THOMAS MORE

Lab 4.1

Running the Pipeline and Observing the Monitoring Baseline

Toepasbare Cloud Security voor IT-Professionals

2025 - 2026

Contents

1. Running the Pipeline and Observing the Monitoring Baseline

1.1. Context

Day 4 starts with a fresh repository containing only the Day 4 infrastructure — the same resource layout as Day 3, but with monitoring enabled from the start. This lab runs the pipeline to deploy those resources and observe the Terraform plan (what changes are being made) and the apply (which resources are created or modified).

Sentinel is enabled interactively in the Azure portal during Lab 4.4, and the daily ingestion cap is set in Lab 4.2 — neither is configured through Terraform.

1.2. Setting up a fresh repository

For Day 4, start with a clean repository that contains only the Day 4 infrastructure files. This gives you a focused view of the monitoring changes without the noise of previous labs.

1. Create a new Git repository for Day 4 (e.g., `day-4-monitoring`)
2. Copy the Day 4 infrastructure files into it:
   - `infrastructure/` — Terraform configuration with diagnostic settings enabled
   - `.checkov.yml`, `.semgrep.yml`, `trivy.yaml` — security scan configs
3. Create a new Azure DevOps project and repository for this repo
4. Push the Day 4 files to your new repo

> **Note**
>
> You can remove or archive your Day 3 repository if you prefer — it won't be needed for Day 4.

1.3. Running the pipeline

> **I hope you changed the vars**
>
> Make sure to update `infrastructure/dev.tfvars` with your unique prefix and environment values before running the pipeline. This ensures that the resources are created in your subscription without naming conflicts.

The Day 4 pipeline reuses the same structure as Day 3, with three security scanning stages that are **commented out** in the starter template:

- **Checkov** — IaC scanning of Terraform files
- **Semgrep** — static code analysis
- **Trivy** — dependency vulnerability scanning

In this lab you will uncomment and enable these stages. The pipeline then produces artifact reports — downloadable from the Azure DevOps pipeline run. In later labs, you will learn how to query security findings alongside infrastructure logs in Sentinel.

1.4. Learning goals

By the end of this lab, you should be able to explain:

- What the Terraform plan shows when enabling diagnostic settings.
- The relationship between Application Insights, Log Analytics, and Sentinel.
- Why Sentinel and the daily cap are configured in the portal, not through Terraform.
- How the Azure Activity Log connector must be enabled manually in the portal.

1.5. Estimated time

100 minutes

Suggested timing:

Time	Activity
10 minutes	Review the Terraform configuration
20 minutes	Run the pipeline to enable monitoring
15 minutes	Review Terraform plan and apply output
15 minutes	Identify new resources in the Azure portal
10 minutes	Enable the Azure Activity Log connector in Sentinel
20 minutes	Reflection and discussion

1.6. Before you start

Make sure you have:

- [ ] New Day 4 repository created with infrastructure files pushed
- [ ] Azure DevOps project and pipeline configured for the Day 4 repo, with `azure-subscription` service connection
- [ ] Terraform installed on the pipeline agent (the pipeline installs it automatically)

> **Note**
>
> The Day 4 infrastructure includes all resources needed — App Services, Key Vault, Storage, Log Analytics, Application Insights, and VNet.

2. Review the Terraform Configuration

Open `infrastructure/dev.tfvars` and verify that `enable_diagnostic_settings = true`. The pipeline runs Terraform against this configuration.

- **Count-gated resources**: The diagnostic setting resources already exist in `main.tf` but are only created when `enable_diagnostic_settings` is true.
- **Security scans**: Checkov (IaC), Semgrep (code), and Trivy (dependencies) — stages are commented out; uncomment them before running the pipeline.
- **Sentinel**: Sentinel is enabled interactively in the Azure portal during Lab 4.4, not through Terraform.
- **Daily cap**: The daily ingestion cap is set in the portal during Lab 4.2 to prevent runaway costs.

> **Note**
>
> The Day 4 infrastructure deploys all resources — App Services, Key Vault, Storage, Log Analytics, Application Insights, and VNet — with monitoring enabled from the start.

> **Question**
>
> Why does the pipeline use terraform plan before terraform apply? What information does the plan provide that the apply does not?

2.1. Run Terraform Apply

Before running the pipeline, open the `azure-pipelines.yml` file in your repo and uncomment the three security scanning stages (Checkov, Semgrep, Trivy). These are commented out in the starter template so that students can enable them as part of this lab.

In Azure DevOps, navigate to your Day 4 pipeline and click **Run pipeline**. Select the `main` or `develop` branch and run it. The pipeline provisions the Day 4 infrastructure using Terraform against the Day 4 configuration.

> **Note**
>
> If the pipeline was triggered automatically by your previous commit, you can also view the existing run instead of triggering a new one.

Watch the pipeline execution. Terraform will:

1. Initialize the workspace and connect to the remote backend
2. Run `terraform plan` — review the output to see what resources will be created
3. Run `terraform apply` — confirm the changes

2.2. Review the Terraform Plan Output

The `terraform plan` output shows exactly what changes will be made. Look for `azurerm_monitor_diagnostic_setting` resources routing logs from App Services, Key Vault, and Storage to Log Analytics.

Your answer: Identify which diagnostic settings are being created. List each resource and which log categories it enables.

2.3. Review the Terraform Apply Output

After the plan, `terraform apply` creates the new resources. Look for:

```
Terraform will perform the following actions:

  # azurerm_monitor_diagnostic_setting.public_backend will be created
  + resource "azurerm_monitor_diagnostic_setting" "public_backend" {
      + name               = "diag-app-cloudsec-...-api-d4"
      + log_analytics_workspace_id = "/subscriptions/.../resourceGroups/.../
providers/Microsoft.OperationalInsights/workspaces/law-...-d4"
      + enabled_log {
          + category = "AppServiceHTTPLogs"
        }
      + enabled_log {
          + category = "AppServiceConsoleLogs"
        }
      + enabled_log {
          + category = "AppServiceAppLogs"
        }
      + metric {
          + category = "AllMetrics"
        }
    }
```

> **Question**
> What is the difference between `enabled_log` (text logs) and `metric` (numerical metrics) in diagnostic settings? Which categories will capture HTTP request details?

3. Verify Monitoring Configuration in the Azure Portal

3.1. Navigate to the Resource Group

Open the Azure portal and navigate to your resource group (e.g., `rg-<your_prefix>-monitoring`).

3.2. Verify Diagnostic Settings

Confirm that diagnostic settings have been created on all resources:

Resource	What was added	Location
App Service	HTTP logs, AppServiceLogs, ConsoleLogs ⟶ Log Analytics	App Service ⟶ Monitoring ⟶ Diagnostic settings
Key Vault	AuditEvents ⟶ Monitoring ⟶ Log Analytics	Key Vault ⟶ Monitoring ⟶ Diagnostic settings
Storage account	StorageRead/Write/Transaction ⟶ Log Analytics	Storage account ⟶ Monitoring ⟶ Diagnostic settings

All resources were deployed in this lab — the diagnostic settings are part of the Day 4 infrastructure, not added to existing resources.

3.3. Verify Diagnostic Settings on the Public Backend

Navigate to the public backend App Service (`app-<your_prefix>-api-d4`):

1. Click **Diagnostic settings** in the left menu
2. Verify that a diagnostic setting exists (created by Terraform)
3. Check the enabled log categories:
   - **HTTP Logs** (enabled) — captures every HTTP request
   - **App Service Application Logs** (enabled) — captures stdout/stderr output
   - **App Service Console Logs** (enabled) — captures console output
   - **App Service Platform Logs** (enabled) — captures platform events and errors
4. Check the enabled metrics:
   - **AllMetrics** (enabled) — captures all performance metrics

Your answer: Describe what log categories are enabled for the public backend App Service. Which categories will capture HTTP request details from attackers?

3.4. Verify Diagnostic Settings on Key Vault

Navigate to the Key Vault (`kv-<your_prefix>-d4`):

1. Click **Diagnostic settings** in the left menu
2. Verify that a diagnostic setting exists
3. Check the enabled categories:
   - **AuditEvents** (enabled) — captures all Key Vault audit operations

> **Note**
>
> Key Vault audit events are the primary source for tracking who accessed which secrets. This data flows through diagnostic settings into Log Analytics.

4. Enable the Azure Activity Log Connector

The Azure Activity Log connector must be enabled manually in the Sentinel portal. Terraform cannot enable it automatically.

4.1. Navigate to Microsoft Sentinel

1. In the Azure portal, search for **Microsoft Sentinel**
2. If you have never used Sentinel before, click Try Microsoft Sentinel for free
3. Onboard your Log Analytics workspace to Sentinel (the same workspace that Application Insights uses)
4. You should see the Sentinel workspace open

> **Trial**
>
> Microsoft Sentinel is free to use for the first 31 days, and then has a pay-as-you-go pricing model based on data ingested. The Azure Activity Log connector is included with Sentinel. For this lab, set a daily cap in the Azure portal (Lab 4.2) to control costs.

4.2. Enable the Azure Activity Log Connector

The Azure Activity Log connector uses Azure Policy to apply a log-streaming configuration across your subscriptions. Follow these steps:

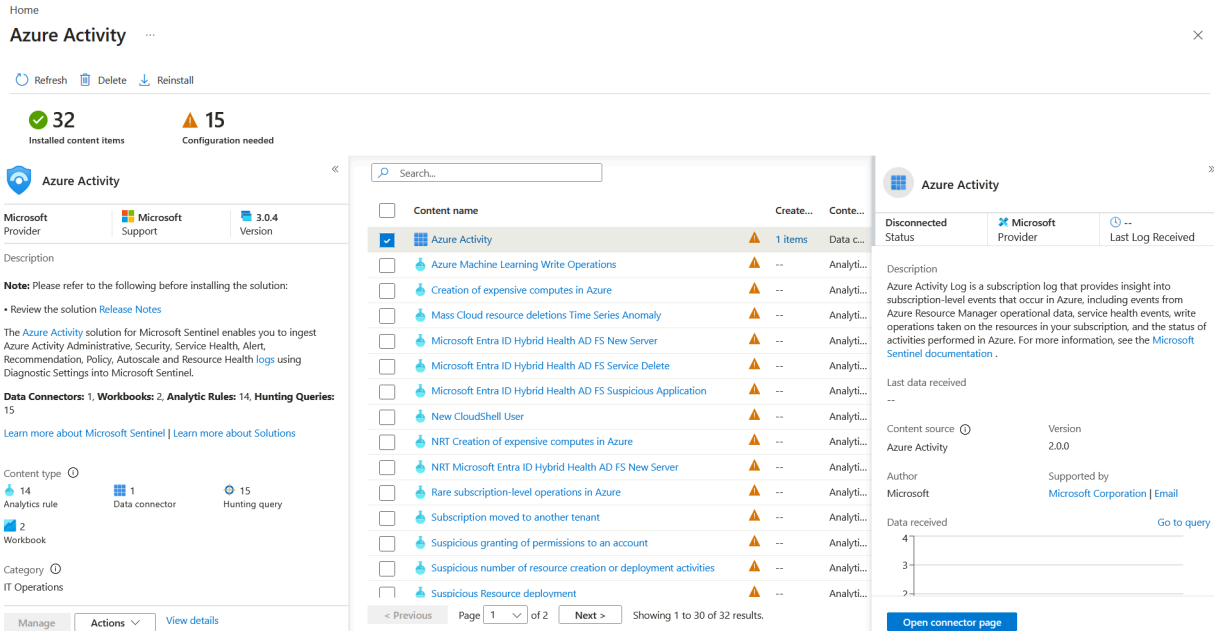
1. In the Sentinel left menu, click **Data connectors**
2. Find **Azure Activity** in the list and click **Open connector page**

The wizard automatically loads the built-in policy **"Configure Azure Activity logs to stream to specified Log Analytics workspace."** Follow the steps below:

Basics tab: Click the button with the three dots under **Scope** and select your subscription. (You can optionally include a resource group, but the subscription is required.) You must have Owner role on the scope to assign this policy.

Parameters tab: Choose your Microsoft Sentinel workspace from the **Log Analytics workspace** drop-down list. Leave all log and metric types checked (marked as "True") to ingest all Azure Activity data.

Remediation tab: Select **Create a remediation task** to apply the policy to your existing resources.



4.3. Verify the Connector is Active

After enabling the connector, the data connector page may show:

```
Status: Not connected
Data received: No data received yet
```

> **Note**
>
> The 'Not connected' status is normal — it means no management operations (create, update, delete) have been performed on your resources since you enabled the connector. The Azure Activity Log connector does not show as 'Connected' until actual data flows through it. This typically takes 15-30 minutes after a student performs any action in the Azure Portal.

> **Question**
>
> The connector shows 'Not connected' immediately after setup. Why doesn't enabling the connector itself generate activity log entries? What specific action should you take now to trigger data flow?

5. Review the Monitoring Configuration

The pipeline has completed and diagnostic settings are now active on your resources. In this section, you will review what was configured and prepare for the next labs where you'll generate traffic and explore the monitoring data.

Your answer:

1. Which resources now have diagnostic settings enabled?
2. What types of logs are being sent to Log Analytics for each resource?
3. Why is it important that both HTTPLogs and AppServiceAppLogs are enabled on the App Service?

6. Review Security Scan Reports in Pipeline Artifacts

If needed uncomment the stages first before getting the artifacts.

The Day 4 pipeline includes three security scanning stages that produce artifact reports:

Stage	Tool	What it scans	Artifact name
`SecurityIaCScanning`	Checkov	Terraform IaC (`infrastructure/`)	`security-reports-checkov`
`SecurityCodeAnalysis`	Semgrep	Application source code	`security-reports-semgrep`
`SecurityDependencyScanning`	Trivy	Node.js dependencies (SBOM + CVEs)	`security-reports-trivy`

> **Note**
> Download these artifact reports from the pipeline run. In later labs, you will learn how to query security findings alongside infrastructure logs in Sentinel for a unified view of your security posture.

6.1. Download the Security Reports

> This is a quick refresher and optional step — you will review the contents of these reports in more detail in Lab 4.4 when you integrate them with Sentinel.

After the pipeline completes, go to the pipeline run and click **Artifacts**. Download:

- `security-reports-checkov` — IaC security findings (Checkov JSON)
- `security-reports-semgrep` — Code analysis findings (Semgrep JSON/SARIF)
- `security-reports-trivy` — Dependency vulnerabilities (Trivy JSON + SBOM)

Your answer:

1. What type of vulnerability does each tool detect? Give one example for each.
2. How would you use these reports to improve your pipeline security gates?

7. Reflection Questions

Your answer:

1. Why does the pipeline use `terraform plan` before `terraform apply`? In a production environment, would you skip the plan step? Explain your reasoning in terms of security and operations.
2. What are the trade-offs between enabling all diagnostic log categories versus selecting only specific ones?
3. The pipeline produces three types of security reports (Checkov, Semgrep, Trivy). How could these downloaded reports be integrated with Sentinel to create a unified security view?

8. Summary

In this lab, you ran the Day 4 pipeline to deploy infrastructure with monitoring enabled from the start. You uncommented and enabled the security scanning stages (Checkov, Semgrep, Trivy), observed the Terraform plan and apply in Azure DevOps, identified the diagnostic settings in the Azure portal, enabled Microsoft Sentinel on your Log Analytics workspace, and reviewed the security scan reports as downloadable artifacts. Next, you'll explore the monitoring data and configure Sentinel further.