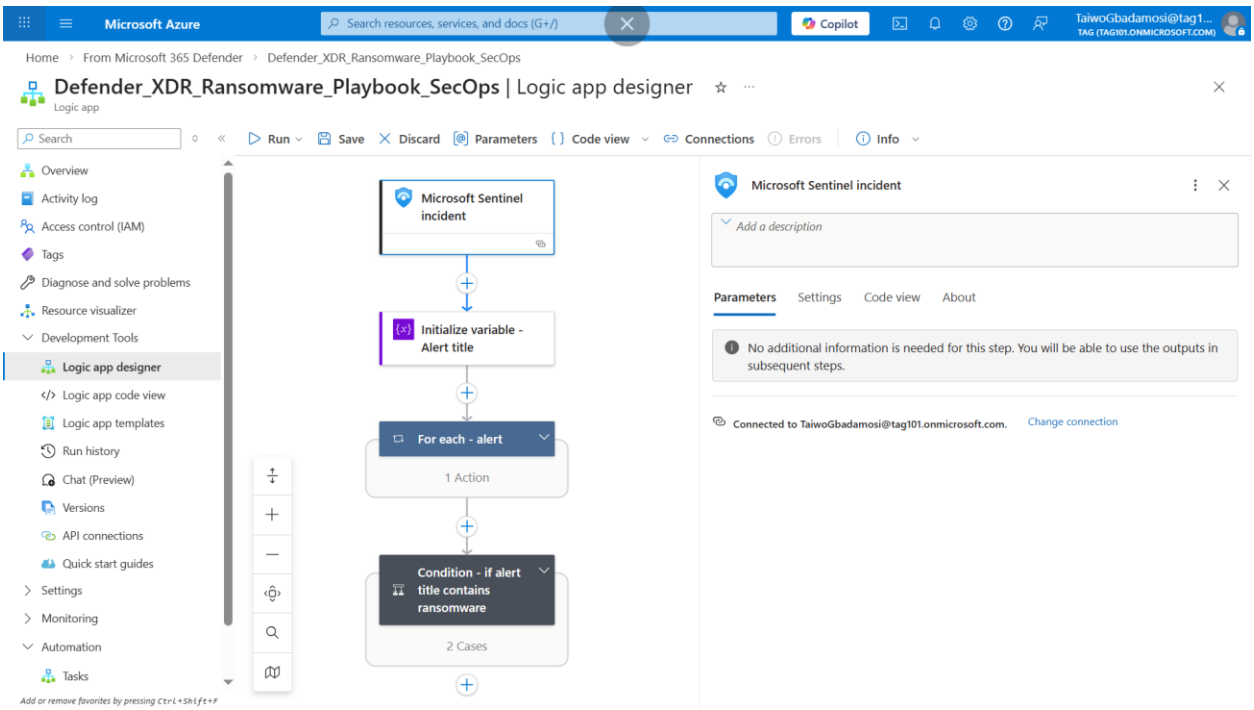


Logic App designer showing configuration of Test playbook Logic flow

Identity Integration: Authenticated the playbook with my Entra ID tenant to allow for identity-based responses.

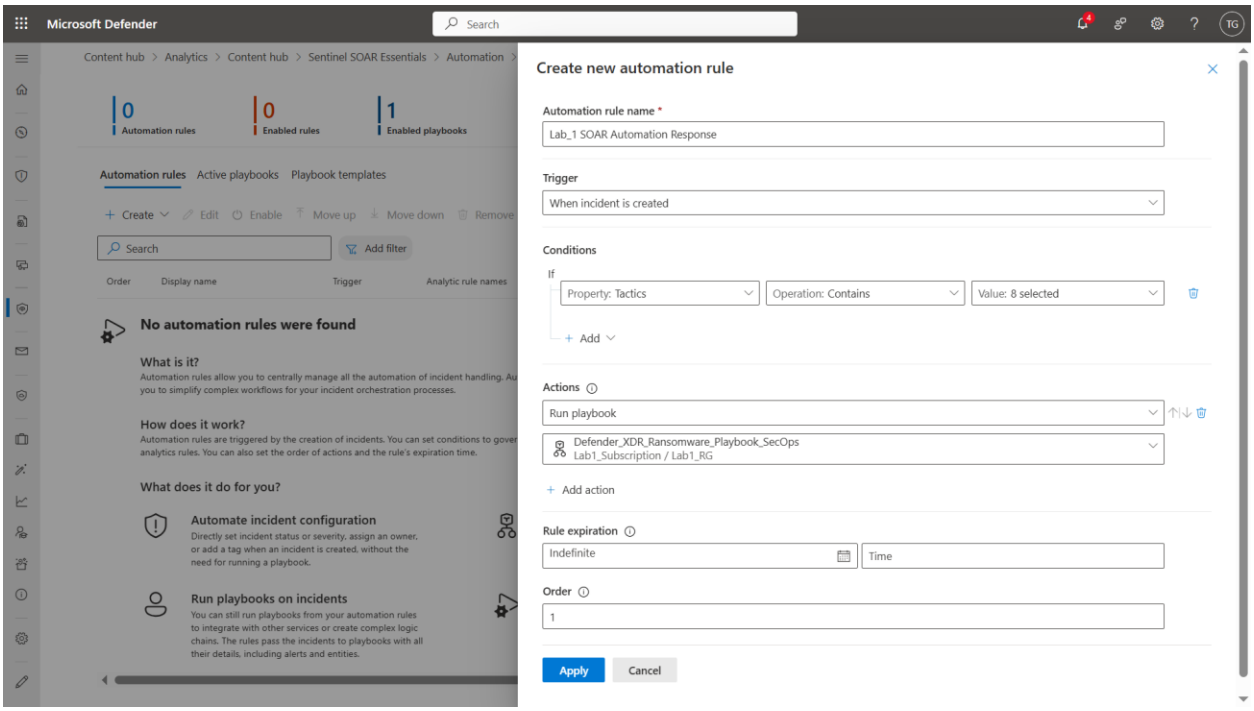


Established API connections and configured authentication between the SOAR playbook and my Microsoft Entra ID tenant to facilitate automated identity actions.

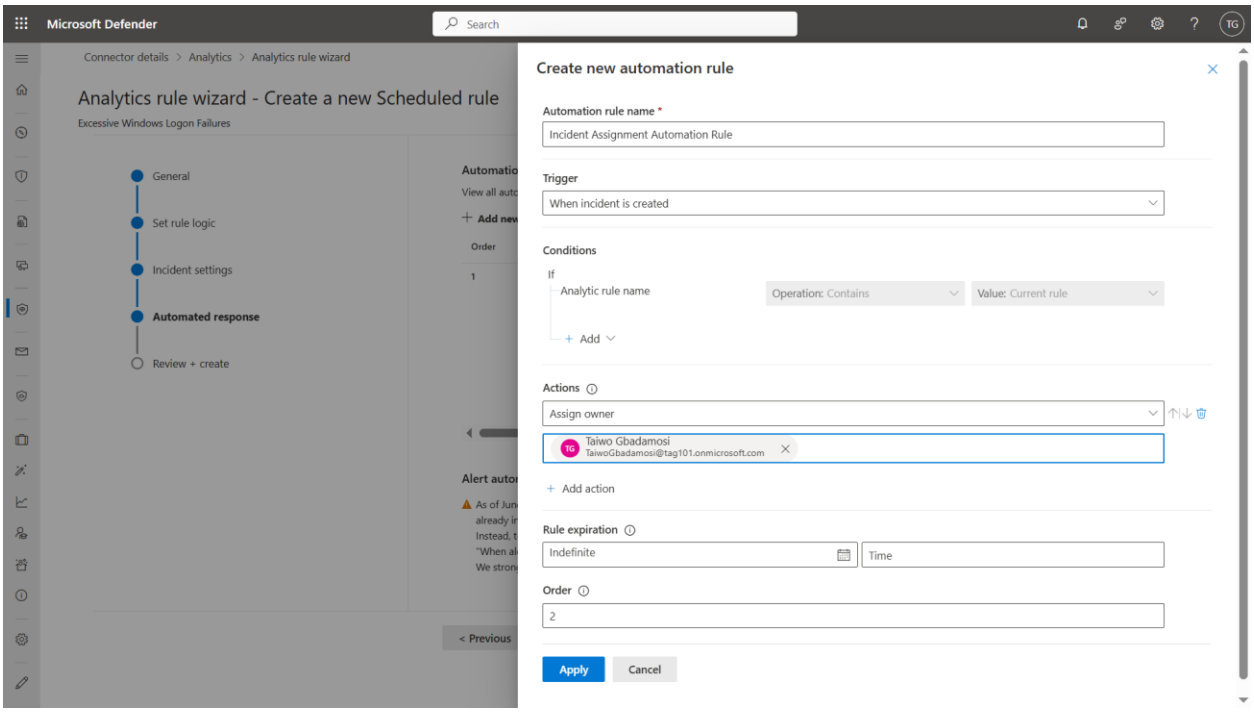


Playbook authenticated to my Entra id tenant account

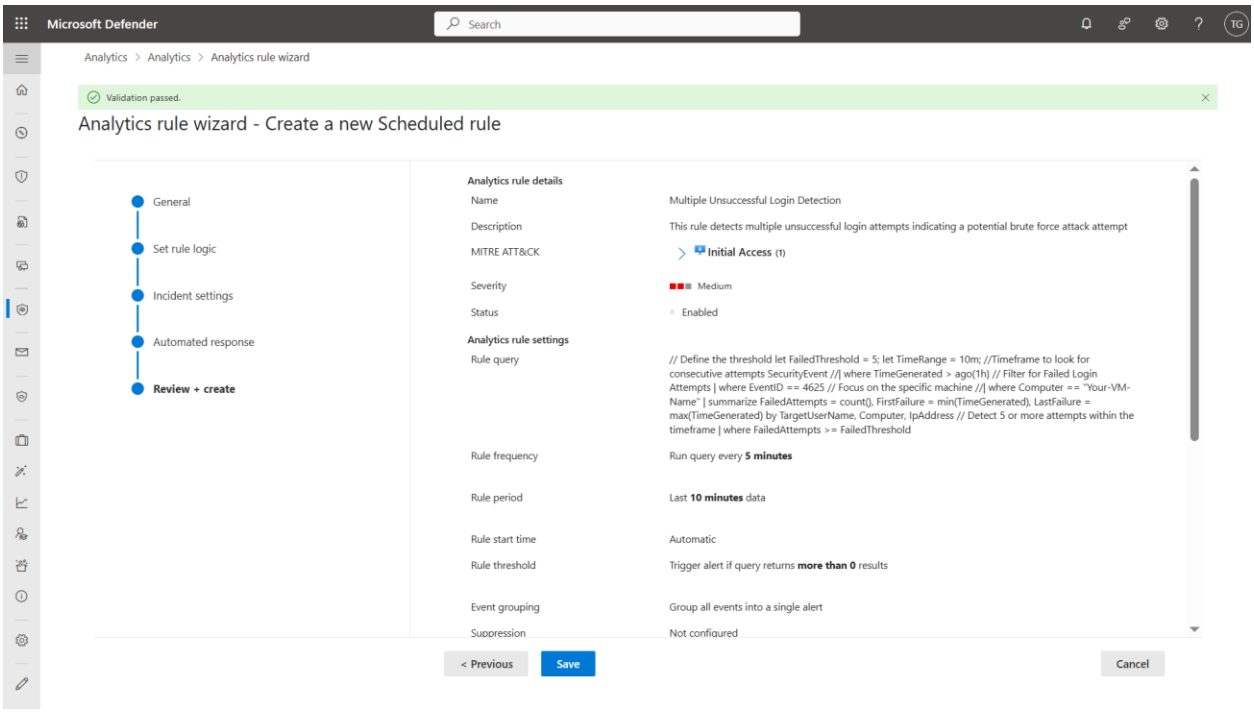
Automation Rule: Created a trigger that automatically calls the playbook and assigns specific alerts to the Security Engineer (myself) upon creation.



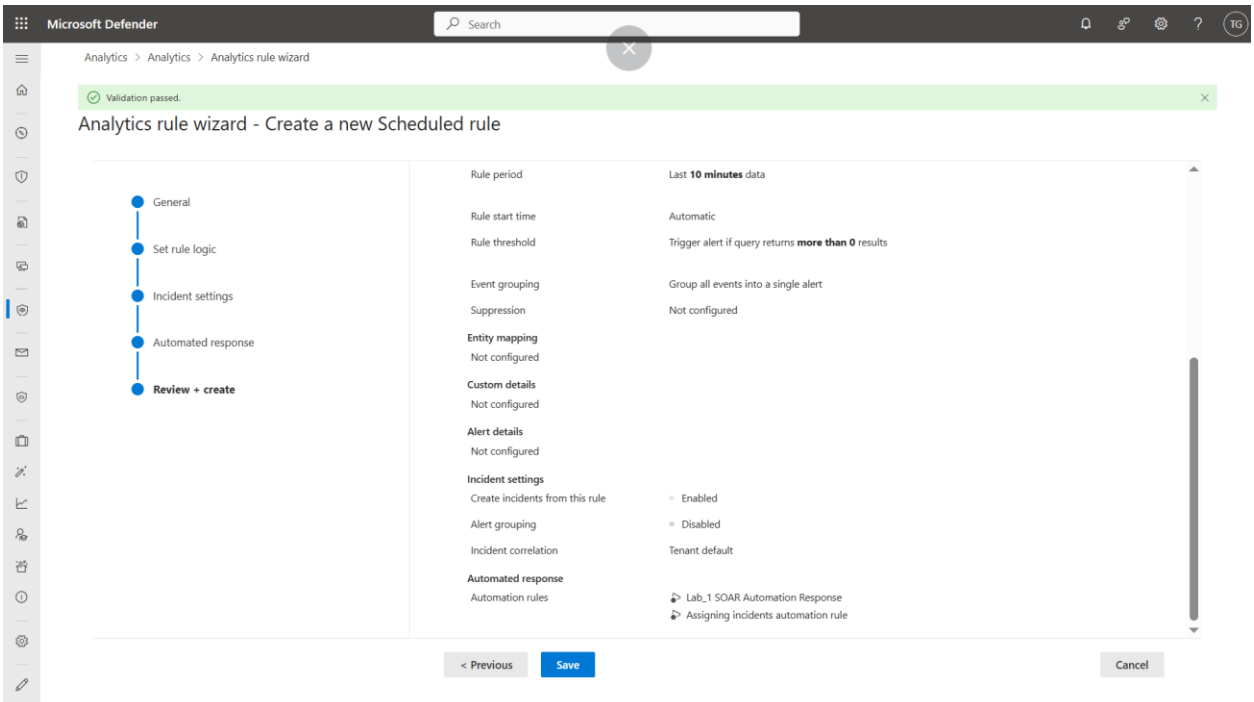
Configured a Sentinel Automation Rule to trigger the SOAR playbook, enabling an automated response whenever specific alerts are generated.



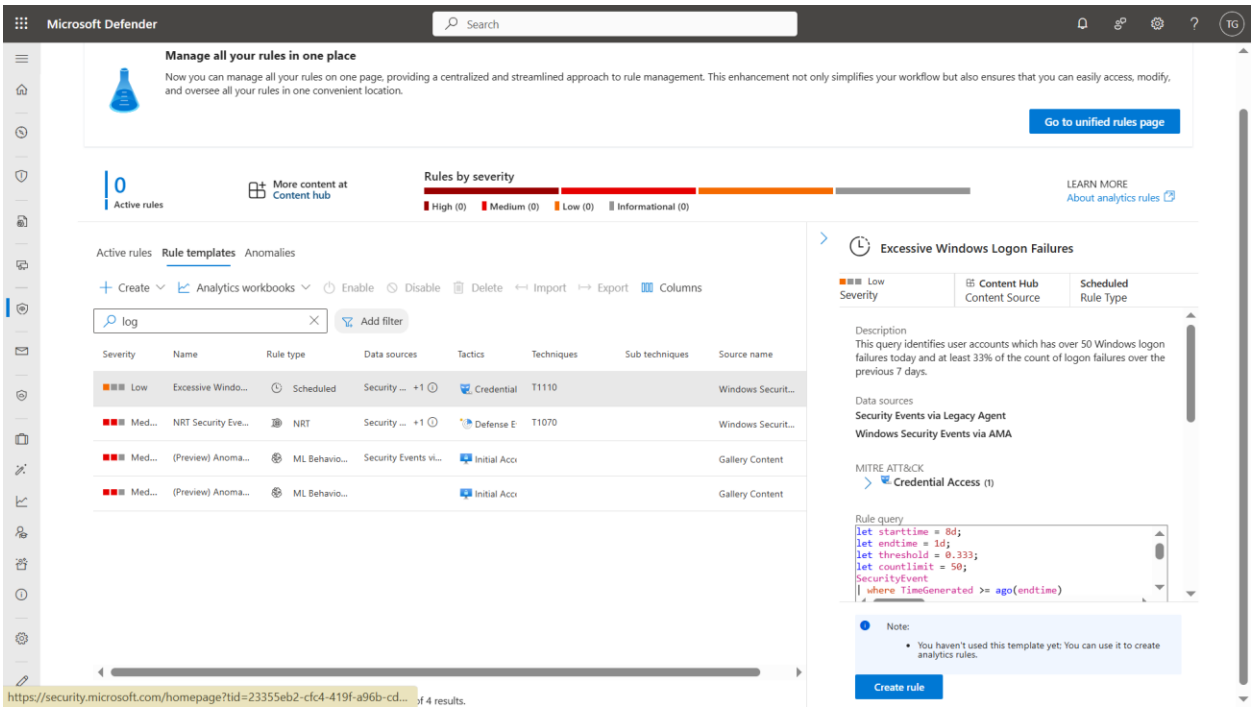
Configured a Sentinel Automation Rule to automatically assign incidents to myself whenever specific alerts are generated.



Configuration of Analytics Rule with custom KQL for automated threat detection and incident creation 1



Configuration of Analytics Rule with custom KQL for automated threat detection and incident creation 2

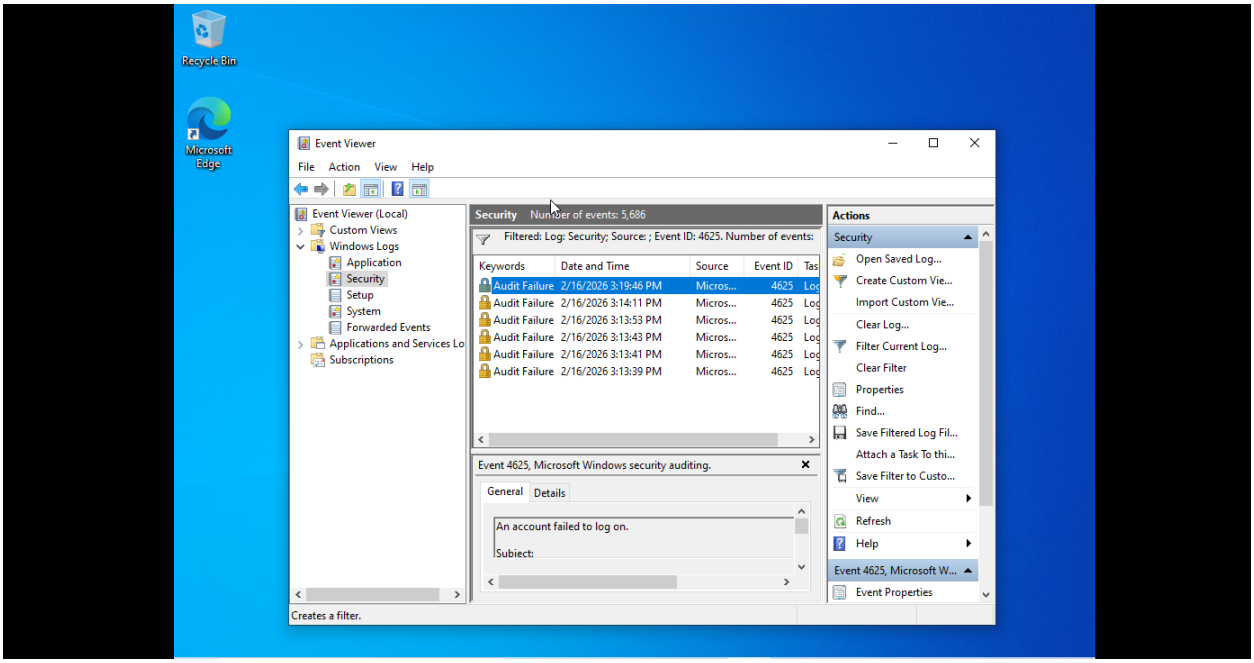


Comprehensive view of deployed Analytics Rules and detection logic

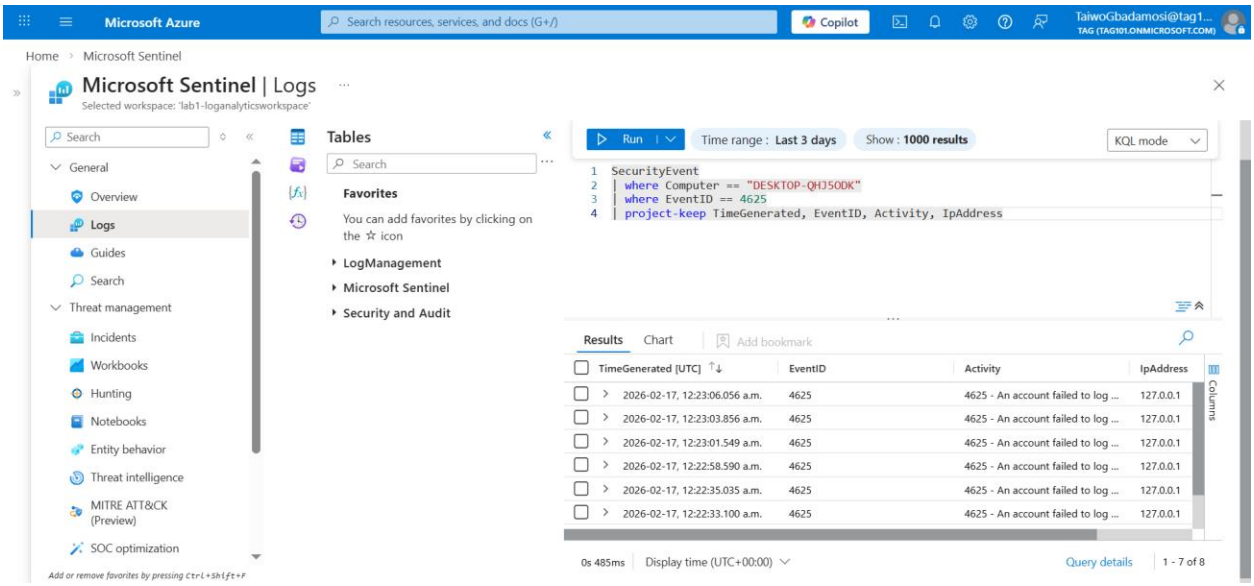
7. Attack Simulation & Incident Handling

To test the End-to-End flow, I simulated a Brute Force Attack.

The Attack: Conducted multiple failed login attempts on the VMware-hosted Windows 10 machine.



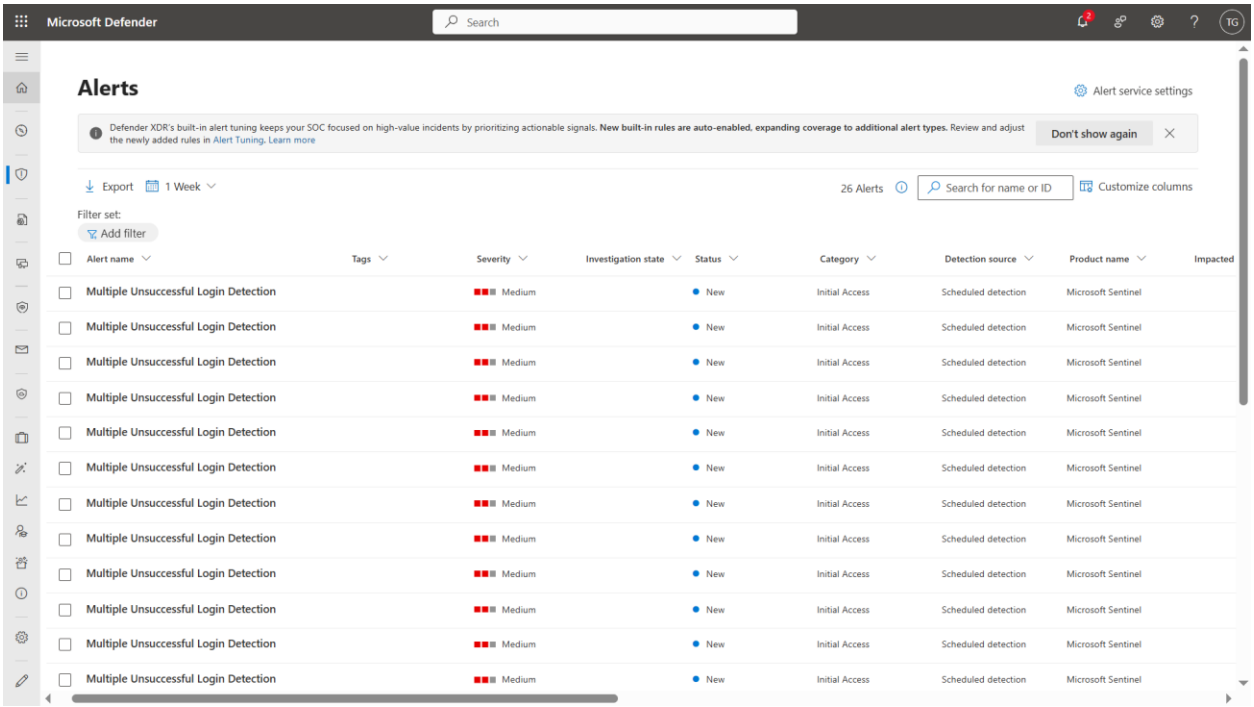
Local Windows Event Viewer showing Event ID 4625 (Audit Failure) during the simulated attack



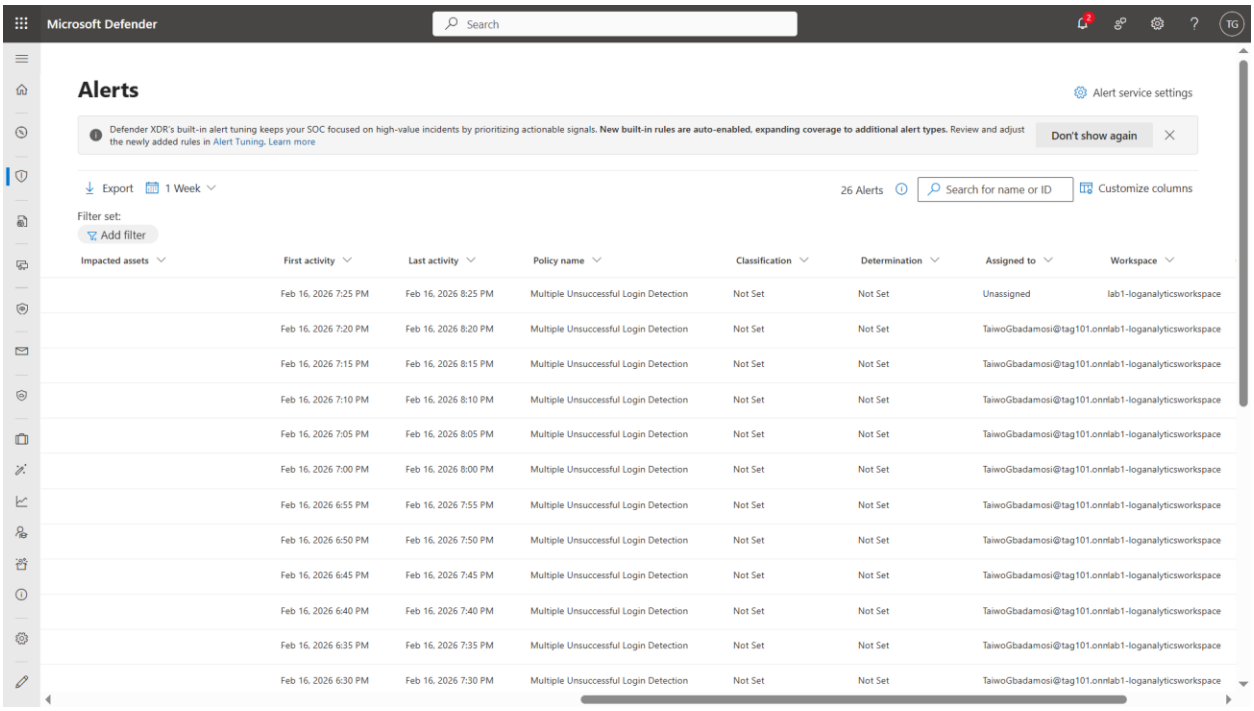
Microsoft Sentinel logs query showing EventID 4625 (unsuccessful logon attempt) after simulated attack

Detection: The Scheduled Analytics Rule identified the pattern of failures.

Alerting: The Scheduled Analytics rule configured to generate an incident upon threat detection in Sentinel generated an incident, triggered the automation rule, and assigned the case to me.

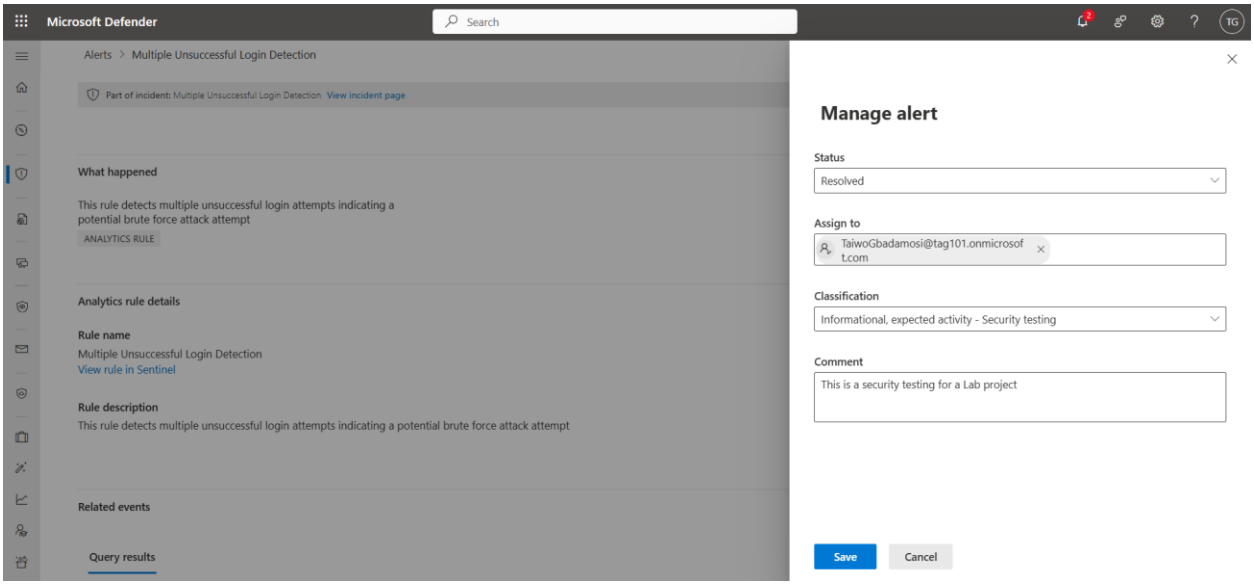


High-severity alerts generated by the Analytics Rule following a detected brute-force attempt

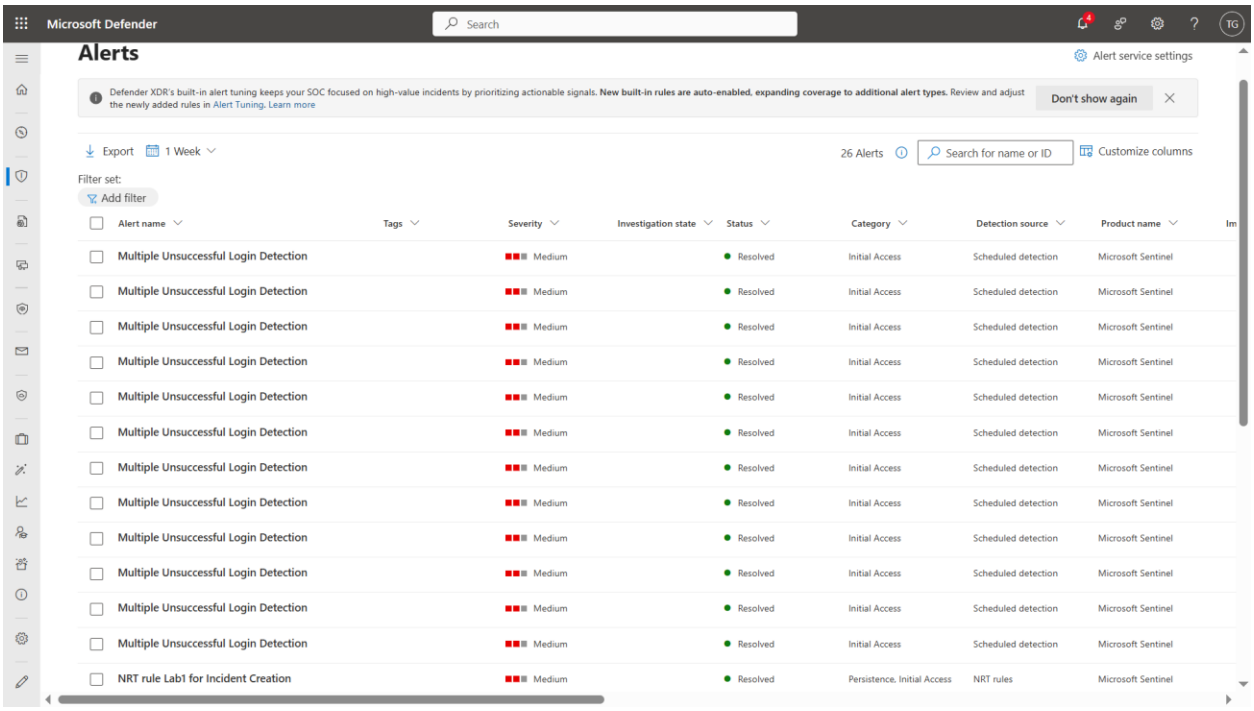


High-severity alerts generated by the Analytics Rule following a detected brute-force attempt automatically assigned to me

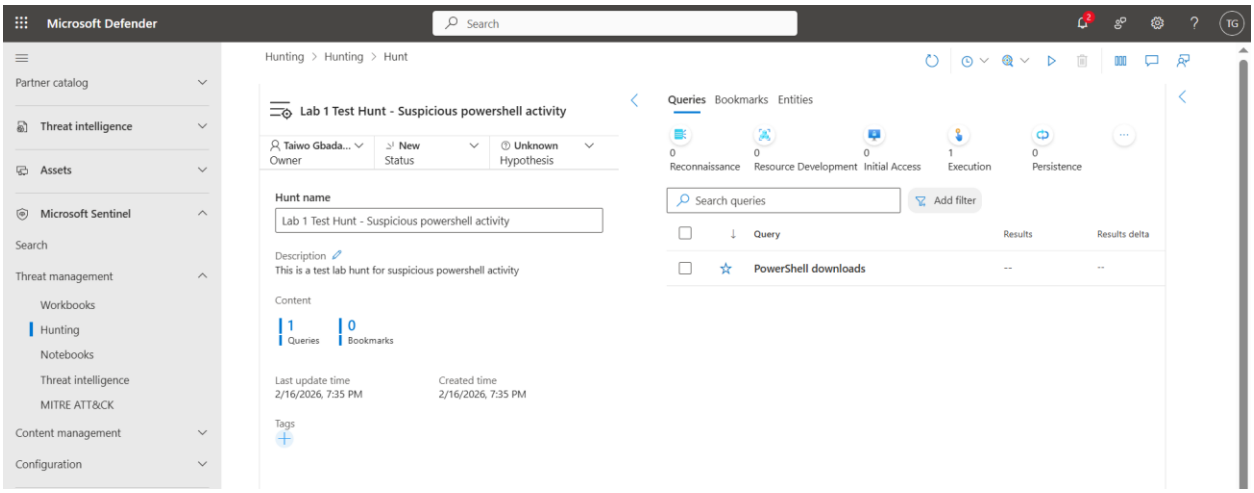
The Response: I reviewed the logs, validated the source IP, classified the incident as "Security Testing," and resolved the ticket.



Incident remediation and closure: Classifying the event as 'Security Testing' and updating the status to 'Resolved' following a successful analysis



Incident detail page showing the 'Resolved' status and classification comments



Implementation of a PowerShell-specific hunting hypothesis within the Microsoft Sentinel Hunting blade

8. Lessons Learned & Outcomes

Agent Evolution: Transitioning to AMA and DCR demonstrated the granular control Azure now provides over what logs are ingested, significantly reducing noise and costs.

Azure Arc Utility: I learned that Azure Arc effectively treats on-premise servers as native Azure objects, simplifying security policy application.

Automation Logic: Configuring the SOAR playbook reinforced the importance of API permissions and Entra ID authentication in the response chain.

Project Outcome: This lab reinforced my ability to manage the full Logs lifecycle from generation on a local VM to ingestion, detection, and automated resolution in the cloud.