THOMAS MORE

Lab 4.6

Creating an Analytics Rule and Investigating an Incident

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Creating an Analytics Rule and Investigating an Incident

1.1. Context

In Lab 4.5 you used KQL to manually hunt through application and infrastructure logs.

In this lab, you turn one of those hunting queries into an automated Microsoft Sentinel analytics rule. The rule detects external access to an endpoint that should only be reachable internally.

You will then trigger the detection, review the generated incident, and perform a short investigation.

1.2. Learning goals

By the end of this lab, you can:

[**Create** a scheduled analytics rule from a KQL query] [**Configure** rule severity, tactics, scheduling, and lookback] [**Trigger** an incident with simulated attack traffic] [**Investigate** an incident using Sentinel and KQL] [**Explain** how a manual hunting query becomes an automated detection]

1.3. Estimated time

45-60 minutes

Time	Activity
10 minutes	Review and test the detection query
15 minutes	Create the analytics rule
10 minutes	Generate attack traffic
15 minutes	Review and investigate the incident
5 minutes	Reflection

1.4. Before you start

Make sure you have:

[Microsoft Sentinel enabled on the Log Analytics workspace] [App Service HTTP logs flowing into `AppServiceHTTPLogs`] [The internal backend URL] [The KQL query from Lab 4.5 that detects external access to `/internal` endpoints]

2. Part 1: Review the Detection Query

2.1. Purpose

The internal backend should not be directly reachable from the internet.

This query detects requests to `/internal` endpoints from public IP addresses.

2.2. Detection Query

Run this query in Log Analytics first. Only create an analytics rule after you have confirmed that the query works.

```
AppServiceHTTPLogs
| where TimeGenerated > ago(24h)
| where _ResourceId has "Microsoft.Web/sites"
| extend
    Path = tostring(column_ifexists("CsUriStem", "")),
    ClientIP = tostring(column_ifexists("CIp", "")),
    Method = tostring(column_ifexists("CsMethod", "")),
    StatusCode = toint(column_ifexists("ScStatus", int(null)))
| where Path startswith "/internal"
| where ipv4_is_private(ClientIP) == false
| extend
    AccessResult = case(
        StatusCode between (200 .. 299), "Access succeeded",
        StatusCode in (401, 403), "Blocked by authentication or authorisation",
        StatusCode == 404, "Endpoint not found",
        StatusCode >= 500, "Server error",
        "Other"
    )
| project
    TimeGenerated,
    Alert = "External client requested internal endpoint",
    AccessResult,
    ClientIP,
    Method,
    Path,
    StatusCode,
    Resource = _ResourceId
| order by TimeGenerated desc
```

2.3. What the Query Does

Line	Purpose
`AppServiceHTTPLogs`	Uses platform-level HTTP logs from App Service
`Path startswith "/internal"`	Focuses on internal backend endpoints
`ipv4_is_private(ClientIP) == false`	Filters out private IP ranges and keeps public clients
`AccessResult`	Classifies whether the request succeeded, failed, or was blocked

`project`	Keeps only the columns needed for the alert and investigation

> **Question**
>
> Why is `AppServiceHTTPLogs` a good source for this detection? Why is a successful 2xx response more serious than a 401 or 403?

3. Part 2: Create the Analytics Rule

3.1. Navigate to Analytics

1. Open Microsoft Sentinel.
2. Go to Analytics.
3. Click Create.
4. Choose Scheduled query rule.

3.2. General Settings

Use the following settings:

Setting	Value
Rule name	External access to internal backend
Description	Detects public IP addresses requesting `/internal` endpoints on the internal backend
Severity	High
MITRE tactics	InitialAccess, LateralMovement
MITRE techniques	T1190 - Exploit Public-Facing Application

3.3. Rule Query

For the analytics rule, use a shorter time window. The rule will run regularly, so it does not need to search the full 24 hours.

```
AppServiceHTTPLogs
| where TimeGenerated > ago(10m)
| where _ResourceId has "Microsoft.Web/sites"
| extend
    Path = tostring(column_ifexists("CsUriStem", "")),
    ClientIP = tostring(column_ifexists("CIp", "")),
    Method = tostring(column_ifexists("CsMethod", "")),
    StatusCode = toint(column_ifexists("ScStatus", int(null)))
| where Path startswith "/internal"
| where ipv4_is_private(ClientIP) == false
| summarize
    Attempts = count(),
    SuccessfulAttempts = countif(StatusCode between (200 .. 299)),
    FirstSeen = min(TimeGenerated),
    LastSeen = max(TimeGenerated),
    Paths = make_set(Path, 10),
    StatusCodes = make_set(StatusCode, 10)
    by ClientIP, Resource = tostring(_ResourceId)
| extend
    AlertTitle = "External access to internal backend",
    AccessResult = iff(
        SuccessfulAttempts > 0,
        "External access succeeded",
```

```
        "External access attempted"
    )
```

3.4. Entity Mapping

If Sentinel asks for entity mapping, map:

Entity	Column
IP	`ClientIP`
Azure Resource	`Resource`

3.5. Query Scheduling

Use the following schedule:

Setting	Value
Run query every	5 minutes
Lookup data from the last	10 minutes

> **Note**
> The lookback window is slightly longer than the rule frequency to reduce the chance of missing delayed logs. This can create duplicate matches, but Sentinel can group alerts into incidents.

3.6. Incident Settings

Enable incident creation.

Use alert grouping if available:

[Group alerts into a single incident if they have the same `ClientIP`] [Group alerts generated within a short time window, such as 1 hour]

> **Question**
> Why might a 10-minute lookback be safer than a 5-minute lookback when logs can arrive late?

4. Part 3: Trigger the Rule

4.1. Generate Test Traffic

From your terminal, send requests directly to the internal backend.

> You might want to enable public access just to show what would happen.

```
curl https://<your-internal-backend-url>/internal/data/welcome
curl https://<your-internal-backend-url>/internal/health
```

Then wait for:

- App Service logs to arrive in Log Analytics
- The scheduled analytics rule to run
- Sentinel to create the incident

> **Note**
>
> This can take several minutes. If no incident appears, first run the KQL query manually in Log Analytics to confirm that the test traffic is visible.

5. Part 4: Review the Incident

5.1. Find the Incident

In Sentinel, go to Incidents.

Open the incident named:

`External access to internal backend`

Review:

Field	What to check
Severity	Is the severity appropriate?
Status	Is the incident New, Active, or Closed?
Entities	Which IP address triggered the detection?
Timeline	Which requests caused the incident?
Tactics and techniques	Do they match the scenario?

> **Question**
> Which IP address triggered the incident? Which internal paths were requested? Did any request return a successful status code?

6. Part 5: Investigate with KQL

6.1. Step 1: Investigate the Source IP

Replace `<suspicious-ip>` with the IP address from the incident.

```
let suspiciousIp = "<suspicious-ip>";

AppServiceHTTPLogs
| where TimeGenerated > ago(24h)
| extend
    Path = tostring(column_ifexists("CsUriStem", "")),
    ClientIP = tostring(column_ifexists("CIp", "")),
    Method = tostring(column_ifexists("CsMethod", "")),
    StatusCode = toint(column_ifexists("ScStatus", int(null)))
| where ClientIP == suspiciousIp
| project TimeGenerated, ClientIP, Method, Path, StatusCode, Resource = _ResourceId
| order by TimeGenerated asc
```

6.2. Step 2: Check Application Insights for the Same IP

```
let suspiciousIp = "<suspicious-ip>";

AppRequests
| where TimeGenerated > ago(24h)
| where tostring(ClientIP) == suspiciousIp
| project
    TimeGenerated,
    Name,
    ResultCode,
    Success,
    DurationMs,
    OperationId,
    AppRoleName
| order by TimeGenerated asc
```

6.3. Step 3: Check Sensitive Events

Check whether there were Key Vault secret operations around the same time.

```
AZKVAuditLogs
| where TimeGenerated > ago(24h)
| where OperationName has "Secret"
| project
    TimeGenerated,
    OperationName,
    CallerIpAddress,
    Identity,
    ResultDescription,
    _ResourceId
| order by TimeGenerated desc
```

Check whether Azure resources or role assignments changed.

```
AzureActivity
| where TimeGenerated > ago(24h)
| where OperationNameValue has_any (
    "Microsoft.Authorization/roleAssignments/write",
    "Microsoft.Authorization/roleAssignments/delete",
    "write",
    "delete",
    "action"
)
| project
    TimeGenerated,
    OperationNameValue,
    ActivityStatusValue,
    Caller,
    ResourceGroup,
    ResourceId
| order by TimeGenerated desc
```

6.4. Step 4: Decide the Response

Use your findings to decide what you would do next.

Finding	Possible response
External request returned 2xx	Treat as confirmed exposure; review access restrictions and network design
External request returned 401 or 403	Service is reachable but protected; still review why it is publicly exposed
Many requests from same IP	Possible scanning; consider blocking or rate limiting
Key Vault secret access near same time	Investigate identity abuse and consider secret rotation
Role assignment changes	Investigate privilege escalation or persistence

> **Question**
>
> Was this a failed attempt, a successful exposure, or a broader incident? Which evidence supports your conclusion?

7. Reflection

Answer briefly.

1. What is the difference between a KQL hunting query and a Sentinel analytics rule?
2. Why should the rule look back slightly further than the run frequency?
3. Why is external access to an internal backend a high-severity finding?
4. Which logs were most useful during the investigation?
5. What would you change in the architecture to prevent this incident?

8. Summary

In this lab, you converted a manual KQL query into a Microsoft Sentinel analytics rule.

You then generated test traffic, reviewed the incident, and investigated the suspicious IP across App Service logs, Application Insights, Key Vault audit logs, and Azure Activity.

The key lesson is that detection is only useful when it leads to investigation. A good analytics rule should provide enough context for an analyst to answer:

“What happened, how serious is it, and what should we do next?”