# Sentinel as Code

## What is it?

Azure DevOps is a suite of development tools and services provided by Microsoft to support the entire software development lifecycle. It includes:

* **Azure Repos**: Source control using Git.
* **Azure Pipelines**: CI/CD (Continuous Integration and Continuous Deployment) automation.
* **Azure Boards**: Agile project management tools.
* **Azure Artifacts**: Package management.
* **Azure Test Plans**: Testing tools for quality assurance.

Azure DevOps enables teams to plan work, collaborate on code development, and build and deploy applications efficiently.

## Why use a Repository?

| **Aspect** | **Azure DevOps** | **Microsoft Teams** |
| --- | --- | --- |
| **Versioning** | Full Git history | None |
| **Collaboration** | Branching, pull requests, reviews | Manual sharing |
| **Automation** | CI/CD pipelines | Not supported |
| **Security** | Role-based access, audit logs | Limited control |
| **Scalability** | Designed for enterprise DevOps | Not suitable for code deployment |

Using Teams to share code is informal and error-prone. Azure DevOps provides a structured, secure, and scalable way to manage and deploy Sentinel content.

## Potential Folder Structure:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer



## Potential Repositories:

### Azure DevOps

Azure DevOps is a comprehensive suite of tools for managing the entire software development lifecycle, tightly integrated with Microsoft Azure.

**Key Benefits of Azure DevOps**

* **End-to-End DevOps Platform**:
  + Boards for project tracking.
  + Repos for source control.
  + Pipelines for CI/CD.
  + Artifacts for package management.
  + Test Plans for QA.
* **Native Azure Integration**:
  + Seamless deployment to Azure services including Microsoft Sentinel.
  + Azure Resource Manager (ARM) and Bicep support out of the box.
* **Enterprise-Grade Security**:
  + Azure Active Directory (AAD) integration.
  + Role-based access control (RBAC).
  + Audit logs and compliance features.
* **Scalability**:
  + Designed for large teams and complex projects.
  + Supports multi-stage pipelines and environment approvals.
* **Cost Management**:
  + Free tier for small teams.
  + Pay-as-you-go model for pipelines and users.
* **Custom Agent Pools**:
  + Use Microsoft-hosted or self-hosted agents.
* **Convenience**:
  + YAML and classic pipeline editors.
  + Built-in dashboards and analytics.
* **Governance and Compliance**:
  + Policy enforcement, gated check-ins, and traceability.

### GitHub

GitHub is a widely used platform for source code management and collaboration. It supports Git-based workflows and integrates with a wide range of tools and services.

**Key Benefits of GitHub**

* **Global Collaboration**: Ideal for open-source and distributed teams with powerful pull request and code review features.
* **GitHub Actions**: Built-in CI/CD automation that supports workflows triggered by events like commits or pull requests.
* **Marketplace Integrations**: Thousands of pre-built actions and integrations with third-party tools.
* **Cost-Effective**: Free tier available with generous limits; GitHub Enterprise available for advanced needs.
* **Security Features**:
  + Dependabot for automated dependency updates.
  + Code scanning and secret detection.
* **Developer Experience**:
  + Clean, intuitive UI.
  + GitHub Copilot integration for AI-assisted coding.
* **Community and Ecosystem**:
  + Massive developer community.
  + Extensive documentation and tutorials.
* **Convenience**:
  + Easy to fork, clone, and contribute.
  + GitHub Codespaces for cloud-based development environments.

### Why DevOps is preferred

While GitHub is a powerful platform for version control and collaboration, Azure DevOps offers several advantages in enterprise environments:

| **Feature** | **Azure DevOps** | **GitHub** |
| --- | --- | --- |
| **Integrated CI/CD** | Built-in with Azure Pipelines | Requires GitHub Actions or external tools |
| **Project Management** | Native Boards for Agile/Scrum | GitHub Projects (less mature) |
| **Security & Compliance** | Enterprise-grade, Azure-native | Strong, but less integrated with Azure |
| **Permissions & Policies** | Granular control, AD integration | Simpler, GitHub-based |
| **Artifacts & Testing** | Built-in Artifacts and Test Plans | Requires third-party integrations |

## Implementation:

### Method one (Native Repository Integration in Microsoft Sentinel)

This method uses **Microsoft Sentinel’s built-in support** for syncing content from GitHub or Azure DevOps repositories.

**✅ Prerequisites**

* Owner or equivalent permissions on the Sentinel workspace
* GitHub or Azure DevOps repo with supported content (e.g., analytic rules in JSON or Bicep)
* Repo must be in the **same tenant** as Sentinel

1

**🛠️ Steps to Set Up**

1. **Prepare Your Repository**
   * Organize content in folders like:
   * /AnalyticRules/
   * /Workbooks/
   * /Playbooks/
   * Use supported formats: JSON (for rules), Bicep (for infrastructure)
2. **Go to Microsoft Sentinel**
   * Open the **Microsoft Sentinel** workspace in the Azure or Defender portal
   * Navigate to **Content Management > Repositories**
3. **Create a Repository Connection**
   * Click **+ Add repository**
   * Choose **GitHub** or **Azure DevOps**
   * Authenticate and select:
     + Repo
     + Branch
     + Root folder (e.g., /SentinelContent)
   * Assign a name and click **Create**
4. **Validate and Sync**
   * Sentinel will automatically validate the content
   * If valid, it will deploy the content to your workspace
   * You can monitor sync status and errors in the portal

📘 Full guide: [Deploy custom content from your repository - Microsoft Learn](https://learn.microsoft.com/en-us/azure/sentinel/ci-cd)

### Method two (Sentinel-As-Code with Azure DevOps Pipelines)

1. **Import the Sentinel-As-Code Repo**
   * Use this repo: [Sentinel-As-Code on GitHub](https://github.com/noodlemctwoodle/Sentinel-As-Code)

2

* + In Azure DevOps:
    - Go to **Repos > Import**
    - Paste the GitHub URL or upload the ZIP

1. **Review the Repo Structure**
2. /Bicep/
3. ├── main.bicep
4. └── sentinel.bicep
5. /Scripts/
6. └── Set-SentinelContent.ps1
7. azure-pipelines.yml
8. **Create a Service Connection**
   * Go to **Project Settings > Service Connections**
   * Add an **Azure Resource Manager** connection
   * Use it in the pipeline as azureSubscription
9. **Configure the Pipeline**
   * Go to **Pipelines > New Pipeline**
   * Choose YAML and point to azure-pipelines.yml
   * Customize variables like:
10. **Run the Pipeline**
    * Commit a change or manually trigger the pipeline
    * It will:
      + Deploy infrastructure via Bicep
      + Deploy analytic rules via PowerShell

📘 Full walkthrough: [Automating Microsoft Sentinel Deployment with Azure DevOps](https://sentinel.blog/automating-microsoft-sentinel-deployment/)

### More info about method two

To manage and deploy Microsoft Sentinel content as code, we use **Infrastructure as Code (IaC)** principles, typically with **ARM templates**, **Bicep**, or **Sentinel as Code (SAC)** frameworks. The deployment process involves storing content in a Git repository and using **Azure Pipelines** to automate deployment.

**6.1 Analytics Rules**

**Purpose**: Detect threats and suspicious activities in your environment.

**Deployment Steps**:

* Define rules in JSON format (based on ARM templates or SAC schema).
* Store them in a structured folder (e.g., Sentinel/AnalyticsRules/).
* Use a pipeline task to deploy via:
  + az sentinel alert-rule create (CLI)
  + ARM/Bicep deployment
* Include validation steps to check for syntax and schema compliance.

**Pipeline Example**:

A screen shot of a computer code



**6.2 Automation Rules**

**Purpose**: Automatically respond to incidents by triggering actions like tagging, assigning, or running playbooks.

**Deployment Steps**:

* Define automation rules in JSON format.
* Store in Sentinel/AutomationRules/.
* Deploy using:
  + az sentinel automation-rule create
  + ARM templates

**Best Practice**: Use rule order and conditions to avoid conflicts.

**6.3 Hunting Queries**

**Purpose**: Proactively search for threats using KQL (Kusto Query Language).

**Deployment Steps**:

* Store .kql files in Sentinel/HuntingQueries/.
* Use a script or pipeline task to import them using:
  + az sentinel hunting-rule create
  + REST API or PowerShell

**Tip**: Include metadata like tactics, techniques (MITRE ATT&CK), and description in a YAML or JSON wrapper.

**6.4 Parsers (Custom Logs / Normalization)**

**Purpose**: Normalize and transform raw data into a usable format for Sentinel.

**Deployment Steps**:

* Define parsers as KQL functions.
* Store in Sentinel/Parsers/ as .kql files.
* Deploy using:
  + az monitor log-analytics workspace data-export create
  + REST API for saved functions

**Note**: Ensure parsers are tested in a dev workspace before production deployment.

**6.5 Playbooks**

**Purpose**: Automate incident response using Azure Logic Apps.

**Deployment Steps**:

* Define playbooks as ARM templates or Logic App JSON definitions.
* Store in Sentinel/Playbooks/.
* Deploy using:
  + az deployment group create with ARM template
  + Azure DevOps ARM deployment task

**Pipeline Snippet**:

A screenshot of a computer program

AI-generated content may be incorrect.

**6.6 Workbooks**

**Purpose**: Visualize data and create dashboards for monitoring and investigation.

**Deployment Steps**:

* Define workbooks in JSON format.
* Store in Sentinel/Workbooks/.
* Deploy using:
  + az monitor workbook create
  + ARM templates

**Best Practice**: Parameterize workbook templates for reusability across environments.

### Comparison of both methods:

**Why Use Native Repository Integration (Method 1)?**

**🔹 Pros:**

* **Simple setup**: Just connect your GitHub or Azure DevOps repo in the Sentinel UI.
* **Automatic syncing**: Sentinel pulls in content like analytic rules, workbooks, and playbooks automatically.
* **No pipeline needed**: No need to write YAML, manage service connections, or build CI/CD logic.
* **Great for small to medium teams** or those who want to manage content declaratively.

**🔸 When to Use:**

* You want to **version control** your Sentinel content.
* You don’t need complex deployment logic.
* You’re working in a **single environment** (e.g., just production).

**Why Use Sentinel-As-Code with Azure DevOps Pipelines (Method 2)?**

**🔹 Pros:**

* **Full control**: You can add logic, validations, approvals, and branching strategies.
* **Multi-environment support**: Easily deploy to dev, test, and prod with different parameters.
* **Infrastructure-as-Code**: Deploy not just content, but also the Sentinel workspace, Log Analytics, and more.
* **Custom workflows**: Integrate with other tools, run tests, or notify teams.

**🔸 When to Use:**

* You need **complex deployment scenarios**.
* You’re managing **multiple environments**.
* You want to **automate everything**, including infrastructure.
* You’re part of a **larger SecOps or DevSecOps team** with strict governance.

## Helpful documentation and video links:

[Manage content as code with Microsoft Sentinel repositories (Document)](https://learn.microsoft.com/en-us/azure/sentinel/ci-cd-custom-content)

[Deploying and Managing Microsoft Sentinel as Code (Document)](https://techcommunity.microsoft.com/blog/microsoftsentinelblog/deploying-and-managing-microsoft-sentinel-as-code/1131928)

[Microsoft Sentinel: Repositories, the Future of Content-as-Code & Best Practices (Document)​](https://techcommunity.microsoft.com/blog/microsoftsentinelblog/microsoft-sentinel-repositories-the-future-of-content-as-code--best-practices%E2%80%8B/4422936)

[Deploying and Managing Azure Sentinel as Code (Presentation)](https://onedrive.live.com/?redeem=aHR0cHM6Ly8xZHJ2Lm1zL2IvcyFBbkVQanI4dEhjTm1nbU5RQnlFRm84NUNlYmZU&cid=66C31D2DBF8E0F71&id=66C31D2DBF8E0F71%21355&parId=66C31D2DBF8E0F71%21354&o=OneUp)

[Deploying and Managing Azure Sentinel as Code (Video)](https://www.youtube.com/watch?v=ITZrOiPX3So)

[Deploying Microsoft Sentinel Analytic Rules with Azure DevOps (Video)](https://www.youtube.com/watch?v=6q98XwvmD6Q)

[Sample repo example](https://github.com/SentinelCICD/RepositoriesSampleContent)

[How to deploy Microsoft Sentinel Analytic Rules with Azure DevOps (look at AI overview)](https://www.google.com/search?q=how+to+deploy+Microsoft+Sentinel+Analytic+Rules+with+Azure+DevOps&sca_esv=05f7f9216b2ed83f&rlz=1C1RXQR_enUS1165US1165&ei=Ffh0aMfoE96-p84PvY2SgQg&ved=0ahUKEwiHx83zrLyOAxVe38kDHb2GJIAQ4dUDCBA&uact=5&oq=how+to+deploy+Microsoft+Sentinel+Analytic+Rules+with+Azure+DevOps&gs_lp=Egxnd3Mtd2l6LXNlcnAiQWhvdyB0byBkZXBsb3kgTWljcm9zb2Z0IFNlbnRpbmVsIEFuYWx5dGljIFJ1bGVzIHdpdGggQXp1cmUgRGV2T3BzSIIgUPQJWK8dcAN4AZABAJgBWqAB6gaqAQIxNbgBA8gBAPgBAfgBApgCEaAChgfCAgoQABiwAxjWBBhHwgILEAAYgAQYkQIYigXCAhEQLhiABBixAxjRAxiDARjHAcICDhAAGIAEGLEDGIMBGIoFwgIOEC4YgAQYsQMY0QMYxwHCAgsQABiABBixAxiDAcICBRAAGIAEwgIOEC4YgAQYsQMYgwEYigXCAggQABiABBixA8ICChAAGIAEGEMYigXCAhAQLhiABBjRAxhDGMcBGIoFwgILEC4YgAQYsQMY1ALCAgUQLhiABMICCxAAGIAEGLEDGIoFwgILEC4YgAQYsQMYgwGYAwCIBgGQBgiSBwIxN6AHoXiyBwIxNLgH-gbCBwYwLjIuMTXIB0s&sclient=gws-wiz-serp)

[Microsoft Security Community webinars (Document)](https://techcommunity.microsoft.com/blog/microsoft-security-blog/recordings--security-community-webinars/2865990)