Incident Management Playbook

Incident: CUSTOM: Possible Privilege Escalation (Global Admin Role Assignment)

KQL-Query

```
// Viewing a specific existing password
let CRITICAL_PASSWORD_NAME = "Tenant-Global-Admin-Password";
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.KEYVAULT"
| where OperationName == "SecretGet"
| where id_s contains CRITICAL_PASSWORD_NAME
```

This query is particularly useful for system administrators and security teams to monitor and audit the assignment of critical administrative roles within their organization's IT environment. By keeping track of who is assigned high-level administrative privileges and when, the organization can enhance its security posture by ensuring that only authorized and intended individuals have such access. This is crucial for compliance with security policies and regulations that govern access control and privilege management.

Incident Description

- This incident involves the unexpected assignment of the Global Administrator role to a user account in Azure AD.

Initial Response Actions

- Verify the authenticity of the alert or report.
- Identify the user account that was assigned the Global Administrator role.
- Determine how and when the role assignment occurred.
- Assess the potential impact of the incident.

Containment and Recovery

- Revoke the Global Administrator role from the affected user account immediately if unintended, otherwise skip to the documentation phase.

- Check for any other unauthorized role assignments made by the attacker and revoke them if necessary.
- Identify the root cause of the incident and take corrective actions to prevent similar incidents from occurring in the future.
- Restore any data or system configurations that may have been affected by the incident. This may involve resetting the Global Administrator password for the affected account and updating the secret in Key Vault

Document Findings and Close out Incident

[Back to Top]

CUSTOM: Possible Privilege Escalation (Azure Key Vault Critical Credential Retrieval or Update)

KQL Query

```
// Viewing a specific existing password
let CRITICAL_PASSWORD_NAME = "Tenant-Global-Admin-Password";
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.KEYVAULT"
| where OperationName == "SecretGet"
| where id_s contains CRITICAL_PASSWORD_NAME
```

This query is designed for monitoring access to a specific sensitive secret stored in Azure Key Vault. It's aimed at detecting and auditing when a critical credential, identified here as "Tenant-Global-Admin-Password," is accessed. This helps in ensuring that sensitive credentials are accessed only by authorized users and provides a way to track potentially unauthorized or suspicious retrieval attempts.

Incident Description

- This incident involves the unexpected reading of a critical secret from the organization's Key Vault.

Initial Response Actions

- Verify the authenticity of the alert or report.
- Identify the secret that was read and the user or application that read it.
- Determine how and when the secret was read.

- Assess the potential impact of the incident.

Containment and Recovery

- Revoke access to the secret from the affected user or application immediately if unintended, otherwise skip to the documentation phase.
- Check for any other unauthorized access to the secret and revoke it if necessary.
- Monitor the affected systems for any suspicious activity related to the incident.
- Identify the root cause of the incident and take corrective actions to prevent similar incidents from occurring in the future.
- Change the secret if it was compromised.

Document Findings and Close out Incident

[Back to Top]

CUSTOM: CUSTOM: Malware Detected

KQL Query

```
// Malware detection
// Malware detected grouped by threat.
// To create an alert for this query, click '+ New alert rule'
ProtectionStatus
| where ThreatStatus != "No threats detected"
| summarize AggregatedValue = count() by Threat, Computer, _ResourceId
```

This KQL (Kusto Query Language) query is utilized to monitor and summarize detected malware incidents, grouped by the specific threat type. The query is intended for use in environments where monitoring and alerting on malware detection are critical for maintaining cybersecurity.

Incident Description

- This incident involves malware being detected on a workstation, potentially compromising the confidentiality, integrity, or availability of the system and data.

Initial Response Actions

- Verify the authenticity of the alert or report.
- Identify the primary user account of the system if applicable

- Notify any affected stakeholders, such as users or customers, as appropriate, and provide them with guidance on how to protect themselves from potential harm.
- Run a full system scan using an up-to-date antivirus software to identify and remove the malware.
- If the malware cannot be removed or is suspected to have caused significant damage, shut down the workstation and disconnect it from the network.

Containment and Recovery

- Quarantine the infected workstation and any other systems that may have been impacted by the malware.
- Restore the infected workstation to a known clean state, such as a system image or a clean installation of the operating system and applications.

Document Findings and Close out Incident

[Back to Top]

CUSTOM: Brute Force SUCCESS - Windows and Linux

Windows KQL:

```
// Brute Force Success Windows
let FailedLogons = SecurityEvent
| where EventID == 4625 and LogonType == 3
| where TimeGenerated > ago(1h)
| summarize FailureCount = count() by AttackerIP = IpAddress, EventID, Activity, LogonType,
DestinationHostName = Computer
| where FailureCount >= 5;
let SuccessfulLogons = SecurityEvent
| where EventID == 4624 and LogonType == 3
| where TimeGenerated > ago(1h)
| summarize SuccessfulCount = count() by AttackerIP = IpAddress, LogonType,
DestinationHostName = Computer, AuthenticationSuccessTime = TimeGenerated;
SuccessfulLogons
| join kind = inner FailedLogons on DestinationHostName, AttackerIP, LogonType
| project AuthenticationSuccessTime, AttackerIP, DestinationHostName, FailureCount,
SuccessfulCount
```

This KQL (Kusto Query Language) query is used to identify successful logins following multiple failed attempts, a common indicator of a brute force attack. It specifically analyzes security event logs within the last hour to detect unusual login patterns.

Linux KQL:

```
// Brute Force Success Linux
let FailedLogons = Syslog
| where Facility == "auth" and SyslogMessage startswith "Failed password for"
| where TimeGenerated > ago(1h)
| project TimeGenerated, SourceIP = extract(@"\\b\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\b", 0,
SyslogMessage), DestinationHostName = HostName, DestinationIP = HostIP, Facility,
SyslogMessage, ProcessName, SeverityLevel, Type
| summarize FailureCount = count() by AttackerIP = SourceIP, DestinationHostName
| where FailureCount >= 5;
let SuccessfulLogons = Syslog
| where Facility == "auth" and SyslogMessage startswith "Accepted password for"
| where TimeGenerated > ago(1h)
| project TimeGenerated, SourceIP = extract(@"\\b\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\b", 0,
SyslogMessage), DestinationHostName = HostName, DestinationIP = HostIP, Facility,
SyslogMessage, ProcessName, SeverityLevel, Type
| summarize SuccessfulCount = count() by SuccessTime = TimeGenerated, AttackerIP =
SourceIP, DestinationHostName
| where SuccessfulCount >= 1
| project DestinationHostName, SuccessfulCount, AttackerIP, SuccessTime;
let BruteForceSuccesses = SuccessfulLogons
| join kind = inner FailedLogons on AttackerIP, DestinationHostName;
BruteForceSuccesses
```

This KQL (Kusto Query Language) query helps identify instances where multiple failed login attempts are followed by a successful login, potentially indicating a brute force attack on Linux systems. The query operates on syslog data, specifically analyzing authentication logs within the last hour.

Incident Description

- This incident involves observation of potential brute force attempts against a Windows and Linux VM.

Initial Response Actions

- Verify the authenticity of the alert or report.
- Immediately isolate the machine and change the password of the affected user
- Identify the origin of the attacks and determine if they are attacking or involved with anything else
- Determine how and when the attack occurred
    - Are the NSGs not being locked down? If so, check other NSGs
- Assess the potential impact of the incident.
    - What type of account was it? Permissions?

Containment and Recovery

- Lock down the NSG assigned to that VM/Subnet, either entirely, or to allow only necessary traffic
- Reset the affected user's password
- Enable MFA

Document Findings and Close out Incident

[Back to Top]

CUSTOM: Brute Force SUCCESS - Azure Active Directory

KQL Query

```
// Failed AAD logon
let FailedLogons = SigninLogs
| where Status.failureReason == "Invalid username or password or Invalid on-premise
username or password."
| where TimeGenerated > ago(1h)
| project TimeGenerated, Status = Status.failureReason, UserPrincipalName, UserId,
UserDisplayName, AppDisplayName, AttackerIP = IPAddress,
IPAddressFromResourceProvider, City = LocationDetails.city, State = LocationDetails.state,
Country = LocationDetails.country, Latitude = LocationDetails.geoCoordinates.latitude,
Longitude = LocationDetails.geoCoordinates.longitude
| summarize FailureCount = count() by AttackerIP, UserPrincipalName;
let SuccessfulLogons = SigninLogs
| where Status.errorCode == 0
| where TimeGenerated > ago(1h)
| project TimeGenerated, Status = Status.errorCode, UserPrincipalName, UserId,
UserDisplayName, AppDisplayName, AttackerIP = IPAddress,
IPAddressFromResourceProvider, City = LocationDetails.city, State = LocationDetails.state,
Country = LocationDetails.country, Latitude = LocationDetails.geoCoordinates.latitude,
Longitude = LocationDetails.geoCoordinates.longitude
| summarize SuccessCount = count() by AuthenticationSuccessTime = TimeGenerated,
AttackerIP, UserPrincipalName, UserId, UserDisplayName;
let BruteForceSuccesses = SuccessfulLogons
| join kind = inner FailedLogons on AttackerIP, UserPrincipalName;
BruteForceSuccesses
| project AttackerIP, TargetAccount = UserPrincipalName, UserId, FailureCount, SuccessCount,
AuthenticationSuccessTime
```

This KQL (Kusto Query Language) query helps in identifying potential brute force attacks by analyzing patterns of failed and subsequently successful logins within Azure Active Directory. It monitors login attempts over the past hour and separates them into failed and successful logins, specifically looking for those linked by the same IP and user principal name.

Incident Description

- This incident involves observation of potential brute force success against Azure Active Directory

Initial Response Actions

- Verify the authenticity of the alert or report.
- Immediately identify and Revoke Sessions/Access for affected user
- Identify the origin of the attacker and determine if they are attacking or involved with anything else
- Assess the potential impact of the incident.
    - What type of account was it?
    - What Roles did it have?
    - How long has it been since the breach went unattended?

Containment and Recovery

- Reset the affected user's password and Roles if applicable
- Enable MFA
- Consider preventing any logins from outside the US with Conditional Access

Document Findings and Close out Incident

[Back to Top]

CUSTOM: Possible Lateral Movement (Excessive Password Resets)

KQL Query

```
AuditLogs
| where OperationName startswith "Change" or OperationName startswith "Reset"
| order by TimeGenerated
| summarize count() by tostring(InitiatedBy)
| project Count = count_, InitiatorId = parse_json(InitiatedBy).user.id, InitiatorUpn =
parse_json(InitiatedBy).user.userPrincipalName, InitiatorIpAddress =
parse_json(InitiatedBy).user.ipAddress
| where Count >= 10
```

This KQL (Kusto Query Language) query is used for monitoring operations within Azure environments that might indicate administrative activities or configuration changes occurring at unusually high frequencies. The query filters and summarizes audit log entries related to "Change" and "Reset" operations to help identify and investigate potential unauthorized or risky activities.

Incident Description

- This incident involves observation of potential lateral movement based on excessive password resets

Initial Response Actions

- Verify the authenticity of the alert or report.
- Immediately identify and Revoke Sessions/Access for any affected users
- Identify the attacker and determine if they are attacking or involved with anything else
- Observe the target accounts which had their passwords reset.
    - Have any of them immediately logged in or done anything else?
- Assess the potential impact of the incident.
    - What type of accounts are involved?
    - What Roles did it have?
    - How long has it been since the breach went unattended?

Containment and Recovery

- Reset the affected users’ password and Roles if applicable
- Enable MFA

Document Findings and Close out Incident

[Back to Top]

CUSTOM: Brute Force ATTEMPT – Windows

KQL Query

```
// Failed logon
SecurityEvent
| where EventID == 4625
| where TimeGenerated > ago(60m)
| summarize FailureCount = count() by AttackerIP = IpAddress, EventID, Activity,
DestinationHostName = Computer
| where FailureCount >= 10
```

Incident Description

- This incident involves detecting IP addresses that have made multiple failed attempts to log into various accounts on Windows systems, indicative of a brute force attack.

Initial Response Actions

- **Verify the Alert:** Ensure the failed logins are not due to user errors or legitimate access issues.
- **Identify Attack Source:** Trace back to the IP addresses responsible for the high number of failed attempts.
- **Assess Impact:** Evaluate which systems were targeted and the potential risk associated with these brute force attempts.

Containment and Recovery

- **Block Suspicious IPs:** Temporarily block the identified IP addresses to prevent further attempts.
- **Review Account Security:** Check the security settings for the targeted accounts, ensuring that they have strong, complex passwords and multi-factor authentication is enabled.
- **Monitor for Further Attempts:** Keep an active watch on logs for continued patterns of failed logins or escalations.

Document Findings and Close out Incident

- **Documentation:** Thoroughly document the incident's details, including the source IP addresses, targeted hostnames, and the response actions.
- **Review and Approve:** Have the incident documentation reviewed and approved by IT security management.
- **Close Incident:** Officially close the incident in the incident management system, updating all stakeholders on the resolution and preventive measures implemented.

[Back to Top]

CUSTOM: Brute Force ATTEMPT - MS SQL Server

KQL Query

```
// Brute Force Attempt MS SQL Server
let IpAddress_REGEX_PATTERN = @"\\b\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\b";
Event
| where EventLog == "Application"
| where EventID == 18456
| where TimeGenerated > ago(1hr)
| project TimeGenerated, AttackerIP = extract(IpAddress_REGEX_PATTERN, 0,
RenderedDescription), DestinationHostName = Computer, RenderedDescription
| summarize FailureCount = count() by AttackerIP, DestinationHostName
| where FailureCount >= 5
```

This KQL (Kusto Query Language) query is designed to identify brute force attack attempts against Microsoft SQL Server by monitoring for multiple failed login attempts. It specifically targets the **Application** event log for SQL Server login failure events.

Incident Description

- This incident involves detecting IP addresses that repeatedly fail to log into an MS SQL Server, which may indicate an ongoing brute force attack trying to guess user credentials.

Initial Response Actions

- **Verify the Alerts:** Confirm that the detected failed logins are not due to configuration issues or legitimate access attempts.
- **Identify Attack Source:** Trace the origin of the attack using the IP addresses identified in the logs.
- **Assess Impact:** Evaluate the potential impact on the SQL Server and associated data security.

Containment and Recovery

- **Block Suspicious IPs:** Implement firewall rules or other IP blocking mechanisms to prevent further attempts from the identified sources.
- **Audit Affected Accounts:** Review the security settings of accounts targeted in the brute force attempt, ensuring they are secured with strong passwords and, if applicable, multi-factor authentication.

- **Monitor Network Traffic:** Enhance monitoring of network traffic to and from the SQL Server to detect any further suspicious activity.

Document Findings and Close out Incident

- **Documentation:** Record all pertinent details of the incident, including the source IPs, the number of failed attempts, and measures taken in response.
- **Review and Approve:** Ensure that the incident documentation is reviewed by IT security and approved by compliance officers.
- **Close Incident:** Formally close out the incident in the security management system and communicate the findings and corrective actions taken to all relevant stakeholders.

[Back to Top]

CUSTOM: Brute Force ATTEMPT - Linux Syslog

KQL Query

```
// Brute Force Success Linux
let IpAddress_REGEX_PATTERN = @"\\b\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\b";
Syslog
| where Facility == "auth" and SyslogMessage startswith "Failed password for"
| where TimeGenerated > ago(1h)
| project TimeGenerated, AttackerIP = extract(IpAddress_REGEX_PATTERN, 0,
SyslogMessage), DestinationHostName = HostName, DestinationIP = HostIP, Facility,
SyslogMessage, ProcessName, SeverityLevel, Type
| summarize FailureCount = count() by AttackerIP, DestinationHostName, DestinationIP
| where FailureCount >= 10
```

This KQL (Kusto Query Language) query is crafted to identify potential brute force attacks against Linux systems by monitoring the **Syslog** for patterns of failed password attempts. It focuses on identifying repeated unsuccessful login attempts from specific IP addresses.

Incident Description

- This incident involves detecting IP addresses that have made multiple failed attempts to log into Linux servers, indicative of a possible brute force attack trying to compromise system credentials.

Initial Response Actions

- **Verify the Alert:** Ensure the detected failed logins are not due to user errors or system issues.
- **Identify Attack Source:** Trace back to the IP addresses responsible for the high number of failed attempts.
- **Assess Impact:** Evaluate which systems were targeted and the potential security risk posed by these brute force attempts.

Containment and Recovery

- **Block Suspicious IPs:** Temporarily block the identified IP addresses at the network level to prevent further attempts.
- **Review Account Security:** Check the security settings for the accounts targeted in the brute force attempt, ensuring they are secured with strong, complex passwords and, if applicable, multi-factor authentication.
- **Monitor for Further Attempts:** Keep an active watch on syslog entries for continued patterns of failed logins or escalation to successful logins.

Document Findings and Close out Incident

- **Documentation:** Thoroughly document the incident's details, including the source IP addresses, targeted hostnames, number of failed attempts, and the response actions taken.
- **Review and Approve:** Have the incident documentation reviewed and approved by IT security management.
- **lose Incident:** Formally close the incident in the incident management system and update all stakeholders on the resolution and preventive measures implemented.

[Back to Top]

CUSTOM: Brute Force ATTEMPT - Azure SQL Server

KQL Query (Needs Revision)

```
let threshold = 10; // Define a threshold for failed login attempts
let timeframe = 1h; // Define the timeframe to look back for failed attempts

AzureDiagnostics
| where ResourceProvider == "MICROSOFT.SQL" and Category == "SQLSecurityAuditEvents"
| where OperationName == "FAILED_LOGIN_GROUP" // Or use the appropriate event identifier for failed logins
| where TimeGenerated > ago(timeframe)
| summarize FailureCount = count() by bin(TimeGenerated, 5m), IPAddress = client_IP_s,
ServerName = serverName_s
| where FailureCount >= threshold
| project TimeGenerated, IPAddress, ServerName, FailureCount
```

This KQL (Kusto Query Language) query is specifically designed to monitor Azure SQL Server for signs of brute force attacks by analyzing patterns of failed login attempts. It leverages Azure Diagnostics logs focused on SQL security audit events to detect and summarize suspicious activities.

Incident Description

- This incident involves monitoring for frequent failed login attempts to Azure SQL Server, which may suggest a brute force attack aimed at guessing user credentials.

Initial Response Actions

- **Verify Significance:** Confirm the frequency and pattern of the failed attempts to ensure they are not regular login failures.
- **Trace IP Addresses:** Identify the source IP addresses associated with the excessive failed logins.
- **Evaluate Impact:** Assess the potential security impact on the targeted SQL Server instances.

Containment and Recovery

- **Block Suspicious IPs:** Consider blocking IP addresses that are identified as sources of brute force attempts.
- **Strengthen Authentication:** Review and enhance the authentication mechanisms, such as implementing stronger password policies and multi-factor authentication.
- **Monitor Further Activities:** Increase monitoring to detect any further unusual activities or successful breaches.

Document Findings and Close out Incident

- **Documentation:** Thoroughly record the details of the incident, including the detected patterns, IP addresses involved, and any measures taken in response.
- **Review and Approve:** Ensure that the incident documentation is reviewed by cybersecurity teams and approved by relevant authorities.
- **Close Incident:** Formally conclude the incident in the security management system and communicate the findings and corrective actions to all stakeholders.

[Back to Top]

CUSTOM: Brute Force ATTEMPT - Azure Active Directory

****

KQL Query

```
SigninLogs
| where ResultDescription == "Invalid username or password or Invalid on-premise username or
password."
| project TimeGenerated, ResultDescription, UserPrincipalName, UserId, AppDisplayName,
IPAddress, IPAddressFromResourceProvider, City = LocationDetails.city, State =
LocationDetails.state, Country = LocationDetails.country, Latitude =
LocationDetails.geoCoordinates.latitude, Longitude = LocationDetails.geoCoordinates.longitude
```

This KQL (Kusto Query Language) query is tailored to identify potential brute force attacks targeting Azure Active Directory by monitoring for failed sign-in attempts where invalid usernames or passwords are recorded. The query provides insights into the specific login attempt details, helping to identify and respond to suspicious activities effectively.

Incident Description

- This incident involves monitoring for signs of brute force attacks in Azure Active Directory, characterized by multiple failed login attempts due to incorrect user credentials.

Initial Response Actions

- **Verify Sign-In Details:** Check the authenticity and frequency of the failed login attempts to confirm if they signify a brute force attack.
- **Identify Affected Users:** Determine which user accounts are being targeted and assess whether these attempts follow any specific patterns or trends.
- **Assess Impact:** Evaluate the potential impact on affected user accounts and the broader security posture.

Containment and Recovery

- **Secure Affected Accounts:** Promptly enforce password resets and, if feasible, enable multi-factor authentication for affected accounts to heighten security.

- **Block Suspicious IPs:** Implement IP blocking or rate limiting for IPs that show repeated suspicious activities to prevent further attempts.
- **Monitor Further Activities:** Increase monitoring on affected user accounts and relevant IP addresses to detect any further suspicious actions.

Document Findings and Close out Incident

- **Documentation:** Log all pertinent details of the incident, including the specifics of the detected failed login attempts, the response actions taken, and any follow-up measures planned.
- **Review and Approve:** Ensure that the incident documentation is thoroughly reviewed and approved by cybersecurity oversight teams.
- **Close Incident:** Officially close the incident in the incident management system, ensuring all stakeholders are informed about the incident's resolution and preventive measures taken.