AZ-2001: Implement security through a pipeline using Azure DevOps

Content

Module 1: Configure a Project and Repository Structure to Support Secure Pipelines

- **Lessons:**
  - Separate a project into team projects and repositories
  - Separate secure files between projects
  - Move the security repository away from a project
  - Assign project and repository permissions
  - Organize a project and repository structure

Module 2: Configure Secure Access to Pipeline Resources

- **Lessons:**
  - Identify and mitigate common security threats
  - Configure pipeline access to specific agent pools
  - Manage secret variables and variable groups
  - Secure files and storage
  - Configure service connections
  - Manage environments
  - Secure repositories

Module 3: Manage Identity for Projects, Pipelines, and Agents

- **Lessons:**
  - Configure a Microsoft-hosted pool
  - Configure agents for projects
  - Configure agent identities
  - Configure the scope of a service connection
  - Convert to a managed identity in Azure DevOps

Module 4: Configure and Validate Permissions

- **Lessons:**
  - Configure and validate user permissions
  - Configure and validate pipeline permissions
  - Configure and validate approval and branch checks
  - Manage and audit permissions in Azure DevOps

Module 5: Extend a Pipeline to Use Multiple Templates

- **Lessons:**
  - Create nested templates
  - Rewrite the main deployment pipeline
  - Configure the pipeline and the application to use tokenization
  - Remove plain text secrets
  - Restrict agent logging
  - Identify and conditionally remove script tasks in Azure DevOps

---

Module 6: Configure Secure Access to Azure Repos from Pipelines

- **Lessons:**
  - Configure pipeline access to packages
  - Configure credential secrets, and secrets for services
  - Ensure that the secrets are in the Azure Key Vault
  - Ensure that secrets aren't in the logs

Module 7: Configure Pipelines to Securely Use Variables and Parameters

- **Lessons:**
  - Ensure that parameters and variables retain their type
  - Identify and restrict insecure use of parameters and variables
  - Move parameters into a YAML file that protects their type
  - Limit variables that can be set at queue time
  - Validate that mandatory variables are present and set correctly in Azure DevOps