Microsoft Azure Honeypot & Threat Monitoring Project

Report prepared by Leanne Goldsmith

Objective

To set up a secure honeypot environment in Microsoft Azure to attract and monitor potential threat actors, collect security logs, visualise attacks geographically, and create actionable alerts for incident response.

Tools Used

- **Microsoft Azure (Resource Groups, Virtual Networks, Virtual Machines):** Organised resources, created isolated networks, deployed virtual machines for testing environments.
- **Azure Network Security Groups (NSG):** Controlled network traffic, restricted access, secured virtual machines from threats.
- **Windows 10 Pro (22H2) virtual machine:** Provided Windows environment for testing, logging, and security monitoring.
- **Azure Monitor & Windows Agent (AMA):** Collected metrics, logs, and performance data from virtual machines.
- **Microsoft Sentinel (SIEM):** Centralised threat detection, correlation, and incident response for security events.
- **Log Analytics Workspace & KQL queries:** Stored logs, queried data for investigation and incident analysis.
- **Visualisation tools in Sentinel (Workbooks, Watchlists):** Displayed security data, tracked threats, and highlighted key indicators.

Key Steps

The project was executed through the following key steps:

1. **Resource Group Creation:**
   - Created in West Europe due to compliance restrictions.
   - Acts as a container for all related resources.
2. **Virtual Network & Virtual Machine Setup:**
   - VM configured as a honeypot: named “Admin-PC-Leanne” to attract attackers.
   - Windows 10 Pro installed, RDP port 3389 initially enabled for testing.

- Firewall disabled on VM to allow inbound connections.

3. **Network Security Group (NSG) Configuration:**
   - Created inbound rules to allow all traffic, simulating a vulnerable target.
   - Prioritised rules to ensure honeypot visibility.
4. **VM Access & Testing:**
   - Connected via Windows Remote Desktop app.
   - Tested connectivity over public internet (ping successful).
   - Conducted failed login attempts to generate security logs.
5. **Log Collection & SIEM Integration:**
   - Log Analytics Workspace created and linked to Microsoft Sentinel.
   - Installed Windows Security Events connector (via AMA).
   - Enabled log collection and queried using KQL (SecurityEvent) for login failures and other events.
6. **Threat Visualisation:**
   - Created watchlists to map attacker IP locations.
   - Used Sentinel Workbooks to display global maps showing attack sources.
7. **Alerts & Incident Response:**
   - Configured Sentinel to generate alerts for security incidents.
   - Alerts include severity and status tracking for actionable monitoring.

Outcomes & Key Learnings

The project resulted in the following outcomes and key learnings:

- **Realised the scale of global threats:** Leaving the VM open for 24 hours resulted in 20,000+ login attempts from around the world, demonstrating how quickly exposed systems are targeted.
- **Importance of layered security:** Understood that firewalls, security rules, and correct rule order/priority are critical. A misconfigured setting can leave a system fully exposed.
- **Hands-on experience with SIEM & KQL:** Gained practical exposure to Microsoft Sentinel, Log Analytics Workspace, and running KQL queries to collect, filter, and interpret security logs.
- **Developed Azure familiarity:** Learned how Azure resources (VMs, resource groups, NSGs, agents) connect and saw how the platform can centralise security monitoring in a user-friendly way.
- **Practical OS awareness:** Gained familiarity with Windows 10 Pro setup and security logs, which was useful since my day-to-day is on macOS. Recognising how attackers target Windows specifically adds value to my cyber perspective.

- **Incident detection:** Learned the value of proactive monitoring and alerting to move from raw log collection to actionable defence.

Next Steps & Future Improvements

The following next steps and future improvements:

- **Incident Response Simulation:** Move beyond simply receiving alerts and test how to respond to them, such as isolating virtual machines, blocking IP addresses, or escalating incidents.
- **Hardening the Honeypot:** Set up multiple virtual machines with different configurations to compare attacker behaviour, for example Windows versus Linux honeypots.
- **Firewall Best Practices:** Rather than disabling the firewall, configure it with intentional rules to monitor both blocked and allowed traffic, making the setup closer to a real production environment.
- **Automation and Playbooks:** Use Sentinel automation rules and playbooks to automatically react to suspicious activity, such as sending email alerts or quarantining resources.
- **Extended Monitoring:** Keep the honeypot running for a longer period to gather more data, then use KQL dashboards to analyse patterns and common attack methods.

Screenshots